

# Module 13

## Hacking Web Servers

### Hacking Web Servers: An Overview

Hacking a web server refers to exploiting vulnerabilities in the server's software, configurations, or the applications it hosts to gain unauthorized access or disrupt services. Understanding how attacks work helps in building secure systems. Below are the key concepts and common methods involved:

---

#### Common Vulnerabilities:

##### 1. SQL Injection (SQLi):

- Attackers manipulate SQL queries through user inputs (like forms or URLs).
- Can lead to data breaches, authentication bypass, or data corruption.

##### 2. Cross-Site Scripting (XSS):

- Malicious scripts are injected into web pages viewed by other users.
- Often used to steal session cookies, hijack accounts, or deliver malware.

##### 3. Cross-Site Request Forgery (CSRF):

- Tricks a user into executing unwanted actions while authenticated on a website.
- Example: Submitting a form or making a purchase without consent.

##### 4. Remote Code Execution (RCE):

- Attackers execute arbitrary code on the server.
- This can lead to complete control over the server.

##### 5. Directory Traversal:

- Manipulating paths to access restricted directories and files.
- Example: <http://example.com/page=../../etc/passwd>.

## 6. Server Misconfiguration:

- Default credentials, open ports, and unpatched software can be exploited.
  - Tools like **Nmap** and **Nikto** are used to discover these weaknesses.
- 

## 2 Tools Commonly Used for Hacking Web Servers:

- **Nmap**: For network discovery and security auditing.
  - **Nikto**: Web server scanner for vulnerabilities.
  - **Burp Suite**: Used for testing web application security.
  - **OWASP ZAP**: Open-source tool for finding security vulnerabilities.
  - **Metasploit**: Framework for developing and executing exploit code.
- 

## 3 Techniques Employed:

### 1. Footprinting and Scanning:

- Identifying the server, technologies used, and open ports.

### 2. Enumeration:

- Extracting usernames, paths, and configuration details.

### 3. Exploitation:

- Using discovered vulnerabilities to inject code or manipulate data.

### 4. Privilege Escalation:

- Gaining higher access rights after initial exploitation.

### 5. Covering Tracks:

- Clearing logs, hiding backdoors to maintain access.
- 

## 4 Mitigation Strategies:

- **Update and Patch Regularly**: Keep server software and libraries up-to-date.
- **Least Privilege Principle**: Limit access rights to only what's necessary.
- **Web Application Firewall (WAF)**: Protects against common attacks like SQLi and XSS.
- **Input Validation and Sanitization**: Prevent injection attacks.

- **Secure Configuration:** Disable unnecessary features and services.

## Real-World Example: Heartbleed Vulnerability (CVE-2014-0160)

---

### **Background:**

Heartbleed was a major vulnerability discovered in **OpenSSL**, a widely-used library for securing communications over the internet. It affected millions of web servers, including major platforms like Yahoo, GitHub, and many others.

**OpenSSL** uses encryption protocols (like HTTPS, SSL, and TLS) to secure data. Heartbleed specifically targeted the **Heartbeat extension** of OpenSSL, which is meant to keep secure connections alive.

---

### **How the Attack Worked:**

#### 1. **Heartbeat Extension:**

- OpenSSL has a "heartbeat" feature where a client sends a small packet to the server, requesting a response to confirm the connection is still alive.
- The client sends some data (e.g., "ping") along with the length of that data (e.g., 4 bytes).

#### 2. **The Vulnerability:**

- Attackers could manipulate the length of the data packet they claimed to send.
- For example, they would send the word "ping" (4 bytes) but say the length was **64 KB**.
- OpenSSL would then respond with 64 KB of memory data, even if the original message was just 4 bytes.

#### 3. **Data Exposure:**

- This extra memory data was random but could contain sensitive information like:
  - **Usernames and passwords**
  - **Private server keys**
  - **Session cookies**
- This allowed attackers to capture critical information **without detection**.



### Tools Used to Exploit Heartbleed:

- **Nmap (with Heartbleed module):** Scans for vulnerable servers.
  - **Metasploit:** Automated exploitation of the vulnerability.
  - **Heartbleed.py:** A simple Python script to extract data from the server's memory.
- 



### How It Was Fixed:

- OpenSSL released a patch (**OpenSSL 1.0.1g**) that corrected the memory handling.
  - Websites were forced to:
    - Update their OpenSSL libraries.
    - **Regenerate SSL certificates** (since private keys might have been exposed).
    - **Invalidate old sessions** and force users to re-login.
- 



### Lessons Learned:

1. **Patch Early, Patch Often:** Delaying updates creates attack windows.
  2. **Memory Management Matters:** Careless handling of memory can expose sensitive data.
  3. **Monitor Logs for Anomalies:** Massive data leaks often leave trails if monitored.
- 

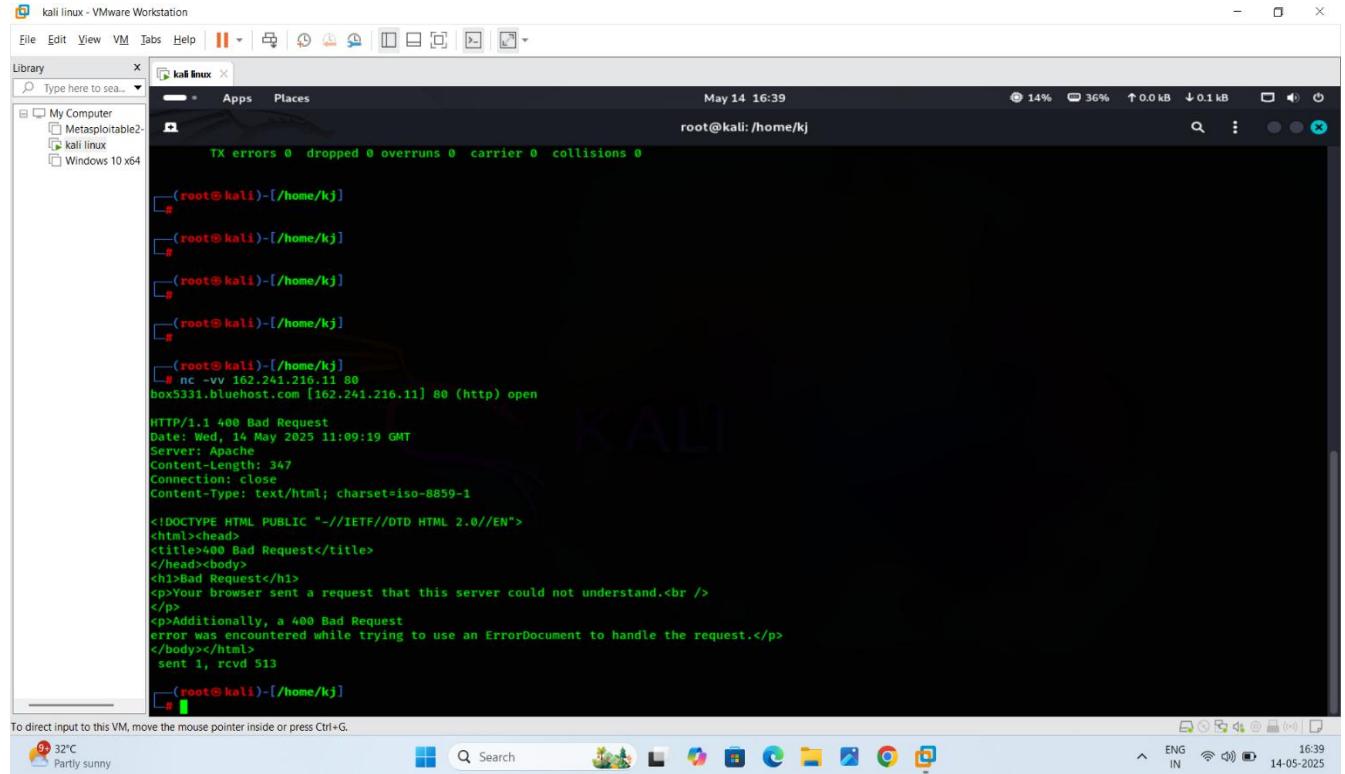
☞ Here are some command line that useful for scanning for hacking web servers

Name : kunal Jawale

## Commands :

**1. # nc -vv <Target IP address> 80**

This command is used for check service or server



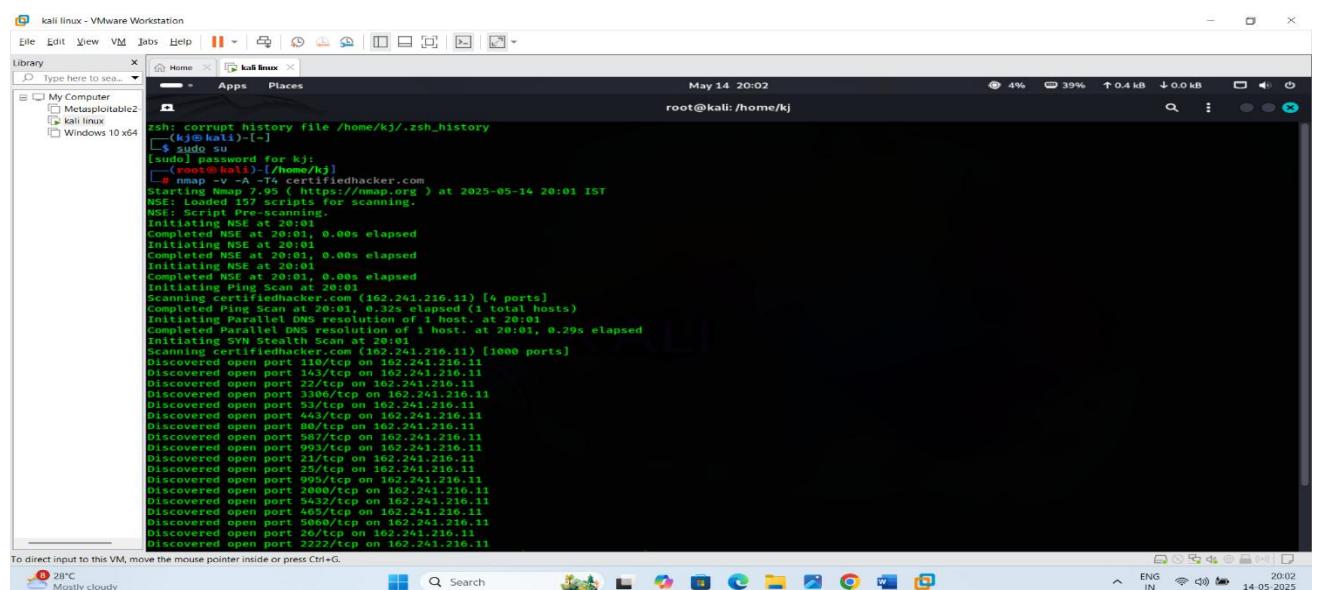
```
TX errors 0  dropped 0  overruns 0  carrier 0  collisions 0

[root@kali ~]#
[root@kali ~]#
[root@kali ~]#
[root@kali ~]#
[root@kali ~]#
[root@kali ~]# nc -vv 162.241.216.11 80
nc: connect to 162.241.216.11 [162.241.216.11] 80 (http) open
HTTP/1.1 400 Bad Request
Date: Wed, 14 May 2025 11:09:19 GMT
Server: Apache
Content-Length: 347
Content-Type: text/html; charset=iso-8859-1

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>400 Bad Request</title>
</head><body>
<h1>Bad Request</h1>
<p>Your browser sent a request that this server could not understand.<br />
</p>
<p>Additionally, a 400 Bad Request<br />
error was encountered while trying to use an ErrorDocument to handle the request.</p>
</body></html>
sent 1, rcvd 513
```

**2. # nmap -v -A -T4 <Target website /IP>**

This command give you all information about the target



```
sh: corrupt history file /home/kj/.zsh_history
[root@kali ~]#
[root@kali ~]# su
[sudo] password for kj:
[root@kali ~]#
[root@kali ~]# nmap -v -A -T4 certifiedhacker.com
Starting Nmap 8.0.0 ( https://nmap.org ) at 2025-05-14 20:01 IST
Nmap: Loaded 157 scripts for scanning.
NSE: Script Pre-scanning
Initiating NSE at 20:01
Completed NSE at 20:01, 0.00s elapsed
Initiating Ping Scan at 20:01
Completed Ping Scan at 20:01, 0.32s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:01
Completed Parallel DNS resolution of 1 host. at 20:01, 0.29s elapsed
Initiating SYN Stealth Scan at 20:01
Scanning certifiedhacker.com (162.241.216.11) [4 ports]
Completed Ping Scan at 20:01, 0.32s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host. at 20:01
Completed Parallel DNS resolution of 1 host. at 20:01, 0.29s elapsed
Initiating SYN Stealth Scan at 20:01
Scanning certifiedhacker.com (162.241.216.11) [1000 ports]
Discovered open port 118/tcp on 162.241.216.11
Discovered open port 123/tcp on 162.241.216.11
Discovered open port 22/tcp on 162.241.216.11
Discovered open port 3306/tcp on 162.241.216.11
Discovered open port 53/tcp on 162.241.216.11
Discovered open port 6000/tcp on 162.241.216.11
Discovered open port 80/tcp on 162.241.216.11
Discovered open port 88/tcp on 162.241.216.11
Discovered open port 987/tcp on 162.241.216.11
Discovered open port 993/tcp on 162.241.216.11
Discovered open port 995/tcp on 162.241.216.11
Discovered open port 2000/tcp on 162.241.216.11
Discovered open port 5423/tcp on 162.241.216.11
Discovered open port 5631/tcp on 162.241.216.11
Discovered open port 5660/tcp on 162.241.216.11
Discovered open port 26/tcp on 162.241.216.11
Discovered open port 2223/tcp on 162.241.216.11
```

Name : kunal Jawale

```
F-BProgNig, 85, "E:\0\0\0\x845FATAL\0C0A000\0Unsupported\x20frontend\x20proto
F:tcp[0x00536],19778,x20server\x20supports\x201.0\x20to\x203.0\0po
F:simmsmaster\c\0L1811\0RProcessStartupPacket\0\0\rKerberos,45,\0\0\0\0
F:x845FATAL\0C0A000\0Unsupported\x20frontend\x20proto\0x027265\,28288
F:x20server\x20supports\x201.0\x20to\x203.0\0postmaster\c\0L1811\0R
F:processStartupPacket\0\0\r
device type: VoIP endpoint|general purpose|load balancer|firewall|printer
running (JUST GUESSING): Cisco embedded (91%), Linux 2.6.X|5.X (91%), F5 Networks embedded (88%), Netasq embedded (86%), Sun OpenSolaris (86%), Dell embedded (85%), Fortinet embedded (85%), Lexmark embedded (83%)
OS CPE: cpe:/h:cisco:unified_call_manager cpe:/o:linux:linux_kernel:2.6.26 cpe:/o:linux:linux_kernel:5.18 cpe:/h:netasq:u70 cpe:/o:sun:opensolaris cpe:/h:dell
1720dm cpe:/h:fortinet:fortigate_100d cpe:/h:lexmark:z2400
aggressive OS guesses: Cisco Unified Communications Manager VoIP adapter (91%), Linux 2.6.20 (91%), Linux 5.18 (91%), Linux 5.4 (91%), F5 BIG-IP Edge Gateway (88%), Netasq U70 Firewall (86%), Sun OpenSolaris 2009.06 (86%), Dell 1720dm printer (85%), Fortinet FortiGate 100D firewall (85%), Fortinet FortiGate 1500B firewall (85%)
no exact OS matches for host (test conditions non-ideal).
uptime guess: 13.474 days (since Thu May 1 08:40:34 2025)
network Distance: 25 hops
Service Info: OS: Linux; CPE: cpe:/o:redhat:enterprise_linux:7

TRACEROUTE (using port 5900/tcp)
HOP RTT ADDRESS
1 13.35 ms 192.168.175.177
2 7.69 ms 192.0.0.1
3 ... 24
25 323.36 ms box5331.bluehost.com (162.241.216.11)

NSE: Script Post-scanning.
Initiating NSE at 20:03
Completed NSE at 20:03, 0.00s elapsed
Initiating NSE at 20:03
Completed NSE at 20:03, 0.00s elapsed
Initiating NSE at 20:03
Completed NSE at 20:03, 0.00s elapsed
Read data files from: /usr/share/nmap
OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 70.32 seconds
Raw packets sent: 1229 (58.046KB) | Rcvd: 1087 (44.426KB)

[root@kali]~[/home/kj]
#
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

28°C Mostly cloudy 20:03 ENG IN 14-05-2025

### 3. # Nmap -v --script http-trace <target IP/website>

This triggers the **http-trace** NSE (Nmap Scripting Engine) script. The **http-trace** script checks if the **HTTP TRACE method** is enabled on the target web server

```
# nmap -v --script http-trace certifiedhacker.com
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-14 16:43 IST
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 16:43
Completed NSE at 16:43, 0.00s elapsed
Initiating Ping Scan at 16:43
Scanning certifiedhacker.com (162.241.216.11) [4 ports]
Completed Ping Scan at 16:43, 0.29s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host, at 16:43
Completed Parallel DNS resolution of 1 host. at 16:43, 0.02s elapsed
Initiating SYN Stealth Scan at 16:43
Scanning certifiedhacker.com (162.241.216.11) [1000 ports]
Discovered open port 587/tcp on 162.241.216.11
Discovered open port 3306/tcp on 162.241.216.11
Discovered open port 995/tcp on 162.241.216.11
Discovered open port 22/tcp on 162.241.216.11
Discovered open port 21/tcp on 162.241.216.11
Discovered open port 80/tcp on 162.241.216.11
Discovered open port 143/tcp on 162.241.216.11
Discovered open port 110/tcp on 162.241.216.11
Discovered open port 53/tcp on 162.241.216.11
Discovered open port 443/tcp on 162.241.216.11
Discovered open port 993/tcp on 162.241.216.11
Discovered open port 25/tcp on 162.241.216.11
Discovered open port 26/tcp on 162.241.216.11
Discovered open port 2222/tcp on 162.241.216.11
Discovered open port 465/tcp on 162.241.216.11
Discovered open port 5432/tcp on 162.241.216.11
Completed SYN Stealth Scan at 16:43, 16.72s elapsed (1000 total ports)
NSE: Script scanning 162.241.216.11.
Initiating NSE at 16:43
Completed NSE at 16:43, 1.43s elapsed
Nmap scan report for certifiedhacker.com (162.241.216.11)
Host is up (0.29s latency).
DNS record for 162.241.216.11: box5331.bluehost.com
Not shown: 981 closed tcp ports (reset)
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

32°C Partly sunny 16:44 ENG IN 14-05-2025

Name : kunal Jawale

**4. # nmap -v -- script http-waf-detect <target IP/website>**

This command is used for check firewall,IDS is present or not

kali linux - VMware Workstation

File Edit View VM Tabs Help | Library

Type here to search

My Computer Metasploitable2 kali Linux Windows 10 x64

kali linux

Apps Places May 14 16:45 root@kali: /home/kj

```
[root@kali]~[~/home/kj]
# nmap -v --script http-waf-detect testfire.net
Starting Nmap 7.95 ( https://nmap.org ) at 2025-05-14 16:44 IST
NSE: Loaded 1 scripts for scanning.
NSE: Script Pre-scanning.
Initiating NSE at 16:44
Completed NSE at 16:44, 0.00s elapsed
Initiating Ping Scan at 16:44
Scanning testfire.net (65.61.137.117) [4 ports]
Completed Ping Scan at 16:44, 0.30s elapsed (1 total hosts)
Initiating Parallel DNS resolution of 1 host at 16:44
Completed Parallel DNS resolution of 1 host at 16:44, 0.29s elapsed
Initiating SYN Stealth Scan at 16:44
Scanning testfire.net (65.61.137.117) [1000 ports]
Discovered open port 8080/tcp on 65.61.137.117
Discovered open port 80/tcp on 65.61.137.117
Discovered open port 443/tcp on 65.61.137.117
Completed SYN Stealth Scan at 16:45, 21.47s elapsed (1000 total ports)
NSE: Script scanning 65.61.137.117.
Initiating NSE at 16:45
Completed NSE at 16:45, 2.91s elapsed
Nmap scan report for testfire.net (65.61.137.117)
Host is up (0.30s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT      STATE SERVICE
80/tcp    open  http
| http-waf-detect: IDS/IPS/WAF detected:
|_testfire.net:80/?pyle0d3=<script>alert(document.cookie)</script>
443/tcp   open  https
8080/tcp  open  http/proxy
| http-waf-detect: IDS/IPS/WAF detected:
|_testfire.net:8080/?pyle0d3=<script>alert(document.cookie)</script>
8443/tcp  closed https-alt

NSE: Script Post-scanning.
Initiating NSE at 16:45
Completed NSE at 16:45, 0.00s elapsed
Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 25.35 seconds
```

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.

32°C Partly sunny

ENG IN 14-05-2025

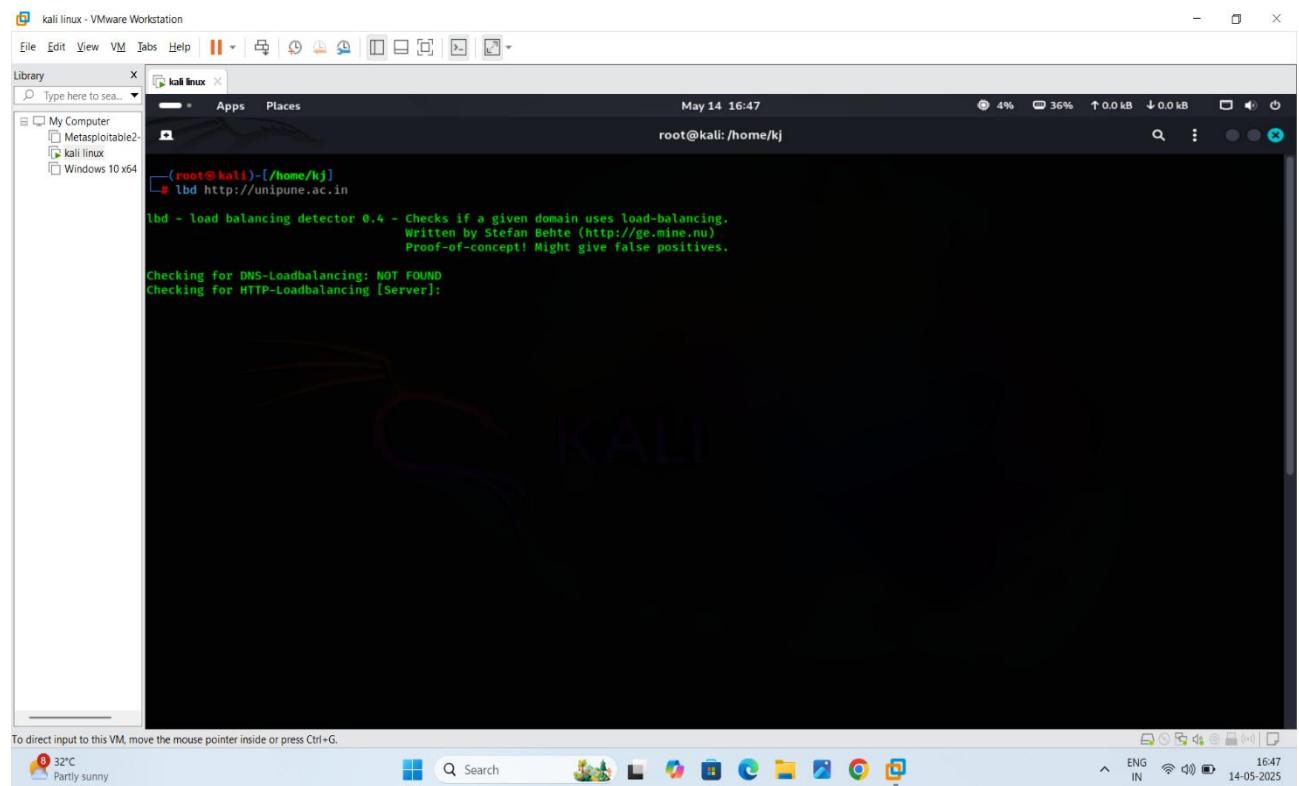
5. # wafwoof <targetIP/website>

This command is check the firewall is present or not

Name : kunal Jawale

## 6. # lbd <target IP/website>

This command is for detect load balancer

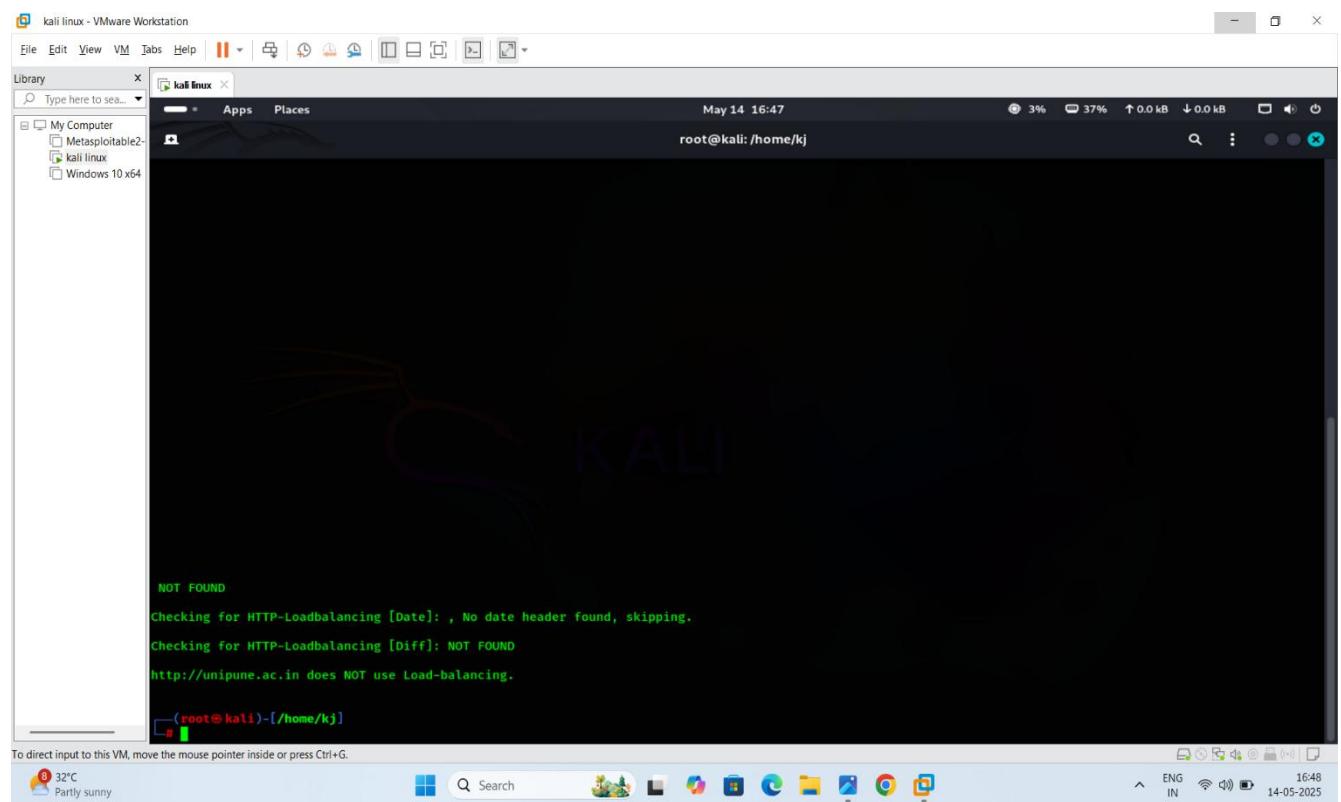


The screenshot shows a terminal window titled "kali linux" running on a Kali Linux virtual machine in VMware Workstation. The terminal displays the output of the "lbd" command, which is used to detect load balancing. The command is run as root on the Kali Linux host, targeting the website "http://unipune.ac.in". The output indicates that no load balancing was found.

```
(root@kali)-[~/home/kj]
# lbd http://unipune.ac.in

lbd - load balancing detector 0.4 - Checks if a given domain uses load-balancing.
Written by Stefan Behte (http://ge.mine.nu)
Proof-of-concept! Might give false positives.

Checking for DNS-Loadbalancing: NOT FOUND
Checking for HTTP-Loadbalancing [Server]:
```



The screenshot shows a terminal window titled "kali linux" running on a Kali Linux virtual machine in VMware Workstation. The terminal displays the output of the "lbd" command, targeting the website "http://unipune.ac.in". The output indicates that no load balancing was found, specifically noting that no date header was found, which caused the check to skip.

```
NOT FOUND
Checking for HTTP-Loadbalancing [Date]: , No date header found, skipping.
Checking for HTTP-Loadbalancing [Diff]: NOT FOUND
http://unipune.ac.in does NOT use Load-balancing.

[root@kali]-[~/home/kj]
```

# / Hydra for hacking web server

Hydra (also known as **THC-Hydra**) is a popular password-cracking tool used for brute-force attacks against various protocols and services. Its primary purpose is to **guess valid login credentials (username and password)** for services running on web servers and other networked applications.

## Common Uses of Hydra in Hacking Web Servers:

### 1. Brute-Force Login Forms:

- Hydra can attempt multiple username and password combinations against web login forms.
- It supports different authentication methods such as **HTTP Basic Auth, HTTP Form-based Auth, and Digest Auth.**

### 2. Attacking Web Applications:

- Hydra can be used to brute-force login pages of web applications like **WordPress, Joomla, and Drupal.**
- Supports web interfaces that require login via POST or GET requests.

### 3. Testing Web APIs:

- Can be configured to brute-force API endpoints that require authentication tokens or basic authentication.

### 4. FTP, SSH, Telnet Services:

- Hydra is not limited to just HTTP(S); it can also target services like **FTP, SSH, Telnet, MySQL**, and more.
- This extends its ability to attack web servers with exposed administrative protocols.

### 5. Multi-threaded Attacks:

- Hydra supports multi-threading, enabling it to perform many attempts simultaneously, significantly speeding up brute-force attacks.

### 6. Custom Parameters and Payloads:

- Supports custom parameters, cookies, headers, and payloads for evading WAFs (Web Application Firewalls) and intrusion detection systems.

---

## Example of a Hydra Command for HTTP Form Login:

```
hydra -l admin -P /path/to/passwords.txt <target_ip> http-post-form  
"/login.php:username=^USER^&password=^PASS^:Invalid Login"
```

- -l admin: Specifies the username to test (admin).
  - -P /path/to/passwords.txt: Path to the password list.
  - <target\_ip>: IP address of the target server.
  - http-post-form: Specifies form-based login.
  - "/login.php:username^USER^&password^PASS^:Invalid Login":
    - /login.php: Target login form.
    - username and password: Field names in the form.
    - Invalid Login: String returned when login fails.
- 

## Important Considerations:

1. **Legality:** Hydra should only be used for ethical hacking, penetration testing, or with **explicit permission**.
  2. **Account Lockout Risks:** Many web servers are configured to **lock accounts** after multiple failed attempts.
  3. **Detection:** Brute-force attacks are noisy and can easily trigger **IDS/IPS alerts**.
  4. **Rate Limiting and CAPTCHA:** Modern web applications often have **rate limiting, CAPTCHAs, and 2FA**, making brute-force less effective.
- 



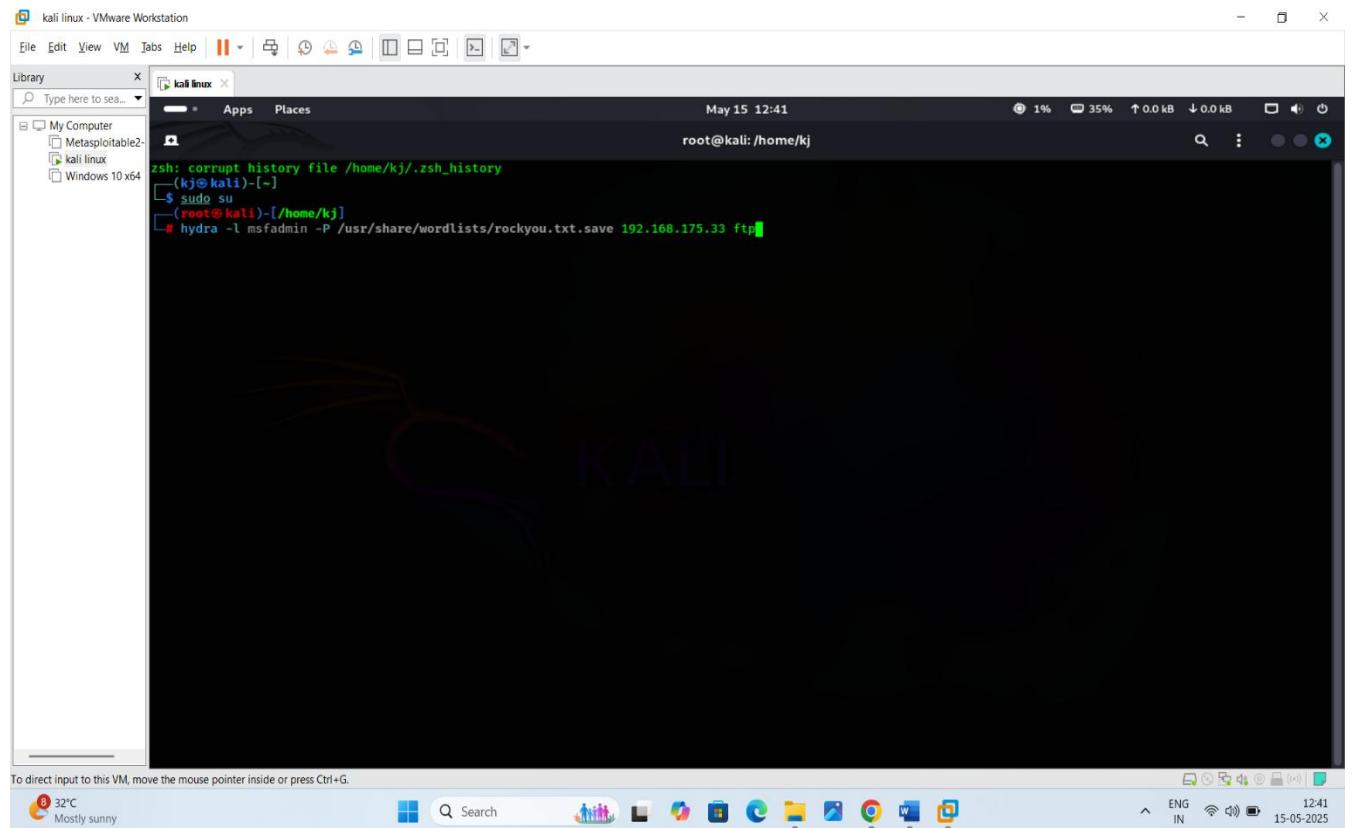
## metasploitable password cracking using hydra brute force attack.

**Command :-** # hydra -l msfadmin -p /usr/share/wordlists/rockyou.txt <metasploitable IP> ftp

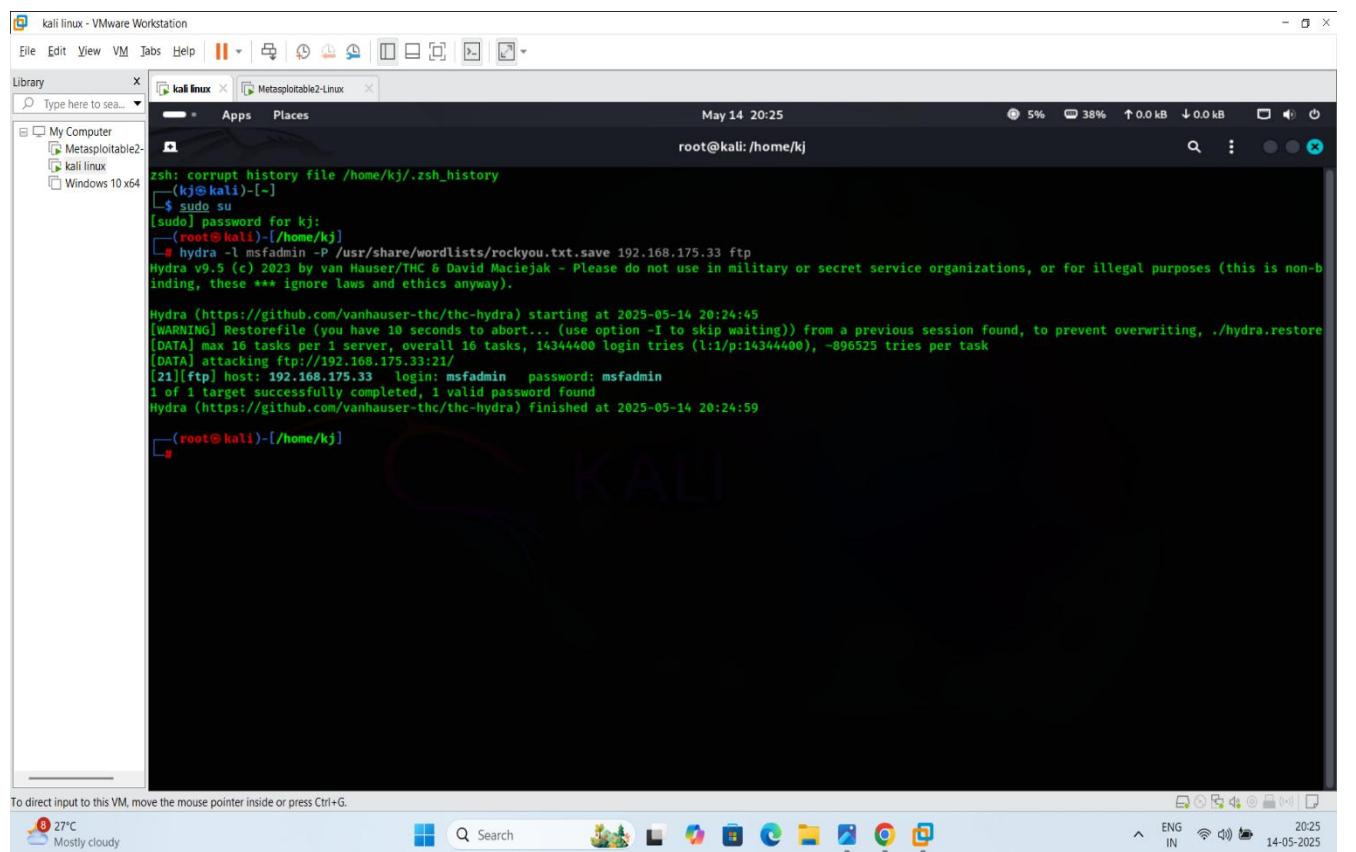
-l for username > if you don't know the username go for capital L and anywordlist like rockyou.txt

-P for password > if you know the password go for small -p and mention password

Name : kunal Jawale



In the following screenshot u can see using hydra tool we crack metasploitable password using rockyou.txt wordlist :



# / Skipfish

is an **automated web application security reconnaissance tool**. It is primarily used for **crawling and scanning web applications** to identify vulnerabilities. Developed by Google, Skipfish is efficient and fast, making it a popular choice in penetration testing and web application security assessments.

---

## Primary Uses of Skipfish in Hacking Web Servers:

### 1. Web Application Crawling:

- Skipfish aggressively crawls the target web server to **map its structure**, discovering hidden directories, files, and endpoints.
- This process helps identify **attack surfaces** that may not be visible during manual inspection.

### 2. Vulnerability Scanning:

- It scans for common vulnerabilities such as:
  - **SQL Injection**
  - **Cross-Site Scripting (XSS)**
  - **File Inclusion Vulnerabilities**
  - **Directory Traversal**
  - **Authentication Bypass**
- It checks for **misconfigurations** and **default credentials** that could be exploited.

### 3. Wordlist-based Testing:

- Skipfish uses **wordlists** to perform directory and file enumeration.
- Custom wordlists can be supplied to extend its reach and improve the depth of the scan.

### 4. SSL/TLS Scanning:

- It identifies issues with SSL/TLS configurations, like **insecure ciphers** or **expired certificates**.

### 5. Session Management and Cookies Analysis:

- Detects **insecure session handling**, including weak cookie configurations and session ID predictability.

### 6. Reporting and Analysis:

- Generates detailed **HTML reports** that are easy to navigate, with information about:
  - The discovered URLs and endpoints.
  - Types of vulnerabilities and their locations.

- Suggested remediations.
- 

### **Example of a Skipfish Command:**

```
skipfish -o /path/to/report -W wordlist.txt http://example.com
```

- **-o /path/to/report:** Specifies the output directory for the HTML report.
  - **-W wordlist.txt:** Uses a custom wordlist for directory and file discovery.
  - **http://example.com:** The target web application.
- 

### **Advantages of Skipfish:**

1. **High Speed & Efficiency:** Multi-threaded for fast crawling and scanning.
2. **Low Bandwidth Consumption:** Optimized to consume minimal bandwidth during scans.
3. **Customizable Wordlists:** Allows deeper and more targeted scans with custom wordlists.
4. **Detailed Reporting:** Provides interactive HTML reports for easy vulnerability analysis.
5. **False Positive Control:** Includes mechanisms to minimize false positives during scanning.

### **Limitations to Keep in Mind:**

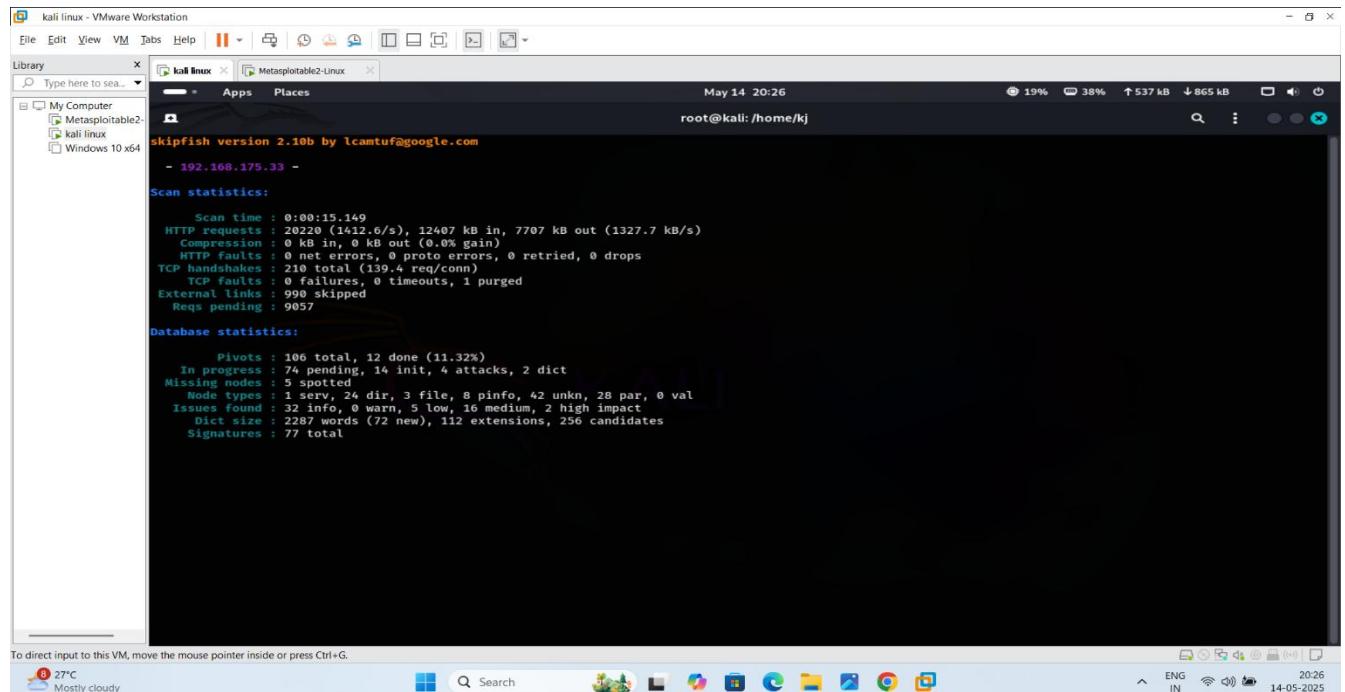
1. **Outdated Development:** Skipfish is no longer actively maintained, *which means it might miss newer vulnerabilities.*
  2. **Limited Custom Scripts:** Unlike tools like **Burp Suite**, it has *limited support for custom scripting.*
  3. **Detection by WAFs:** Many modern **Web Application Firewalls (WAFs)** can detect and block Skipfish scans.
- 

## **Command for detect vulnerabilities in metasploitable using Skipfish :**

Name : kunal Jawale

```
# skipfish -o /home/kj/go -S /usr/share/skipfish/dictionaries/complete.wl/
http://<metasploitable IP>
```

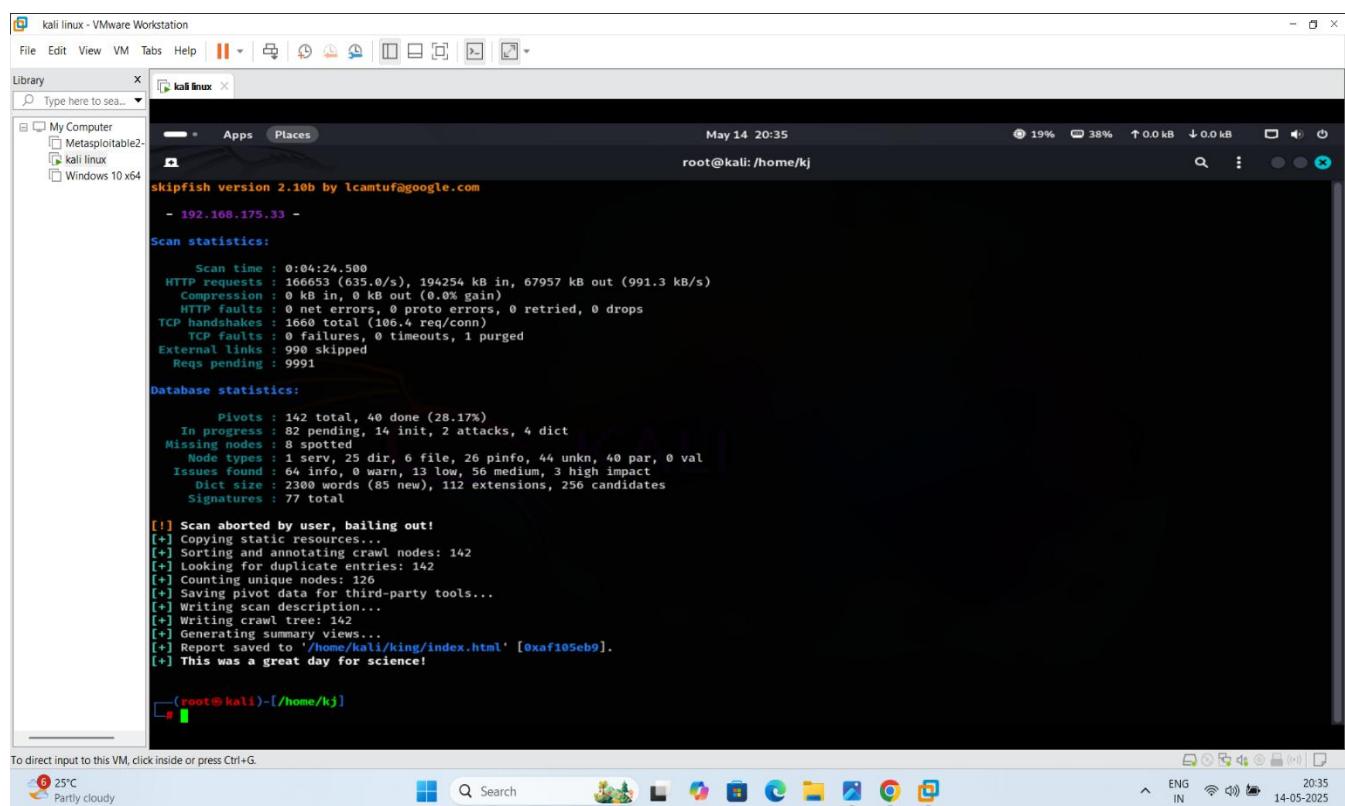
- **-o /home/kj/go** : Specifies the output directory for the HTML report.



```
skipfish version 2.10b by lcamtuf@google.com
- 192.168.175.33 -
Scan statistics:
  Scan time : 0:00:15.149
  HTTP requests : 202407 (1412.0/s), 12407 kB in, 7707 kB out (1327.7 kB/s)
  Compression : 0 kB in, 0 kB out (0.0% gain)
  HTTP Faults : 0 net errors, 0 proto errors, 0 retried, 0 drops
  TCP handshakes : 210 total (139.4 req/conn)
  TCP faults : 0 failures, 0 timeouts, 1 purged
  External links : 990 skipped
  Reqs pending : 9057

Database statistics:
  Pivots : 106 total, 12 done (11.32%)
  In progress : 74 pending, 14 init, 4 attacks, 2 dict
  Missing nodes : 8 spotted
  Node types : 1 serv, 24 dir, 3 file, 8 pinfo, 42 unkn, 28 par, 0 val
  Issues found : 32 info, 0 warn, 5 low, 16 medium, 2 high impact
  Dict size : 2287 words (72 new), 112 extensions, 256 candidates
  Signatures : 77 total

To direct input to this VM, move the mouse pointer inside or press Ctrl+G.
```



```
skipfish version 2.10b by lcamtuf@google.com
- 192.168.175.33 -
Scan statistics:
  Scan time : 0:04:24.500
  HTTP requests : 166653 (635.0/s), 194254 kB in, 67957 kB out (991.3 kB/s)
  Compression : 0 kB in, 0 kB out (0.0% gain)
  HTTP Faults : 0 net errors, 0 proto errors, 0 retried, 0 drops
  TCP handshakes : 1660 total (106.4 req/conn)
  TCP faults : 0 failures, 0 timeouts, 1 purged
  External links : 990 skipped
  Reqs pending : 9991

Database statistics:
  Pivots : 142 total, 40 done (28.17%)
  In progress : 82 pending, 14 init, 2 attacks, 4 dict
  Missing nodes : 8 spotted
  Node types : 1 serv, 25 dir, 6 file, 26 pinfo, 44 unkn, 40 par, 0 val
  Issues found : 64 info, 0 warn, 13 low, 56 medium, 3 high impact
  Dict size : 2300 words (85 new), 112 extensions, 256 candidates
  Signatures : 77 total

[!] Scan aborted by user, bailing out!
[+] Copying static resources...
[+] Sorting and annotating crawl nodes: 142
[+] Looking for duplicate entries: 142
[+] Counting unique nodes: 126
[+] Saving pivot data for third-party tools...
[+] Writing scan description...
[+] Writing crawl tree: 142
[+] Generating summary views...
[+] Report saved to '/home/kali/king/index.html' [0xaf105eb9].
[+] This was a great day for science!

[root@kali]-(~/home/kj) #
```

*Skipfish is best scanning tool skipfish do all types of scanning .*

# / Armitage

## What is Armitage?

**Armitage** is a graphical cyber attack management tool for the **Metasploit Framework**. It allows for **visualizing targets, managing exploits, and performing network-based attacks** with an easy-to-use interface. Armitage was designed to make Metasploit more accessible and collaborative for penetration testers.

---

## Primary Uses of Armitage in Hacking Web Servers:

### 1. Target Discovery & Scanning:

- Armitage can automatically **discover live hosts, open ports, and running services** on a network.
- It uses Nmap and other scanners to map the attack surface before launching attacks.

### 2. Exploitation of Web Server Vulnerabilities:

- Provides a simple interface to launch **Metasploit exploits** against identified web server vulnerabilities.
- Supports attacks on:
  - **Apache, Nginx, IIS** (Web servers)
  - **PHP, ASP.NET** (Web technologies)
  - **Databases (MySQL, MSSQL)** linked to web applications

### 3. Web Application Attacks:

- You can execute attacks like:
  - **SQL Injection**
  - **Remote Code Execution (RCE)**
  - **File Upload Vulnerabilities**
  - **Remote File Inclusion (RFI)**
- Armitage leverages Metasploit's payloads to gain shell access or meterpreter sessions.

### 4. Post-Exploitation Management:

- Once access is gained, Armitage allows you to:
  - **Browse files and directories**
  - **Dump password hashes**
  - **Capture keystrokes**

- **Escalate privileges**

- It supports **pivoting**, which means using a compromised host to attack deeper into the network.

**5. Team Collaboration:**

- Armitage allows multiple users to **collaborate in real-time** during penetration tests.
- Attackers can share sessions, communicate through the platform, and divide tasks efficiently.

**6. Automated Attack Scripts:**

- It includes **Hail Mary**, an automated attack option that attempts multiple exploits to find vulnerabilities quickly.
- 

**Example Workflow in Armitage:**

**1. Scan the Network:**

- Use built-in Nmap scanning to find live hosts and open ports.

**2. Identify Vulnerabilities:**

- Armitage suggests possible exploits based on detected services and versions.

**3. Launch Exploits:**

- Select an exploit, configure payloads, and launch it against the target.

**4. Post-Exploitation:**

- Gain shell access, escalate privileges, and explore the target.

**5. Data Extraction and Cleanup:**

- Extract sensitive data and clear logs if necessary.
- 

**Advantages of Armitage:**

- **Graphical Interface:** Easier to use compared to command-line Metasploit.
  - **Collaboration Support:** Multiple team members can work on the same project.
  - **Real-time Visibility:** Live view of attacks, sessions, and compromised hosts.
  - **Metasploit Integration:** Complete integration with Metasploit, extending its power.
-

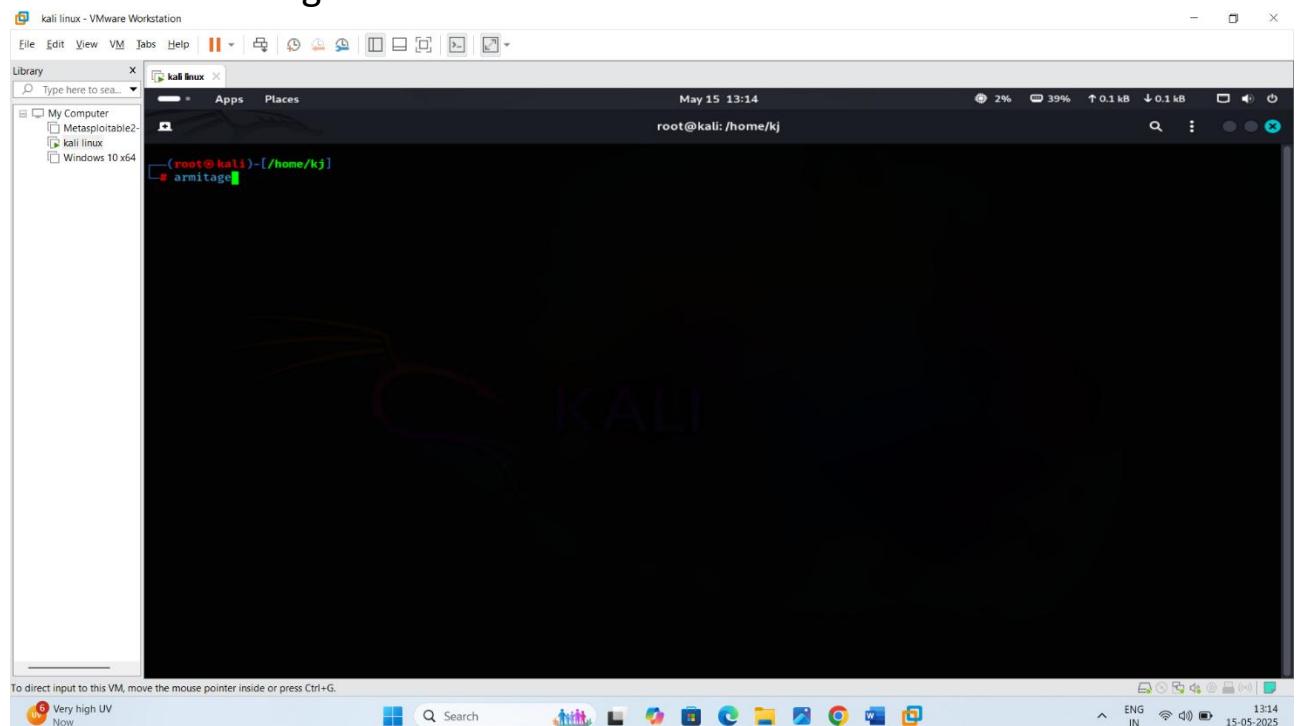
## Limitations:

- Resource Intensive:** Requires significant CPU and memory resources to run smoothly.
  - No Longer Actively Maintained:** As of recent years, Armitage is not actively updated, though Metasploit still is.
  - Detection Risk:** Scans and exploits are easily detectable by Intrusion Detection Systems (IDS) and Web Application Firewalls (WAFs).
  - Limited Web App Features:** Not as focused on web app vulnerabilities as tools like **Burp Suite** or **OWASP ZAP**.
- 

## 👉 Use Armitage for get access of Metasploitable Server :

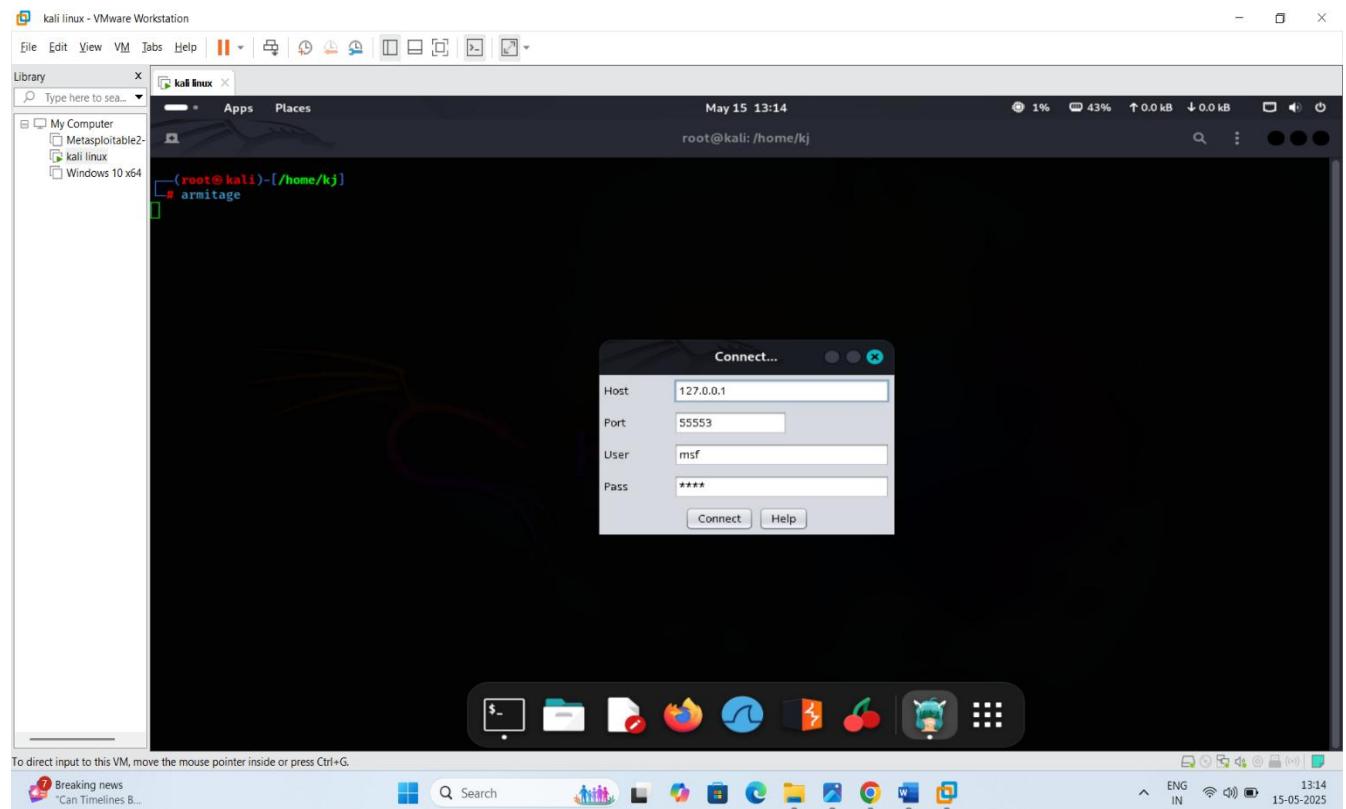
Steps :

1. On kali machine and metasploitable2
2. Run Armitage command in kali linux

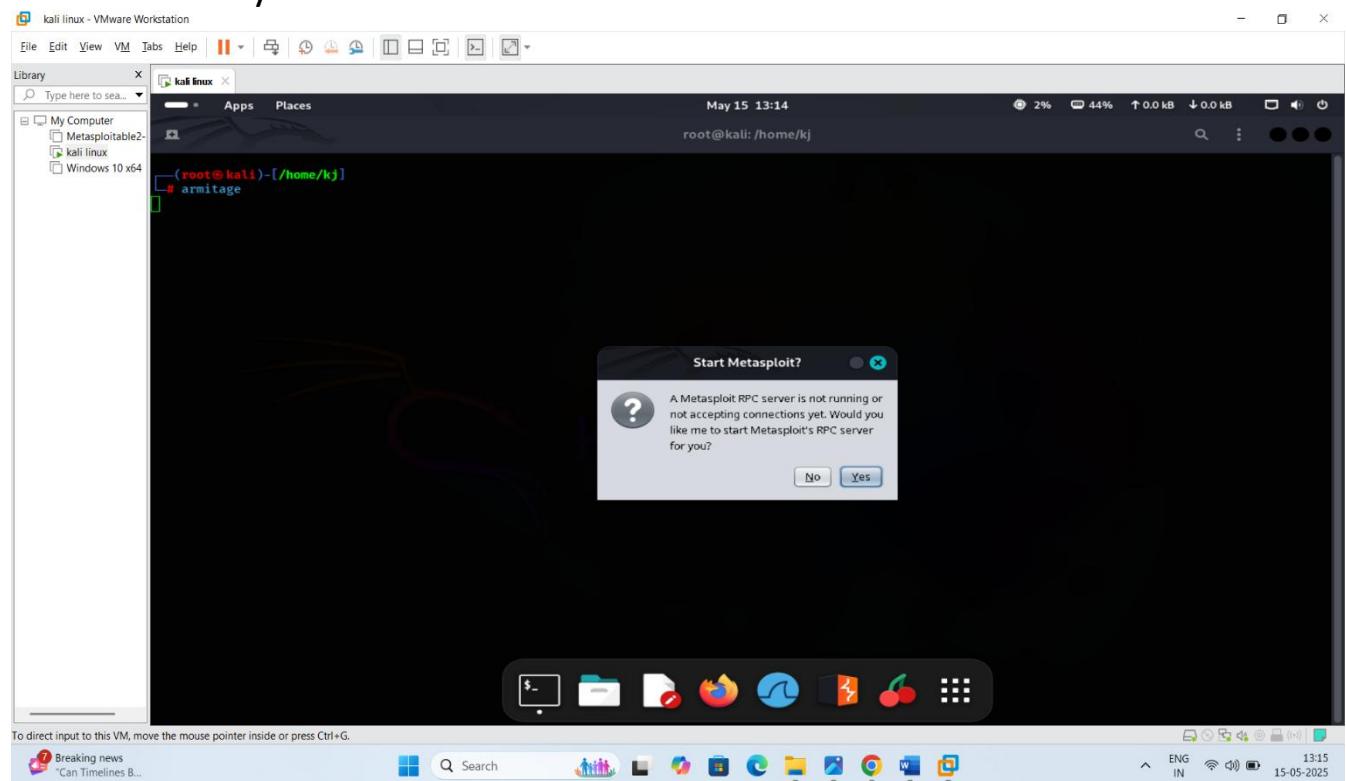


Name : kunal Jawale

### 3. Click on connect

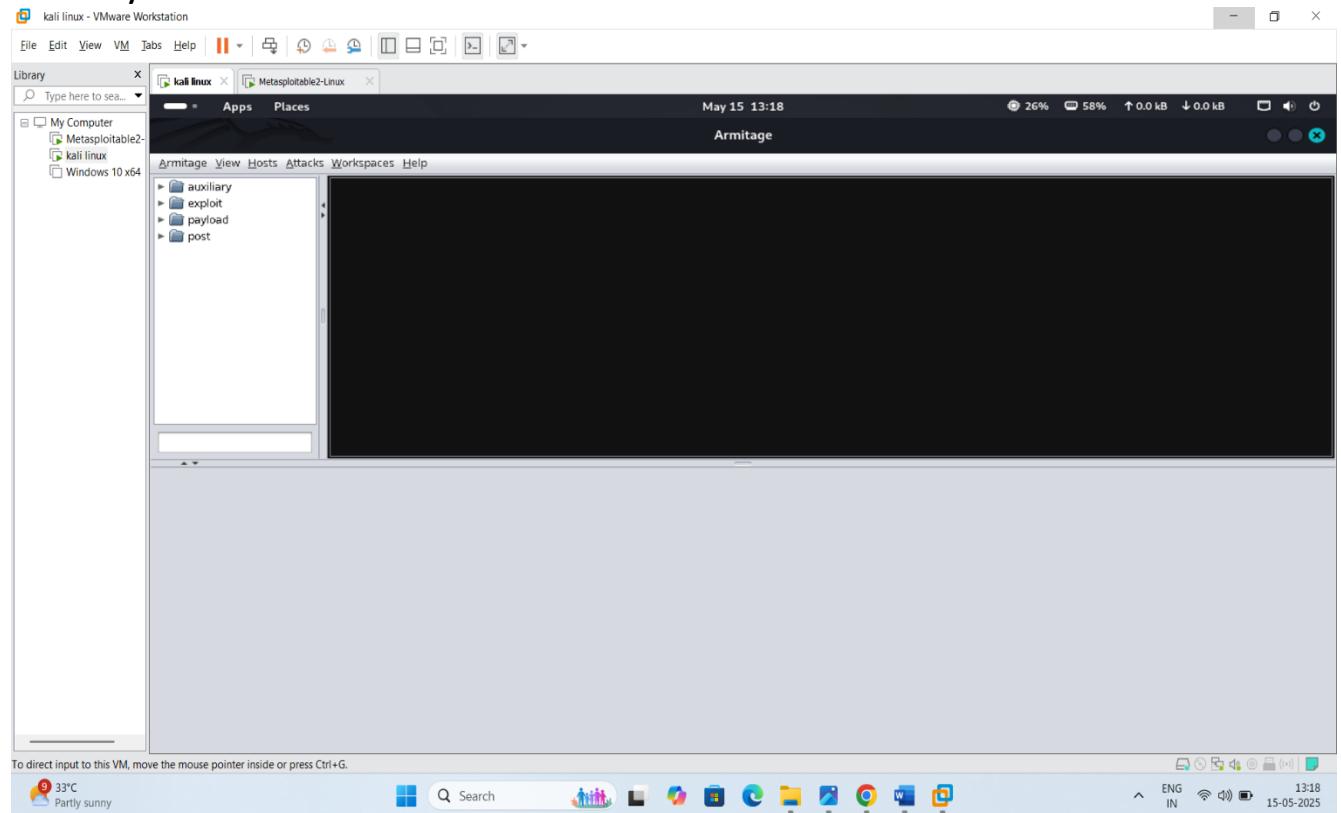


### 4. Click on yes .

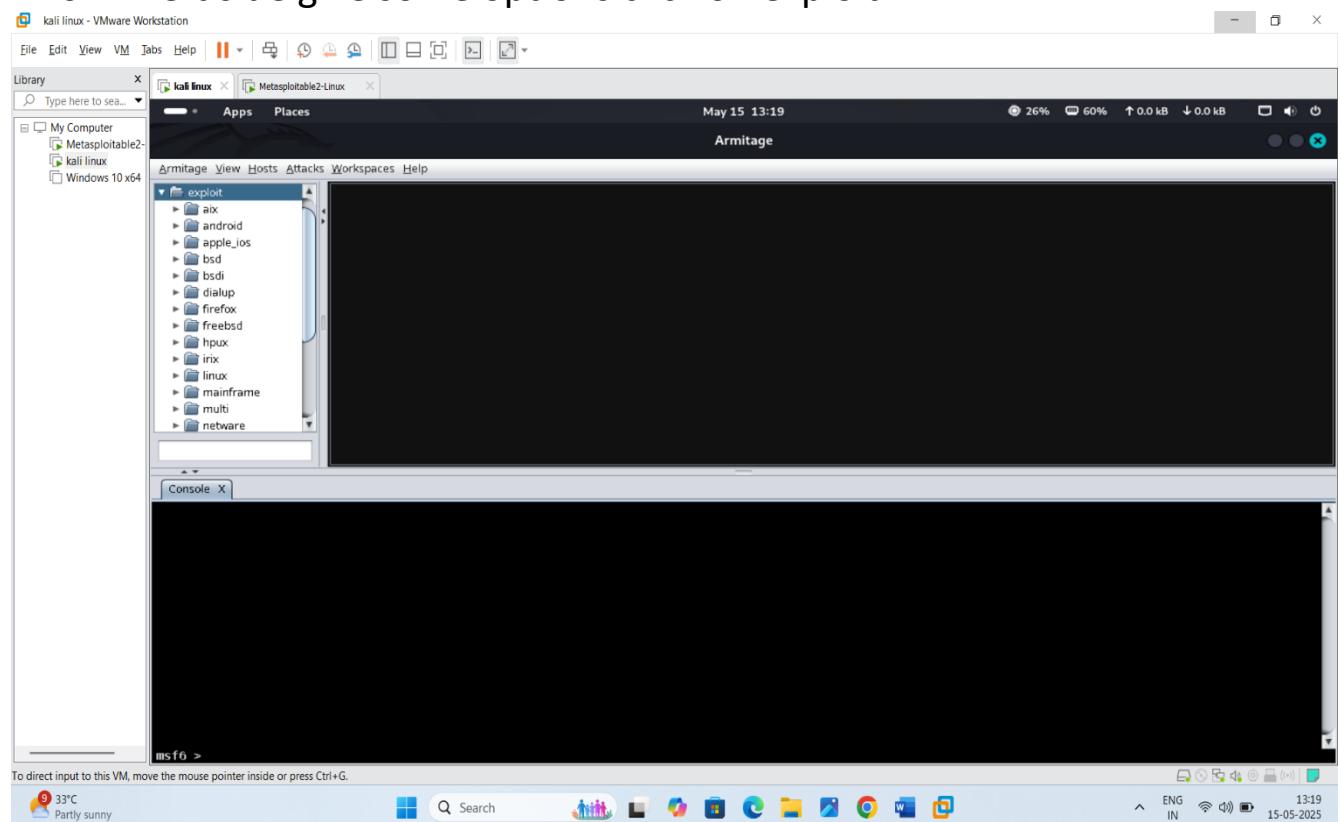


Name : kunal Jawale

## 5. After click on yes you can see the armitage webpage on your system like this screenshot:

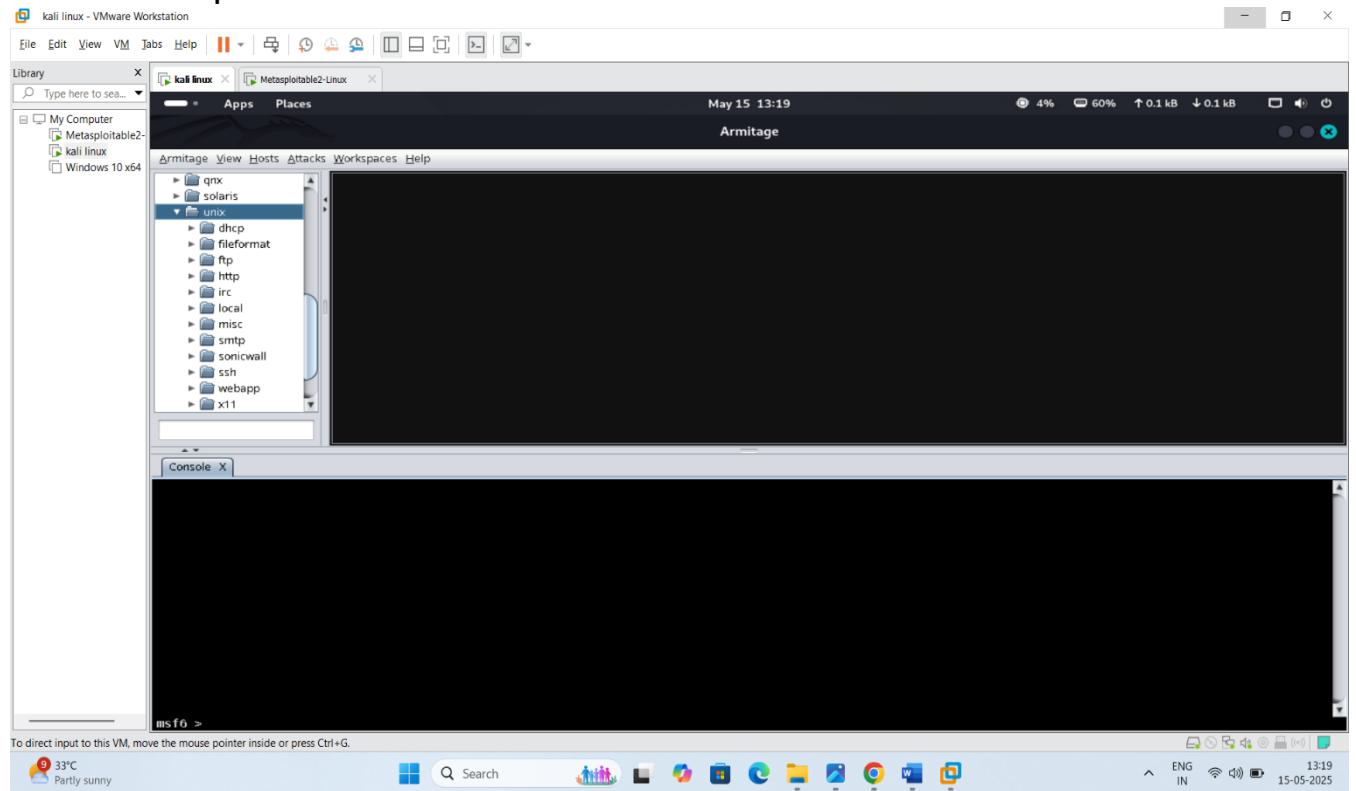


## 6. In left side give some options click on exploit

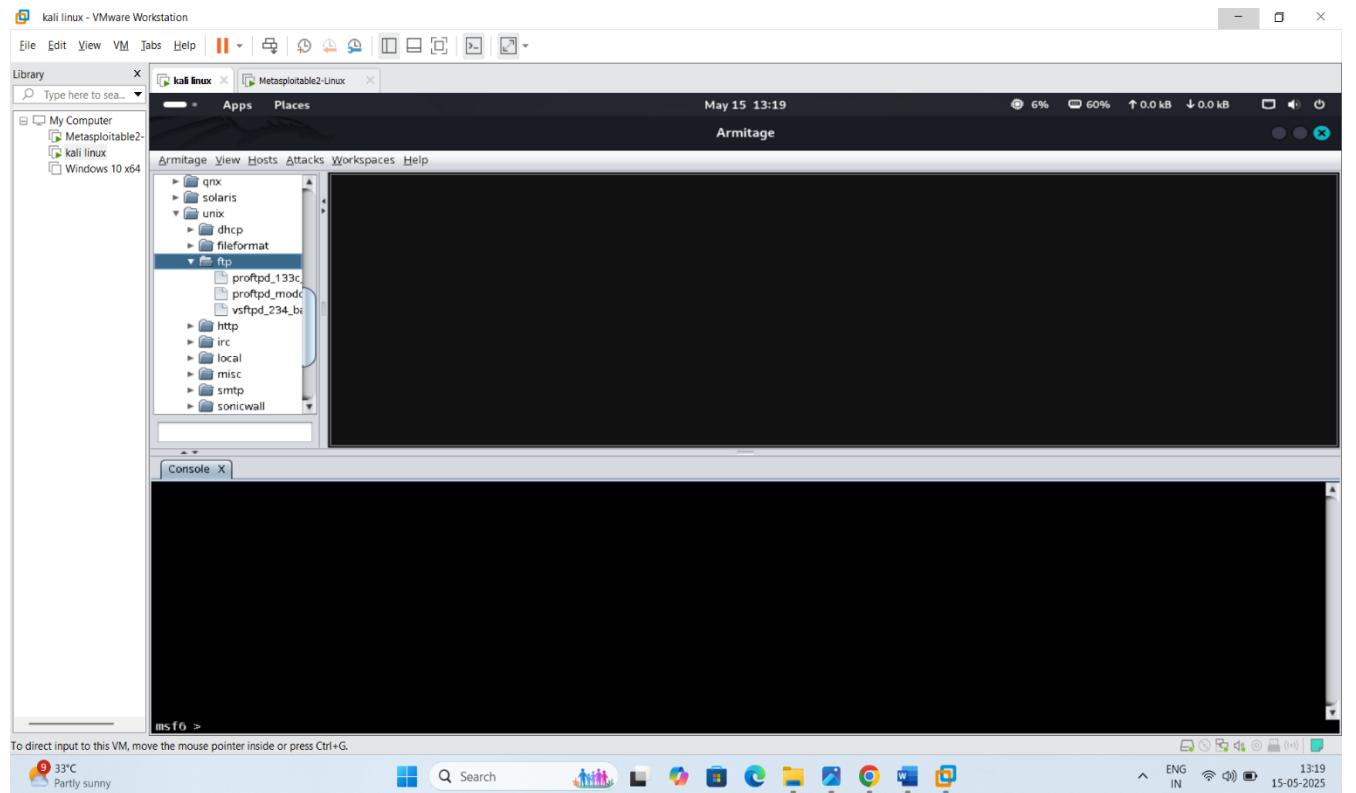


Name : kunal Jawale

## 7. Go down and select unix because we have go for metasploitable



## 8. Scroll down and select ftp



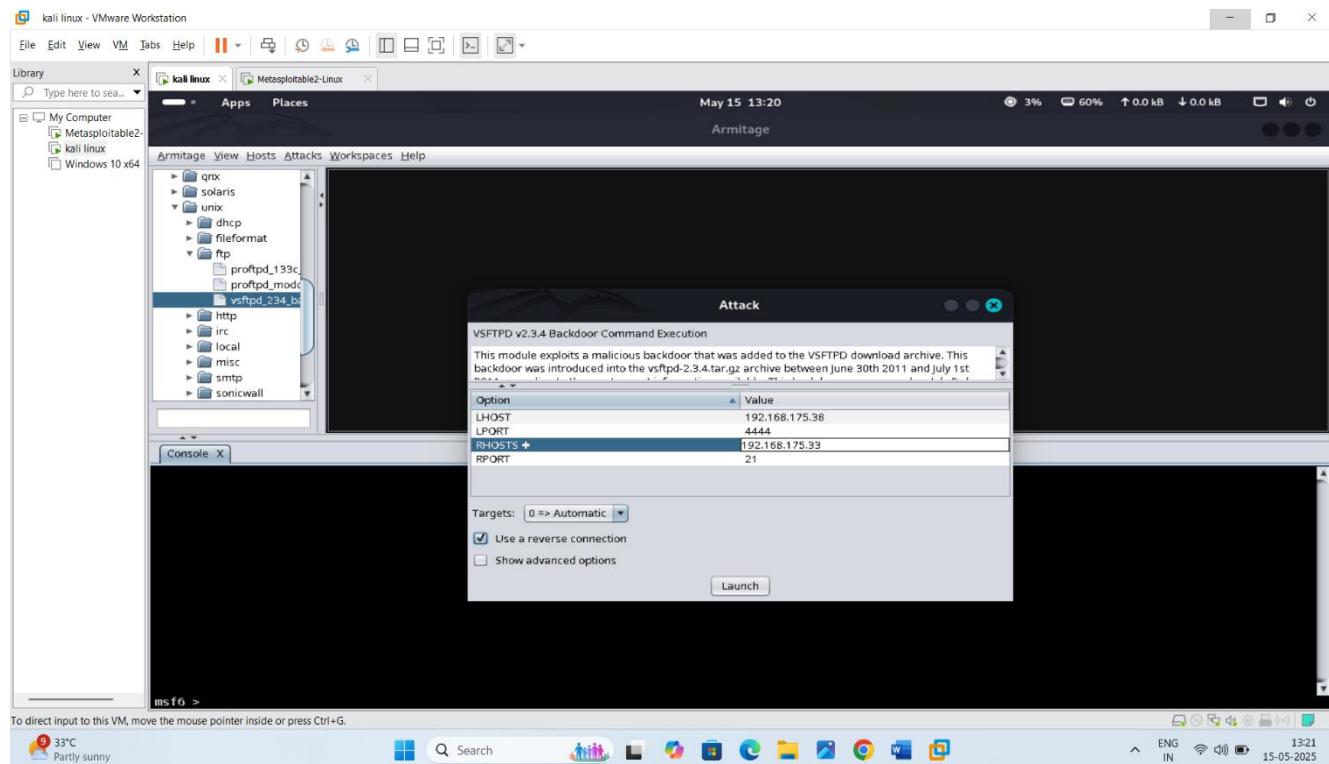
Name : kunal Jawale

9. In ftp select the vulnerable version I select vsftpd and get some changes like LHOST = give your kali IP

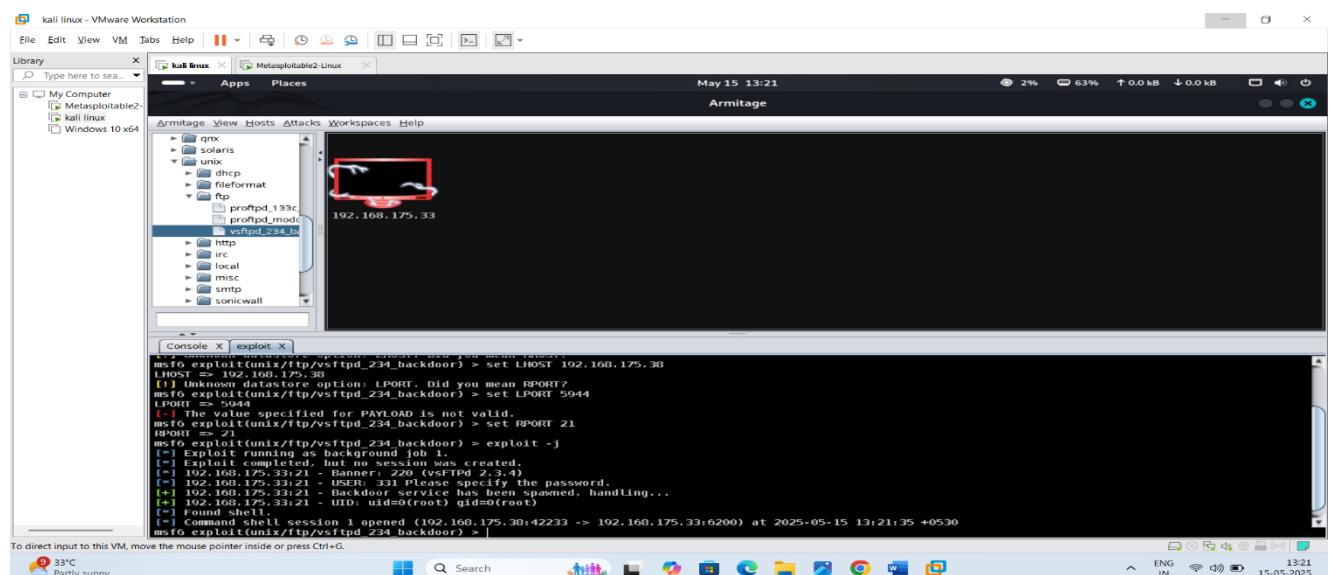
LPORT = 4444

RHOST = give your Metasploitable IP

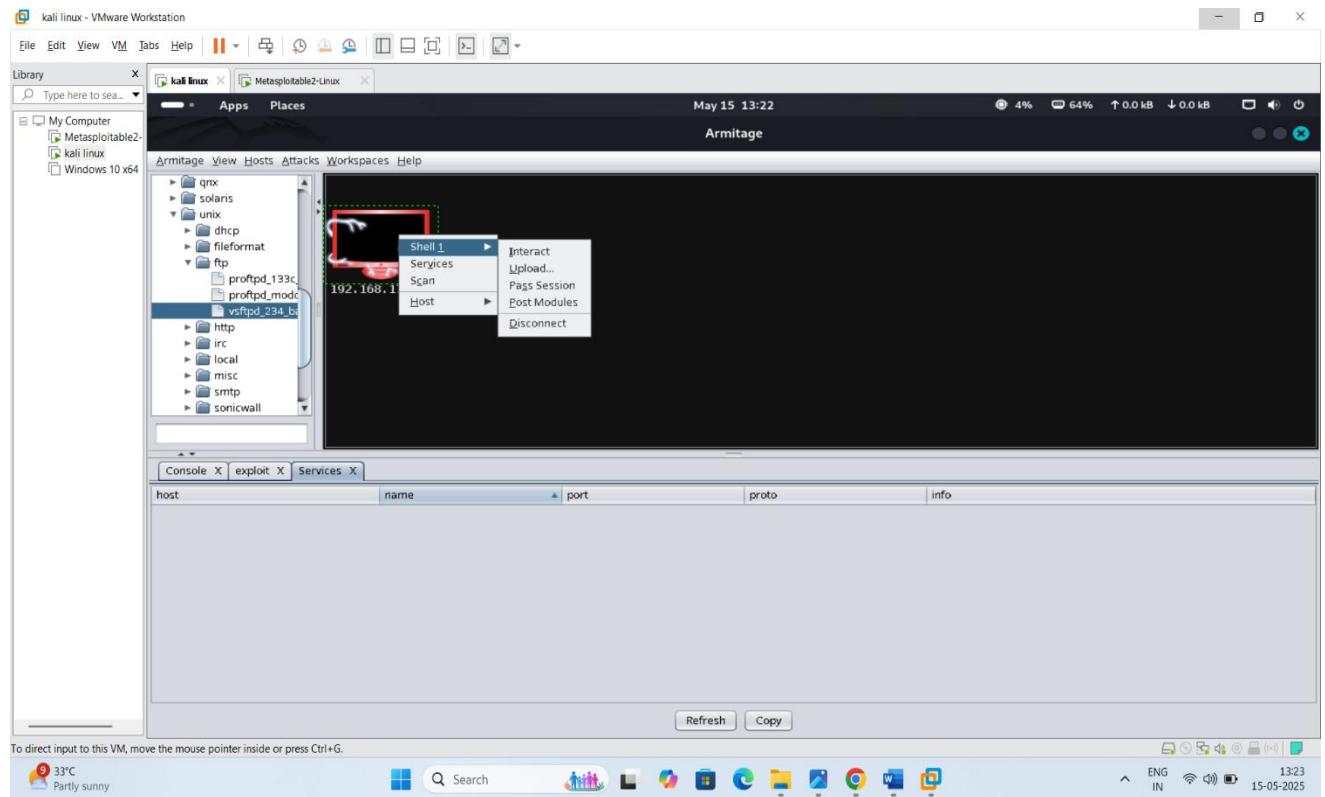
And click on launch for exploit



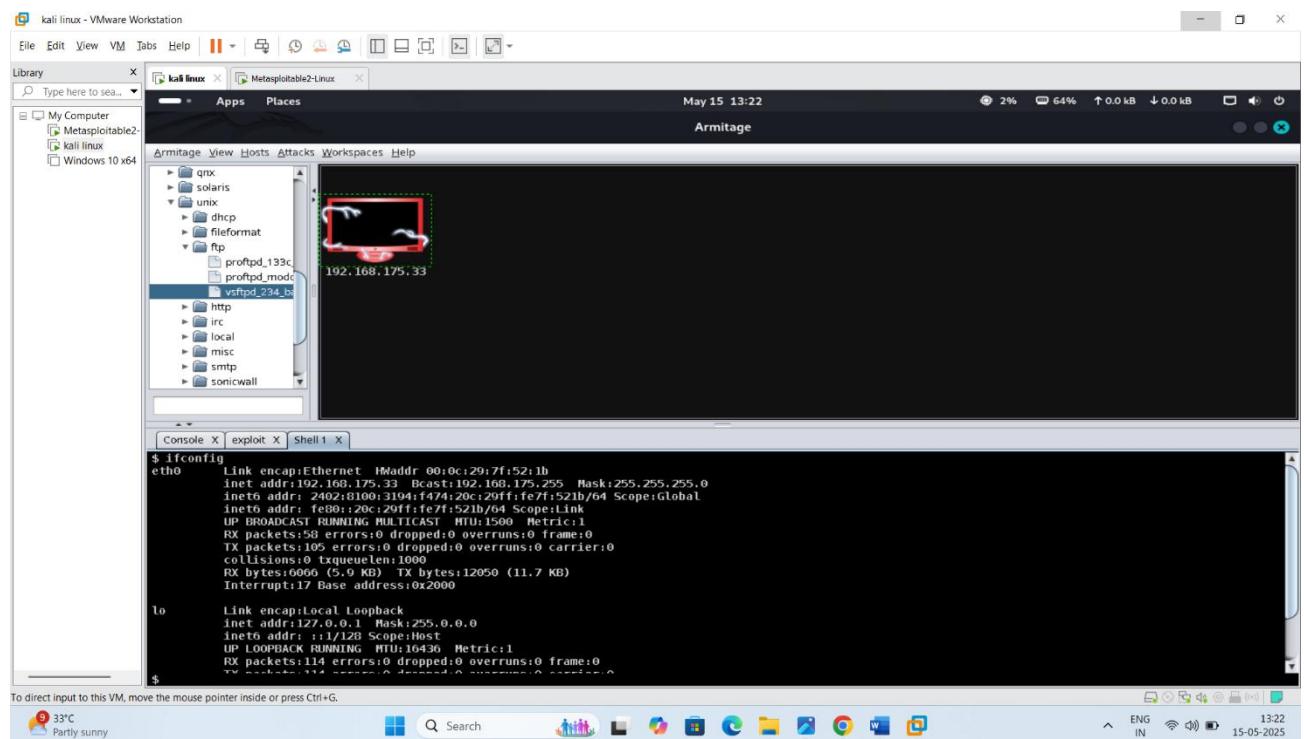
10. After you done this you see the image like following screenshot if you click on this pc image there are options like this :



Name : kunal Jawale



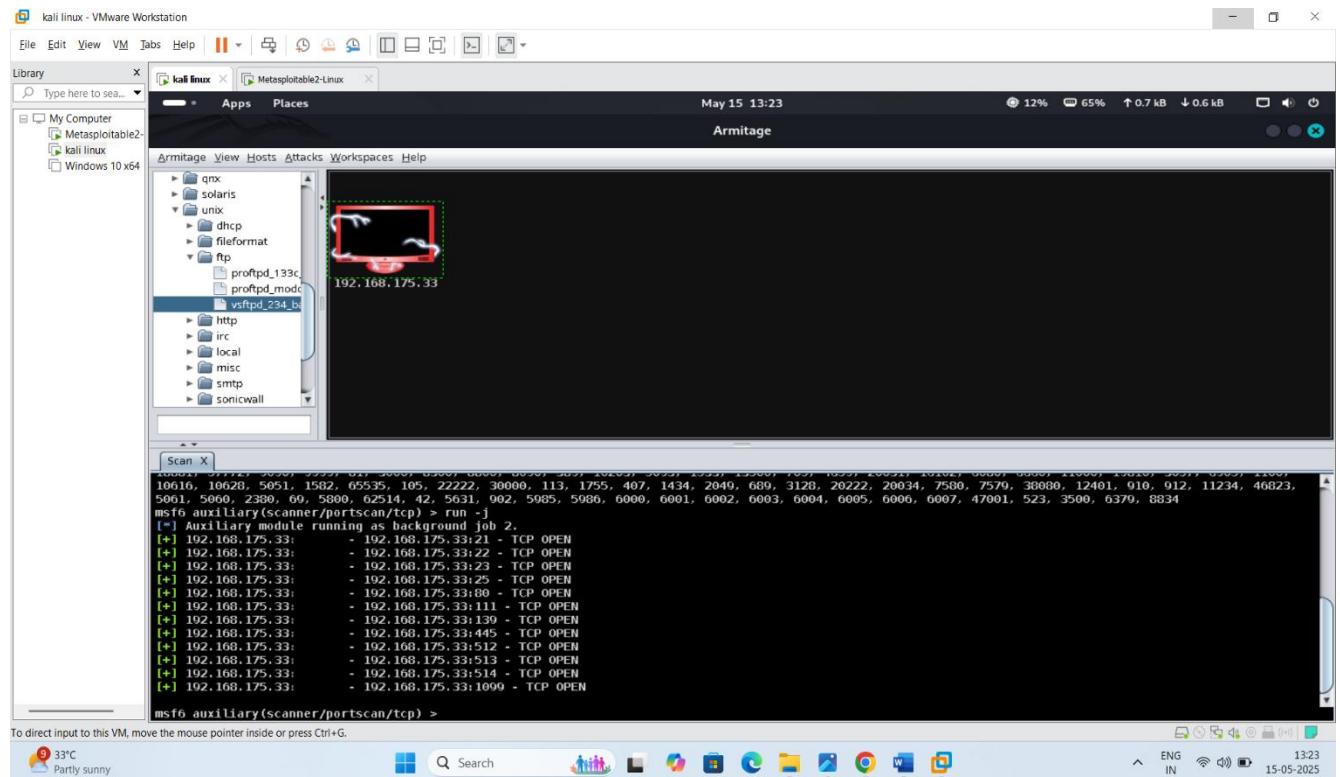
## 11. Click on shell so you can directly go in metasploitable terminal



U can see the IP address of metasploitable

Name : kunal Jawale

12. If you can go for scan the Armitage scan the metasploitable and show the open ports and etc. like this :



Armitage is a graphical version of msfconsole .

# / dirbuster

## What is DirBuster?

**DirBuster** is a web application directory and file brute-forcing tool. It is used to discover hidden directories and files on a web server that are not exposed through standard navigation or linked pages. These hidden resources often include sensitive information or administrative interfaces that can be exploited.

---

## Primary Uses of DirBuster in Hacking Web Servers:

### 1. Directory and File Enumeration:

- DirBuster brute-forces common directory and file names based on wordlists to **uncover hidden content** like:
  - /admin/
  - /backup/
  - /old/
  - /test/
  - .git/, .env, wp-config.php, etc.

## 2. Finding Misconfigured or Forgotten Files:

- Web servers often have:
  - **Backup files** (index.php.bak, config.old)
  - **Temporary files** (temp.php, session.txt)
  - **Configuration files** (config.php, .htpasswd)
- These files might contain **credentials, database information, or source code**.

## 3. Identifying Vulnerable Endpoints:

- By discovering hidden endpoints, you can find:
  - **API endpoints** not intended for public use.
  - **Admin login panels** that are not listed publicly.
  - **Debug interfaces** with verbose error messages.

## 4. Testing for Web Application Firewall (WAF) Evasion:

- DirBuster can attempt different URL encodings and variations to bypass simple WAF protections.

## 5. Recursive Scanning:

- It allows recursive crawling of discovered directories, further expanding its attack surface.

---

## Example of a DirBuster Scan:

To start a typical DirBuster scan:

```
dirbuster -u http://example.com -w  
/usr/share/wordlists/dirbuster/directory-list-2.3-medium.txt -e php,html,txt
```

- -u http://example.com: Target URL.
- -w /path/to/wordlist.txt: Path to the wordlist.
- -e php,html,txt: Limit search to specific extensions (e.g., .php, .html, .txt).

---

## Common Wordlists Used with DirBuster:

- **directory-list-2.3-medium.txt** → Medium-sized list of common directory names.
  - **directory-list-2.3-big.txt** → Large list for deeper searching.
  - **common.txt** → Lightweight, faster scanning with common directories.
  - **Apache and Nginx wordlists** → To find default paths.
- 

### Typical Discoveries from a DirBuster Scan:

Path Found	Description	Potential Exploit
/admin/	Admin panel login	Brute-force or exploit default creds
/backup.zip	Backup of the entire site	Extract sensitive data
/test/	Test scripts and files	Exploit test environments
/phpmyadmin/	Database admin interface	Login brute-force or default passwords
/wp-admin/	WordPress admin area	Brute-force or exploit plugin vulnerabilities
/logs/	Server logs	Sensitive info leakage

---

### Advantages of DirBuster:

1. **Multithreaded Scanning:** Faster enumeration of directories and files.
  2. **Customizable Wordlists:** Supports both pre-built and custom wordlists.
  3. **File Extension Targeting:** Allows you to specify extensions like .php, .html, .txt, etc.
  4. **Recursive Search:** Automatically digs deeper into newly found folders.
- 

### Limitations:

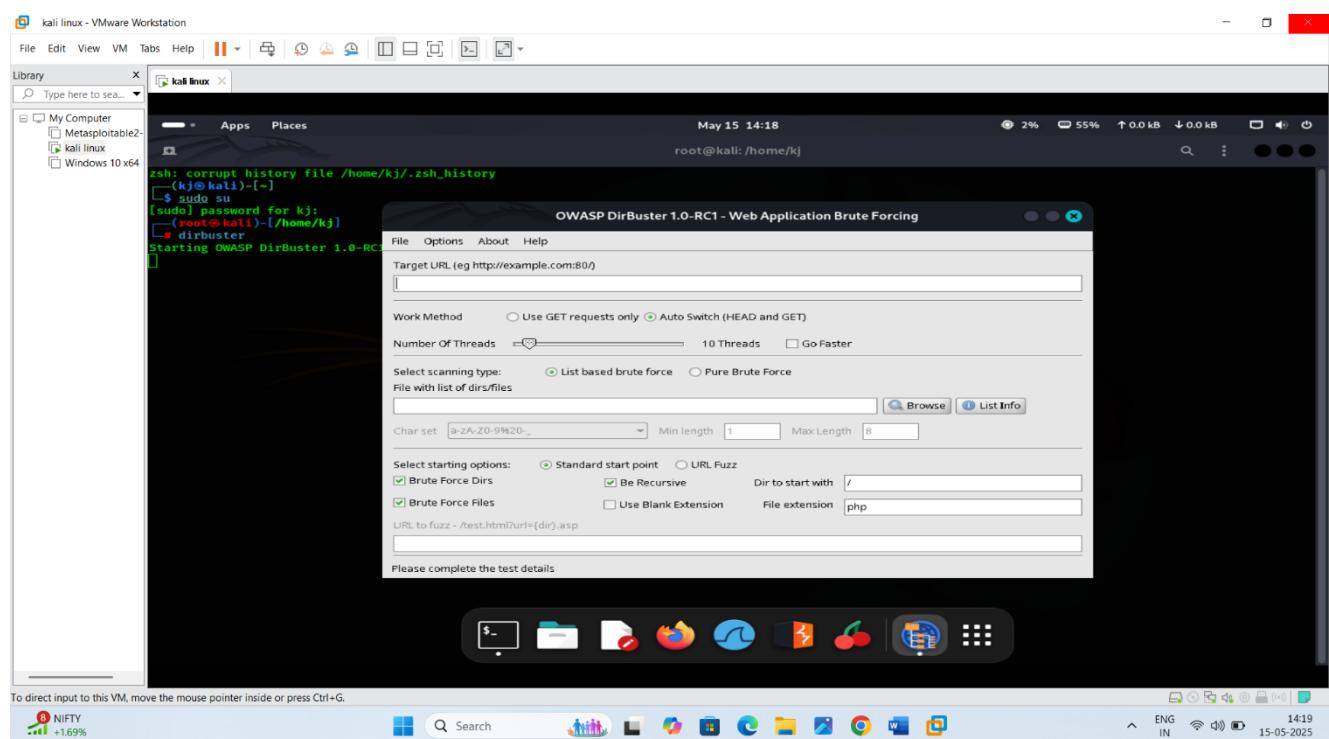
1. **High Noise Level:** Very noisy, easily detected by Intrusion Detection Systems (IDS).
2. **Long Scanning Time:** Large wordlists can take a long time to finish.

3. **Outdated Development:** DirBuster is not actively maintained; **gobuster** and **ffuf** are often used as modern replacements.
  4. **WAF Blocking:** Modern WAFs may block or throttle aggressive scanning.
- 

## ☞ Use dirbuster to find directory using bruteforce

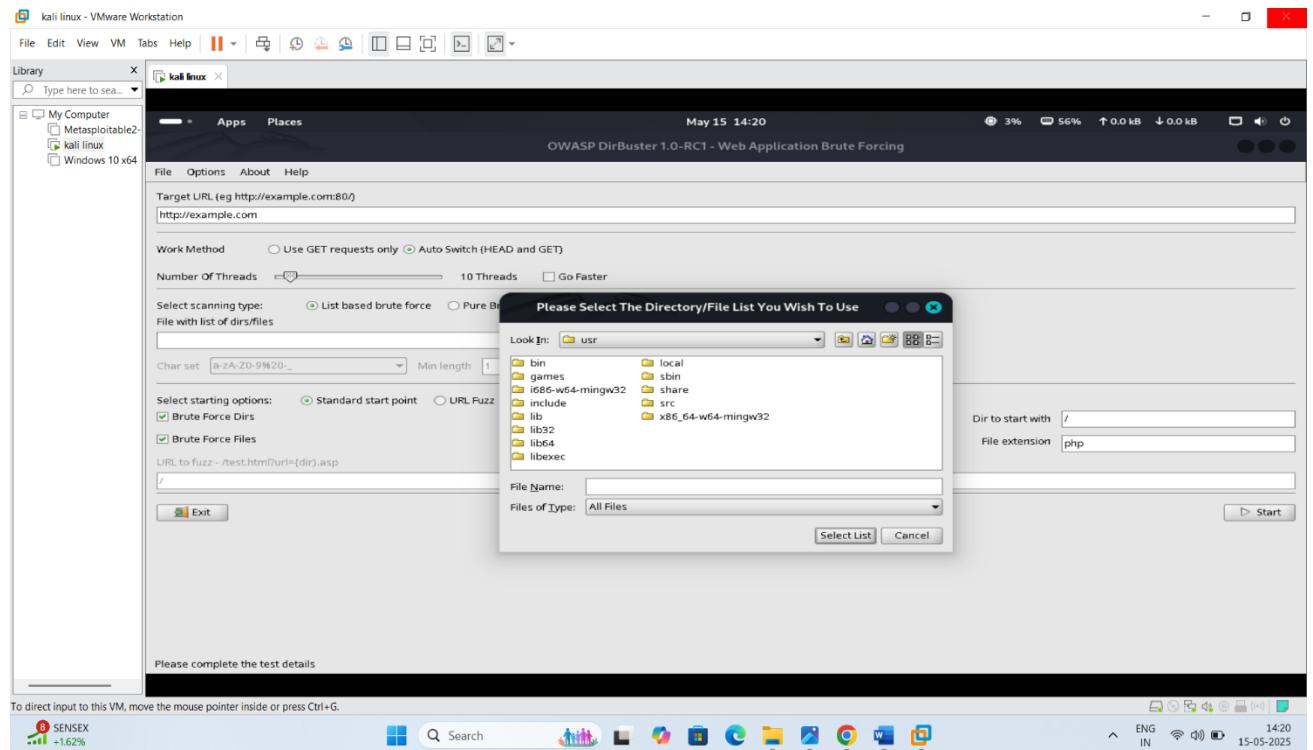
Steps

1. on kali and type dirbuster for run the dirbuster

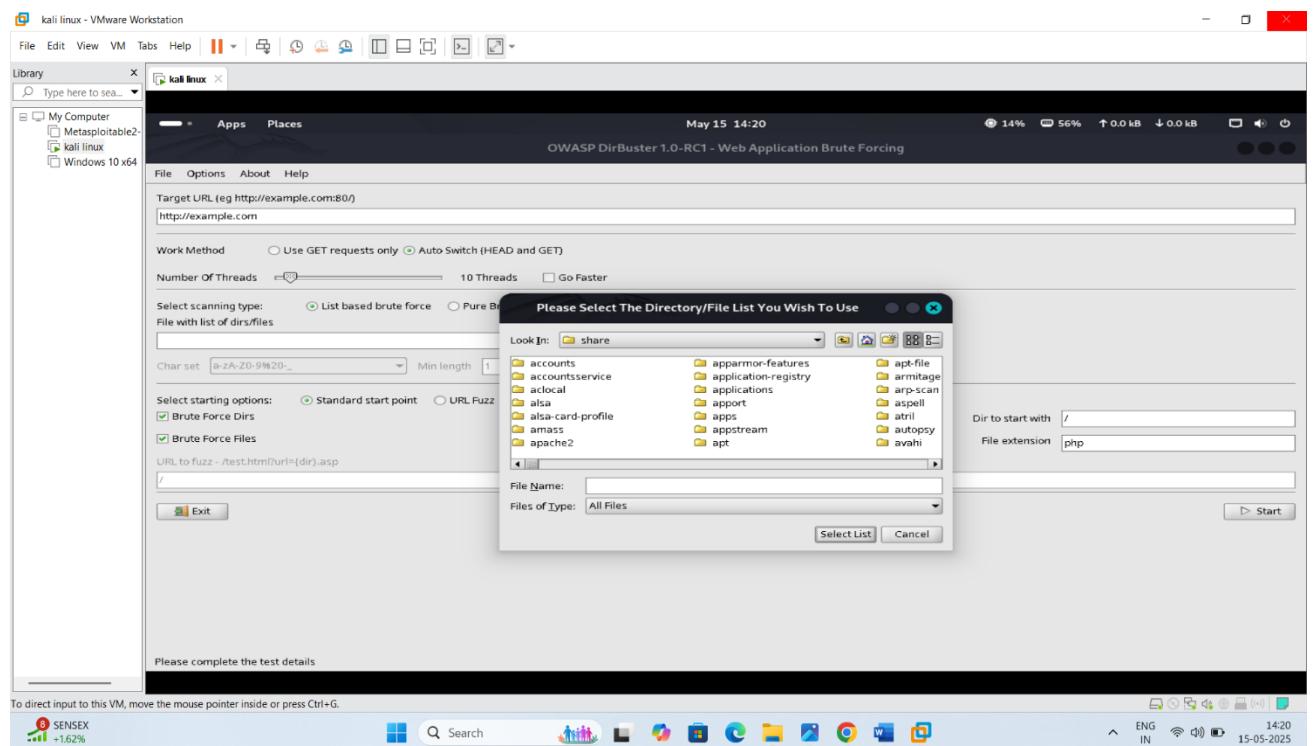


2. complete the target url and after click on browse option and click on usr

Name : kunal Jawale

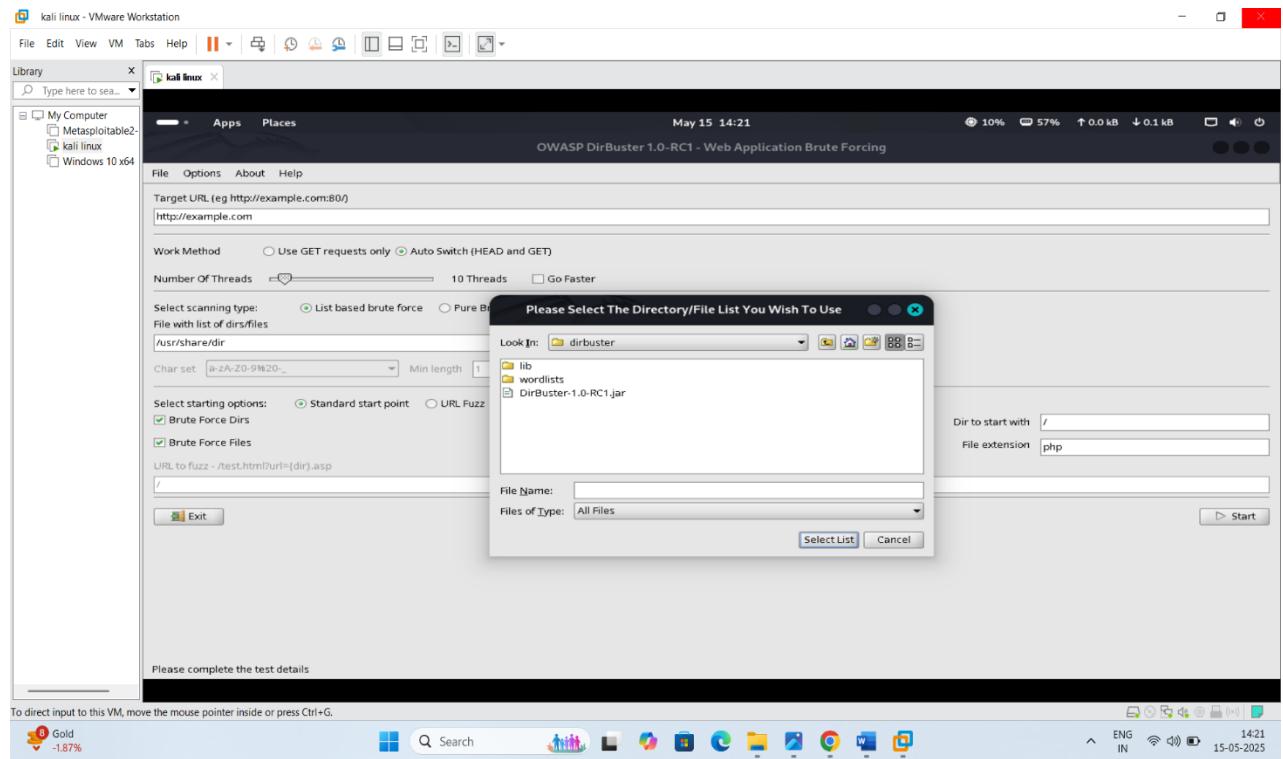


### 3. after this click on share directory

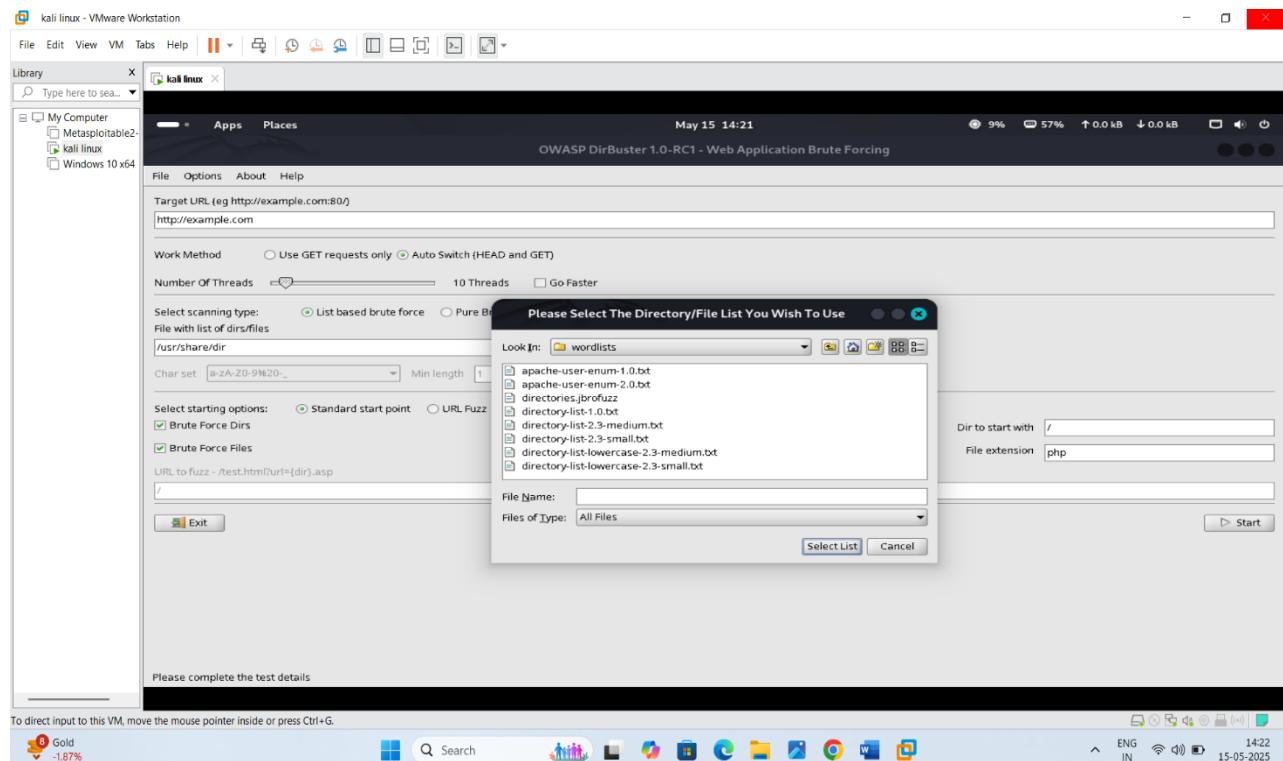


### 4. in share directory search for dirbuster

Name : kunal Jawale

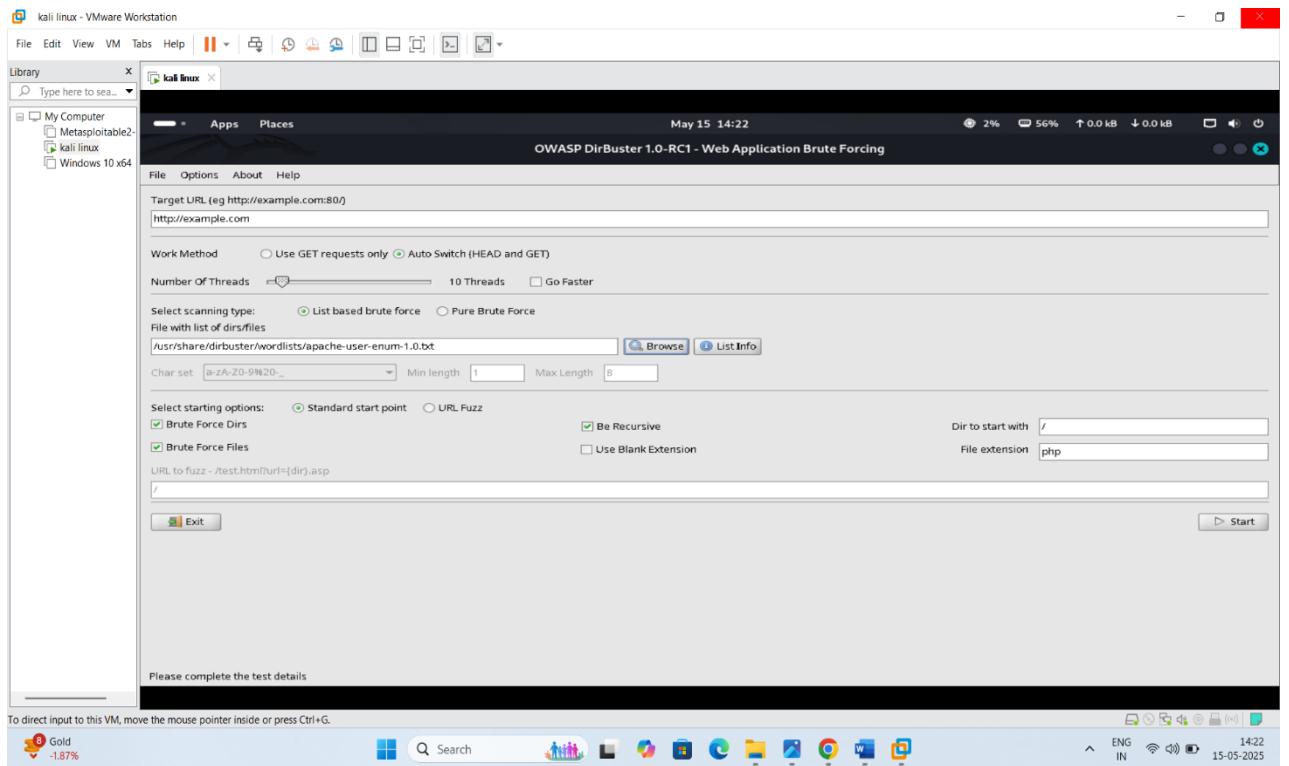


## 5. after coming dirbuster directory click on wordlists.

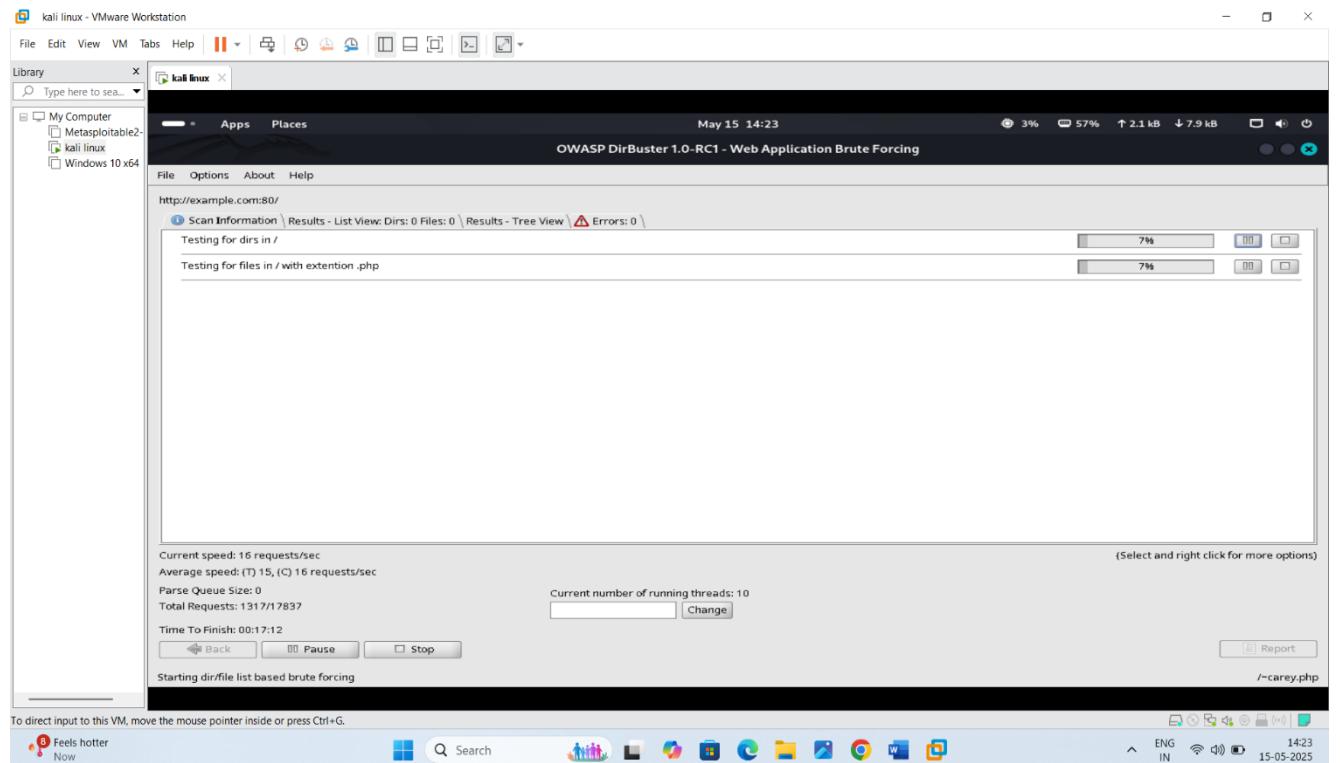


## 6. after come in wordlists select anyone you want

Name : kunal Jawale



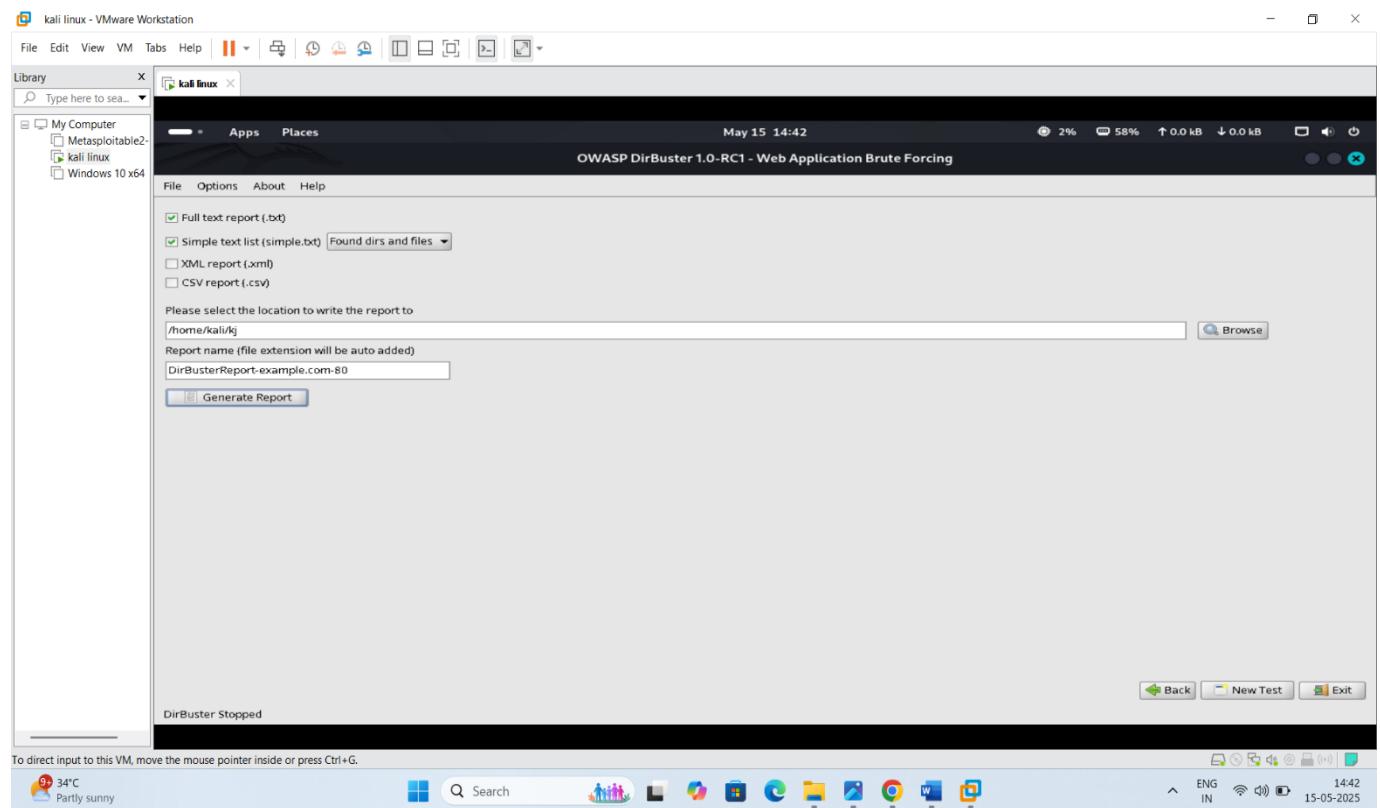
## 7. after all done click on start for brute force



The process is running ...

Name : kunal Jawale

you can also generate the report using dirbuster like following screenshot :



Name : kunal Jawale