Exploiting Network-based vulnerabilities Using enum4linux and Ettercap

Note: These labs are performed based on the lab works provided in "Module 5.1: Exploiting Network-Based Vulnerabilities" in the "Ethical Hacking" Course provided by CISCO.

Network-based vulnerabilities and exploits can be catastrophic due to the types of damage and impact they can cause within an organization.

Lab 1- Scanning for SMB Vulnerabilities with enum4linux

Enum4linux is a tool for enumerating information from Windows and Samba, which is an application that enables Linux and Apple clients to participate in Windows networks.

In this lab, we will do the following actions:

- Launch **enum4linux** and explore its capabilities.
- Identify computers with **SMB** services running.
- Use **enum4linux** to enumerate users and network file shares.
- Use **smbclient** to transfer files between systems.

The following resources will be required to perform this lab.

- Kali VM customized for Ethical Hacker course
- Internet access

Part 1: Launch enum4linux and explore its capabilities.

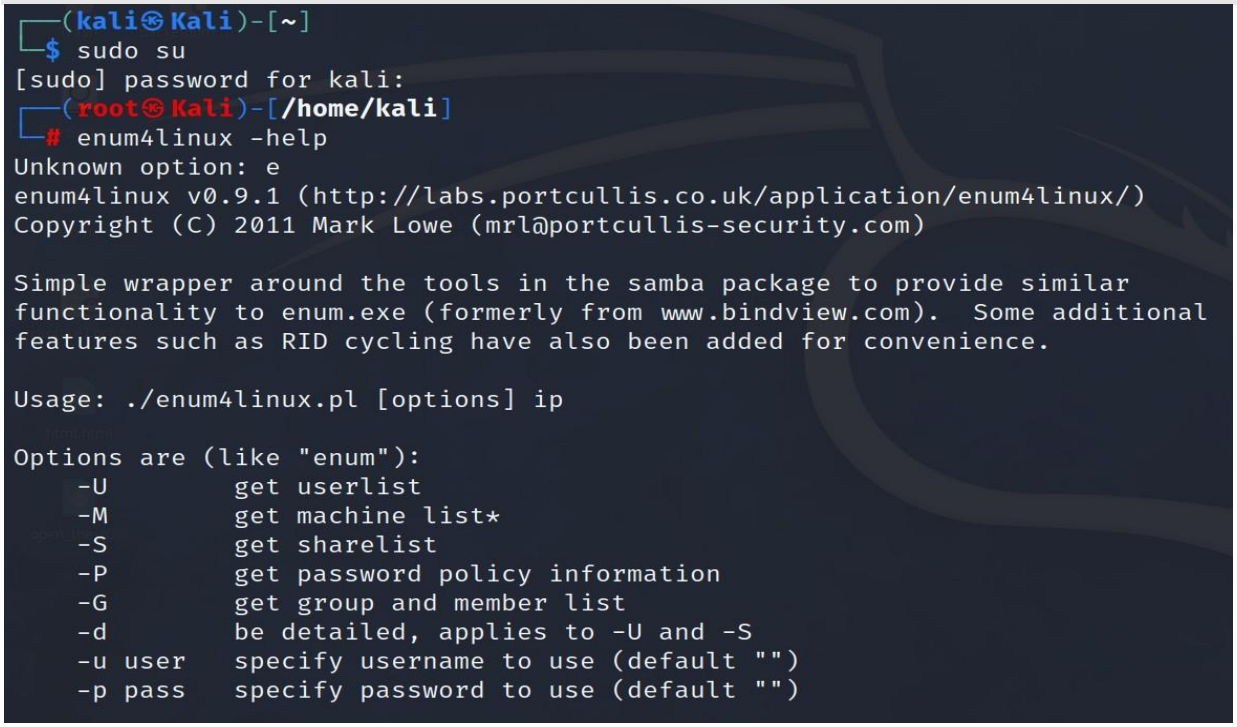
a. At first, load Kali Linux using the username kali and the password kali, then open a terminal session from the menu bar at the top of the screen.

b. The **enum4linux** commands may require running as root, so we should use the **sudo su** command as given below to obtain persistent root access.

```
$ sudo su
```

c. Next, enter the – help command to view the enum4linux help file.

```
# enum4linux –help
```



Part 2: Use Nmap to Find SMB Servers.

a. Now we will scan the virtual networks to find potential targets. We can identify potential targets for SMB enumeration by examining the open ports.

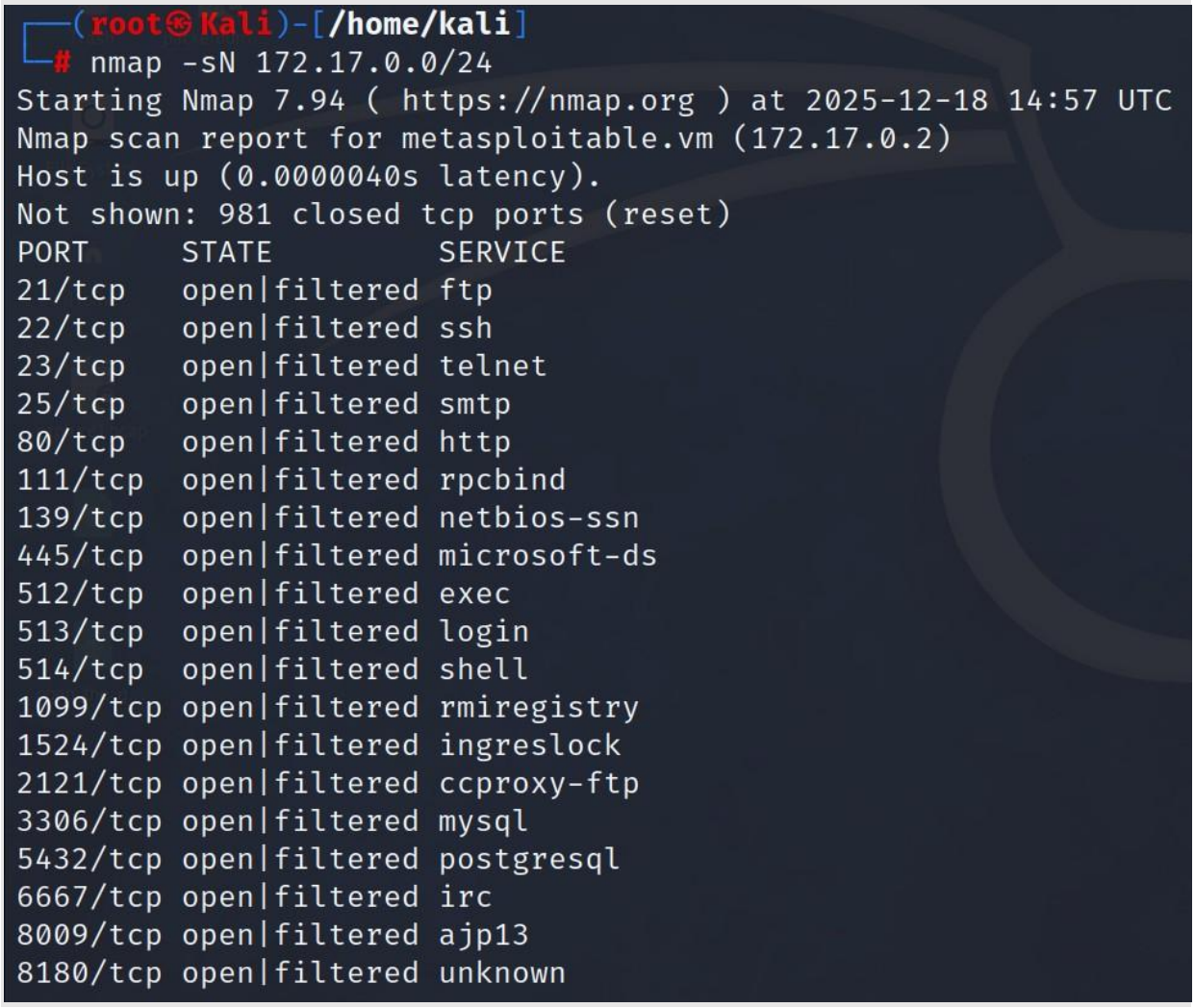
b. Here are the common open ports on SMB servers:

TCP 135	RPC
TCP 139	NetBIOS Session
TCP 389	LDAP Server

TCP 445	SMB File Service
TCP 9389	Active Directory Web Services
TCP/UDP 137	NetBIOS Name Service
UDP 138	NetBIOS Datagram

c. We will use the **nmap -sN** command to find the services available on hosts in the 172.17.0.0 virtual network.

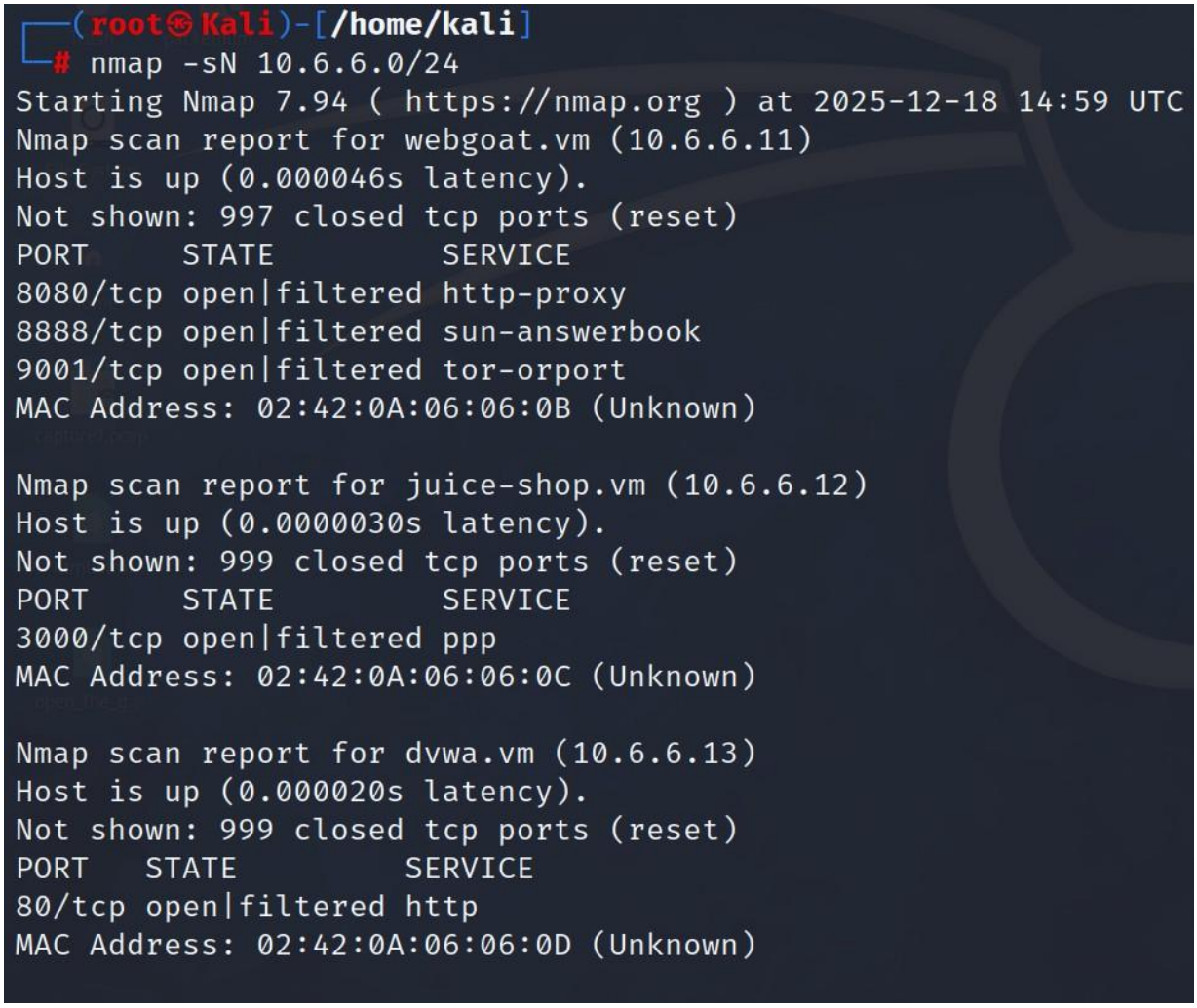
```
# nmap -sN 172.17.0.0/24
```



```
┌──(root㉿Kali)-[/home/kali]
└─# nmap -sN 172.17.0.0/24
Starting Nmap 7.94 ( https://nmap.org ) at 2025-12-18 14:57 UTC
Nmap scan report for metasploitable.vm (172.17.0.2)
Host is up (0.0000040s latency).
Not shown: 981 closed tcp ports (reset)
PORT     STATE         SERVICE
21/tcp   open|filtered ftp
22/tcp   open|filtered ssh
23/tcp   open|filtered telnet
25/tcp   open|filtered smtp
80/tcp   open|filtered http
111/tcp  open|filtered rpcbind
139/tcp  open|filtered netbios-ssn
445/tcp  open|filtered microsoft-ds
512/tcp  open|filtered exec
513/tcp  open|filtered login
514/tcp  open|filtered shell
1099/tcp open|filtered rmiregistry
1524/tcp open|filtered ingreslock
2121/tcp open|filtered ccproxy-ftp
3306/tcp open|filtered mysql
5432/tcp open|filtered postgresql
6667/tcp open|filtered irc
8009/tcp open|filtered ajp13
8180/tcp open|filtered unknown
```

d. Now, perform a **nmap -sN** scan on the 10.6.6.0/24 subnet.

```
# nmap -sN 10.6.6.0/24
```



Part 3: Use enum4linux to enumerate users and network file shares

a. Now we will perform an **enum4linux** scan on target **172.17.0.2**. Here are the most common options that can be used on enum4linux scanning:

-U	find configured users
-S	get a list of file shares
-G	get a list of the groups and their members
-P	list the password policies
-i	get a list of printers

b. Use the **enum4linux -U** option to list the users configured on the target 172.17.0.2.

```
# enum4linux -U 172.17.0.2
```

```
┌──(root㉿Kali)-[/home/kali]
└─# enum4linux -U 172.17.0.2
Starting enum4linux v0.9.1 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Thu Dec 18 15:03:18 2025

 ==========( Target Information )==========

Target ........... 172.17.0.2
RID Range ........ 500-550,1000-1050
Username ......... ''
Password ......... ''
Known Usernames .. administrator, guest, krbtgt, domain admins, root, bin, none


 ==========( Enumerating Workgroup/Domain on 172.17.0.2 )==========


[+] Got domain/workgroup name: WORKGROUP


 ==========( Session Check on 172.17.0.2 )==========


[+] Server 172.17.0.2 allows sessions using username '', password ''
```

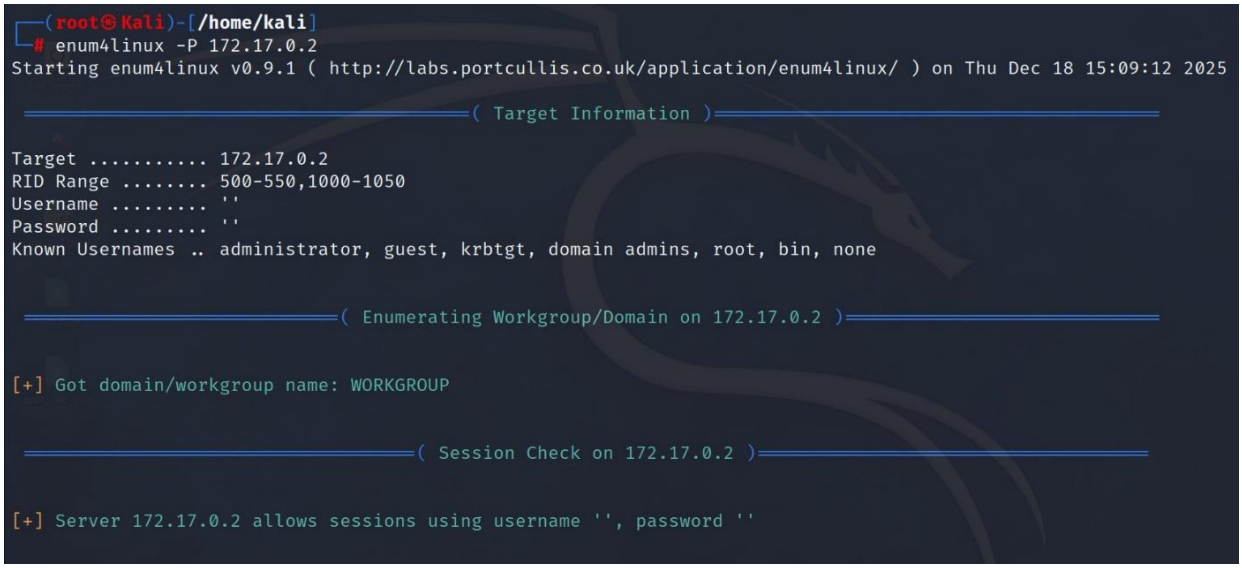
c. We can list the file shares available on **172.17.0.2** using the **enum4linux -S** command.

```
# enum4linux -Sv 172.17.0.2
```

```
┌──(root㉿Kali)-[/home/kali]
└─# enum4linux -Sv 172.17.0.2

[V] Dependent program "nmblookup" found in /usr/bin/nmblookup


[V] Dependent program "net" found in /usr/bin/net


[V] Dependent program "rpcclient" found in /usr/bin/rpcclient


[V] Dependent program "smbclient" found in /usr/bin/smbclient


[V] Dependent program "polenum" found in /usr/bin/polenum


[V] Dependent program "ldapsearch" found in /usr/bin/ldapsearch

Starting enum4linux v0.9.1 ( http://labs.portcullis.co.uk/application/enum4linux/ ) on Thu Dec 18 15:06:07 2025

 ==========( Target Information )==========
```

d. We can use the **enum4linux -P** command to list the password policies.

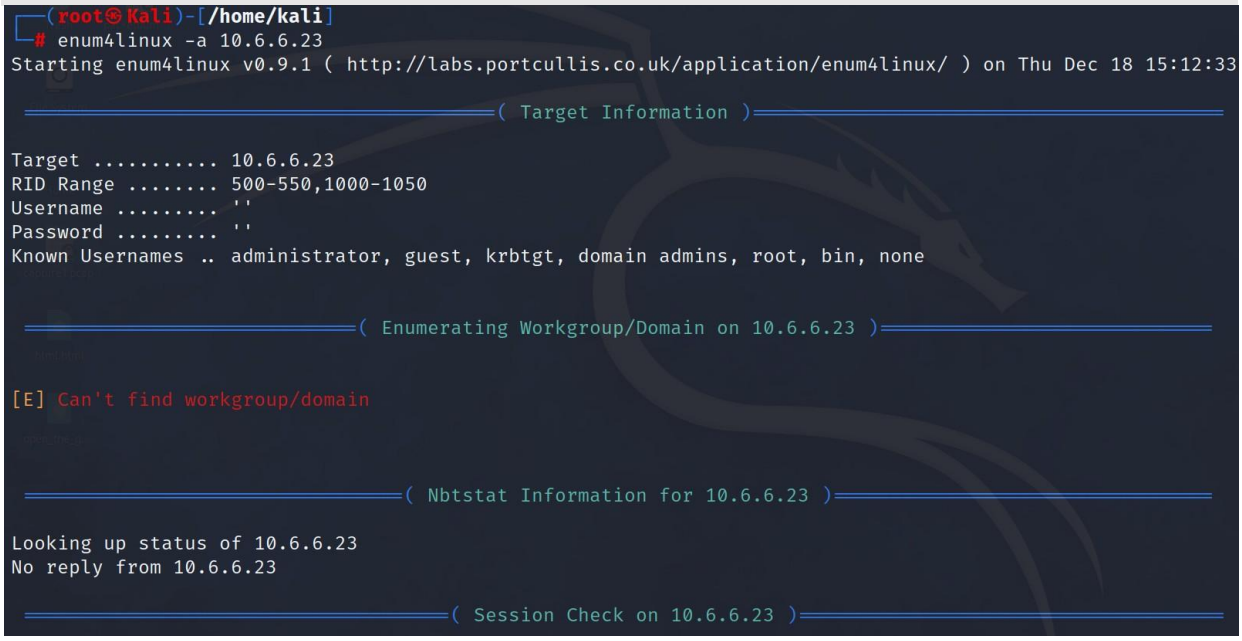
```
# enum4linux -P 172.17.0.2
```



e. Now, we will perform a simple enumeration scan on target 10.6.6.23. For this, use the -a argument, which combines the -U, -S, -G, -P, -r, -o, -n, -i options into one command.

f. Use the **enum4linux -a** command to perform a scan on the potential Samba server.

```
# enum4linux -a 10.6.6.23
```

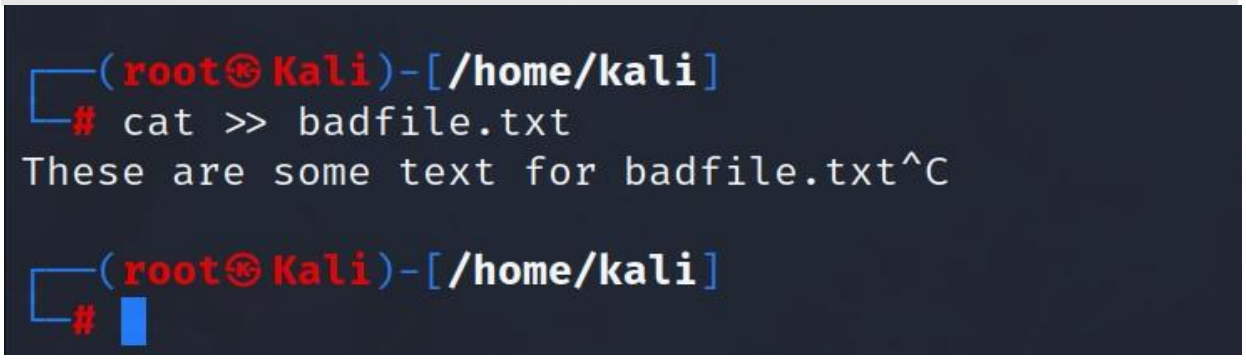


Part 4: Use smbclient to transfer files between systems

Smbclient is a component of Samba that can store and retrieve files, similar to an FTP client. Here we will use smbclient to transfer a file to the target system at **172.17.0.2**.

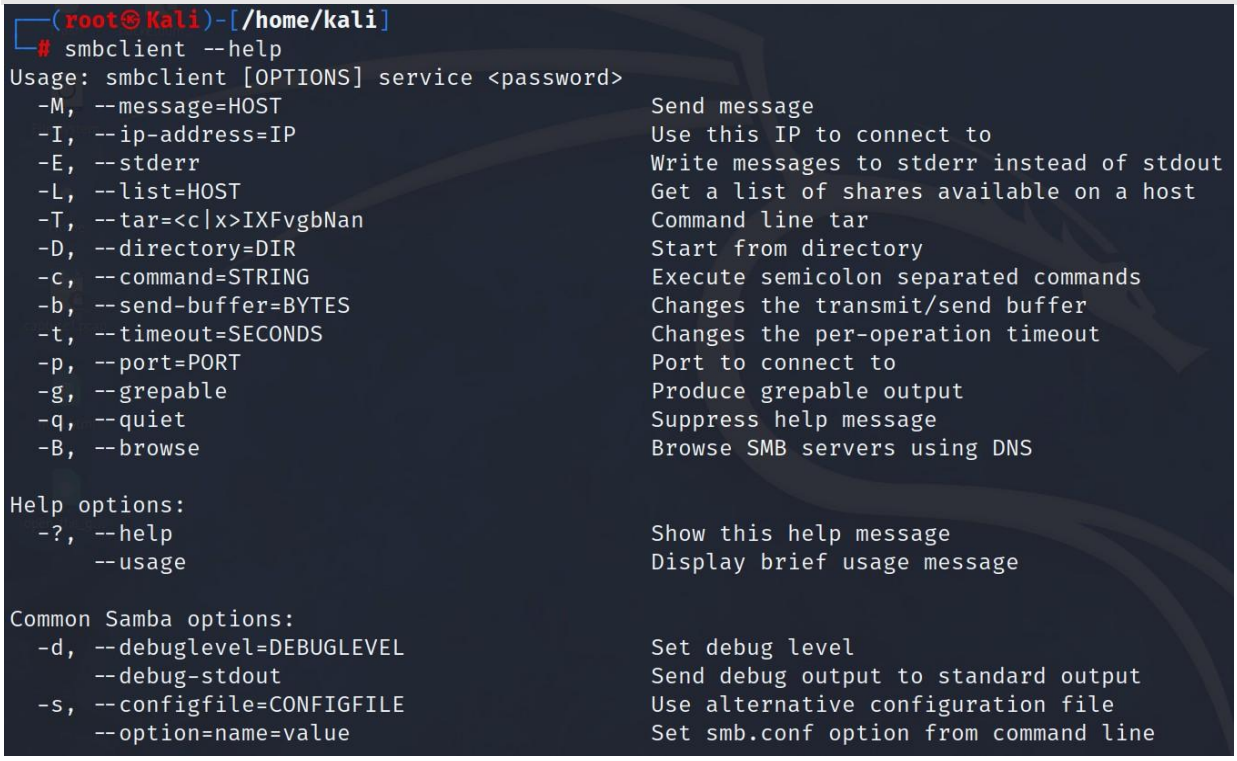
a. At first, we will create a text file with some text inside it using the cat command. In this lab, we use the name of the file **badfile.txt**. Press CTRL-C when finished.

```
# cat >> badfile.txt
```



b. Next, take a look at the options available with **smbclient** using the command **smbclient –help**.

```
# smbclient –help
```



c. Use the **smbclient -L** command to list the shares on the target host. (Press Enter on Password Request)

```
# smbclient -L //172.17.0.2/
```

```
┌──(root㉿Kali)-[/home/kali]
└─# smbclient -L //172.17.0.2/
Password for [WORKGROUP\root]:
Anonymous login successful

        Sharename       Type      Comment
        ---------       ----      -------
        print$          Disk      Printer Drivers
        tmp             Disk      oh noes!
        opt             Disk
        IPC$            IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
        ADMIN$          IPC       IPC Service (metasploitable server (Samba 3.0.20-Debian))
Reconnecting with SMB1 for workgroup listing.
Anonymous login successful

        Server               Comment
        ---------            -------

        Workgroup            Master
        ---------            -------
        WORKGROUP            METASPLOITABLE
```

d. We will connect now to the tmp share using the smbclient command by specifying the share name and IP address.

```
# smbclient //172.17.0.2/tmp
```

e. The **Dir** command can be used to view the contents of the share.

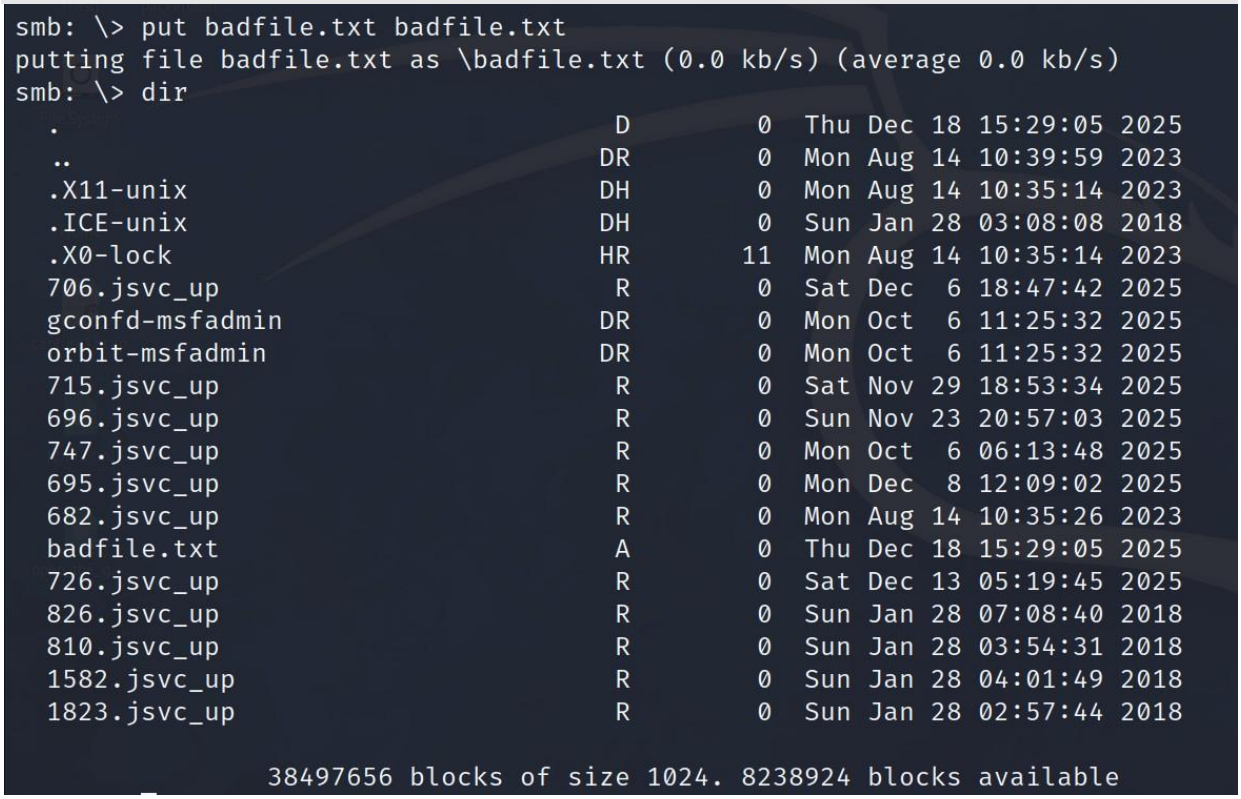
```
┌──(root㉿Kali)-[/home/kali]
└─# smbclient //172.17.0.2/tmp
Password for [WORKGROUP\root]:
Anonymous login successful
Try "help" to get a list of possible commands.
smb: \> dir
  .                                   D        0  Thu Dec 18 15:26:09 2025
  ..                                 DR        0  Mon Aug 14 10:39:59 2023
  .X11-unix                          DH        0  Mon Aug 14 10:35:14 2023
  .ICE-unix                          DH        0  Sun Jan 28 03:08:08 2018
  .X0-lock                           HR       11  Mon Aug 14 10:35:14 2023
  706.jsvc_up                         R        0  Sat Dec  6 18:47:42 2025
  gconfd-msfadmin                    DR        0  Mon Oct  6 11:25:32 2025
  orbit-msfadmin                     DR        0  Mon Oct  6 11:25:32 2025
  715.jsvc_up                         R        0  Sat Nov 29 18:53:34 2025
  696.jsvc_up                         R        0  Sun Nov 23 20:57:03 2025
  747.jsvc_up                         R        0  Mon Oct  6 06:13:48 2025
  695.jsvc_up                         R        0  Mon Dec  8 12:09:02 2025
  682.jsvc_up                         R        0  Mon Aug 14 10:35:26 2023
  726.jsvc_up                         R        0  Sat Dec 13 05:19:45 2025
  826.jsvc_up                         R        0  Sun Jan 28 07:08:40 2018
  810.jsvc_up                         R        0  Sun Jan 28 03:54:31 2018
  1582.jsvc_up                        R        0  Sun Jan 28 04:01:49 2018
  1823.jsvc_up                        R        0  Sun Jan 28 02:57:44 2018
```

f. Upload the **badfile.txt** to the target server using the put command.

```
smb: > put badfile.txt badfile.txt
```

g. Now we have to verify that the file was successfully uploaded using the **dir** command.

```
smb: > dir
```



```
smb: \> put badfile.txt badfile.txt
putting file badfile.txt as \badfile.txt (0.0 kb/s) (average 0.0 kb/s)
smb: \> dir
  .                                   D        0  Thu Dec 18 15:29:05 2025
  ..                                 DR        0  Mon Aug 14 10:39:59 2023
  .X11-unix                          DH        0  Mon Aug 14 10:35:14 2023
  .ICE-unix                          DH        0  Sun Jan 28 03:08:08 2018
  .X0-lock                           HR       11  Mon Aug 14 10:35:14 2023
  706.jsvc_up                         R        0  Sat Dec  6 18:47:42 2025
  gconfd-msfadmin                    DR        0  Mon Oct  6 11:25:32 2025
  orbit-msfadmin                     DR        0  Mon Oct  6 11:25:32 2025
  715.jsvc_up                         R        0  Sat Nov 29 18:53:34 2025
  696.jsvc_up                         R        0  Sun Nov 23 20:57:03 2025
  747.jsvc_up                         R        0  Mon Oct  6 06:13:48 2025
  695.jsvc_up                         R        0  Mon Dec  8 12:09:02 2025
  682.jsvc_up                         R        0  Mon Aug 14 10:35:26 2023
  badfile.txt                         A        0  Thu Dec 18 15:29:05 2025
  726.jsvc_up                         R        0  Sat Dec 13 05:19:45 2025
  826.jsvc_up                         R        0  Sun Jan 28 07:08:40 2018
  810.jsvc_up                         R        0  Sun Jan 28 03:54:31 2018
  1582.jsvc_up                        R        0  Sun Jan 28 04:01:49 2018
  1823.jsvc_up                        R        0  Sun Jan 28 02:57:44 2018

                38497656 blocks of size 1024. 8238924 blocks available
```

h. Type **quit** to exit the smbclient and return to the CLI prompt.

```
smb: > quit
```

Lab 2 – Performing On-Path Attacks with Ettercap

An On-Path Attack, also known as a **Man-in-the-Middle (MitM)** attack, happens when an attacker secretly positions themselves between two communicating devices, such as a user's browser and a web server, to **intercept, eavesdrop** on, or alter their data flow, often by manipulating DNS or hijacking sessions between them. **Ettercap** is used to perform on-path (MITM) attacks.

In this lab, we will practice a common form of on-path attack using a Kali tool and will perform the following actions.

- Launch Ettercap and Explore Its Capabilities
- Perform the On-Path (MITM) Attack
- Use Wireshark to observe the ARP Spoofing Attack

The following resources will be required to perform this lab.

- Kali VM customized for Ethical Hacker course
- Internet access

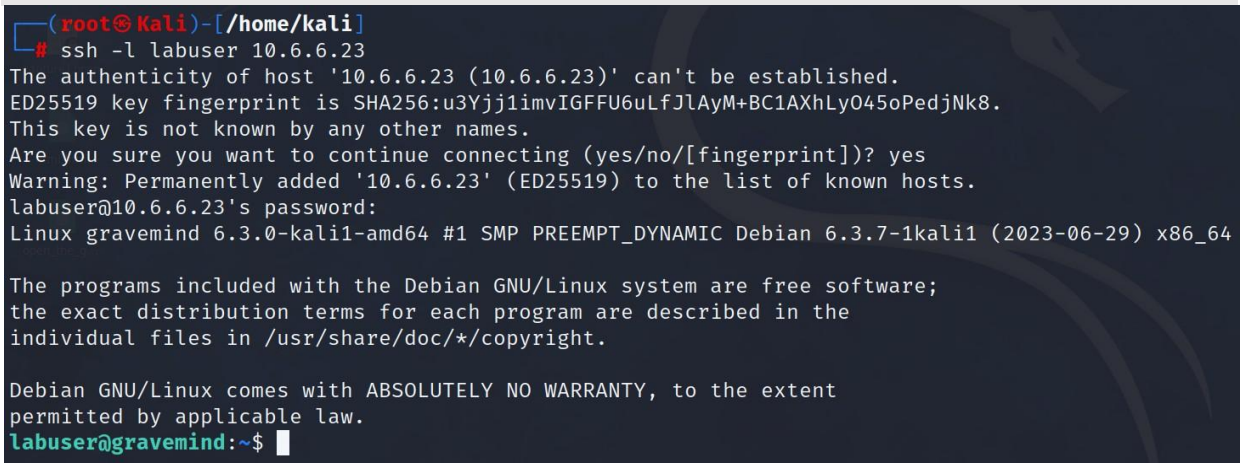
In this lab, we will use ARP spoofing to redirect traffic on the local virtual network to the Kali Linux system at **10.6.6.1**. Since our lab environment uses an internal virtual network, instead of spoofing the default gateway, we will use ARP spoofing to redirect traffic that is destined for a local server with the address **10.6.6.13**.

Part 1: Launch Ettercap and Explore Its Capabilities

a. Load Kali Linux using the username kali and the password kali. Open a terminal session from the menu bar at the top of the screen.
b. The target host in this lab is the Linux device at **10.6.6.23**. To view the network from the target perspective and initiate traffic between the target and the server, use SSH to log in to this host. The username is **labuser**, and the password is **Cisco123**.

c. The user of the 10.6.6.23 host is communicating with the server at 10.6.6.13. The on-path attacker at 10.6.6.1 (Kali VM) will intercept and relay traffic between these hosts.

```
# ssh -l labuser 10.6.6.23
```



d. Because we are creating an on-path attack that uses ARP spoofing, we will be monitoring the ARP mappings on the victim host. The attack will cause changes to those mappings.

e. Use the command **ip neighbor** to view the current ARP cache on the target computer.

```
$ ip neighbor
```

f. Open a new terminal session from the menu bar in Kali Linux. Do not close the SSH terminal that is running the session with 10.6.6.23.

g. Use the **ettercap -h** command to view the help file for the Ettercap application.

```
# ettercap -h
```

```
┌──(kali㉿Kali)-[~]
└─$ ettercap -h

ettercap 0.8.3.1 copyright 2001-2020 Ettercap Development Team

Usage: ettercap [OPTIONS] [TARGET1] [TARGET2]

TARGET is in the format MAC/IP/IPv6/PORTs (see the man for further detail)

Sniffing and Attack options:
  -M, --mitm <METHOD:ARGS>   perform a mitm attack
  -o, --only-mitm            don't sniff, only perform the mitm attack
  -b, --broadcast            sniff packets destined to broadcast
  -B, --bridge <IFACE>       use bridged sniff (needs 2 ifaces)
  -p, --nopromisc            do not put the iface in promisc mode
  -S, --nosslmitm            do not forge SSL certificates
  -u, --unoffensive          do not forward packets
  -r, --read <file>          read data from pcapfile <file>
  -f, --pcapfilter <string>  set the pcap filter <string>
  -R, --reversed             use reversed TARGET matching
  -t, --proto <proto>        sniff only this proto (default is all)
      --certificate <file>   certificate file to use for SSL MiTM
      --private-key <file>   private key file to use for SSL MiTM
```

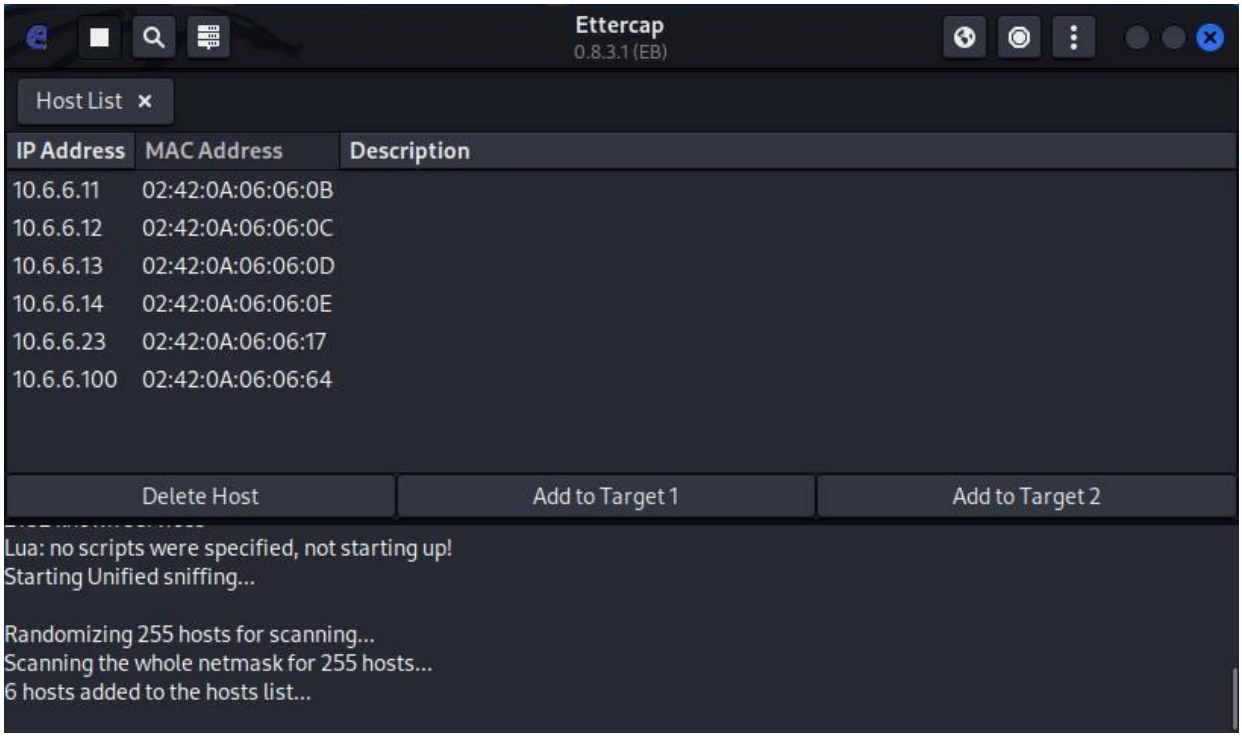
h. Start Ettercap GTK+ graphical user interface using the **ettercap -G** command. Most Ettercap functions require root permissions, so use the sudo command to obtain the required permissions.

```
# sudo ettercap -G
```

i. The Ettercap GUI opens in a new window. You are sniffing traffic on an internal, virtual network. The default setup is to scan using interface **eth0**. Change the sniffing interface to **br-internal**, which is the interface that is configured on the 10.6.6.0/24 virtual network, by changing the value in the **Setup > Primary Interface** dropdown.
j. Click the **checkbox** icon at the top right of the Ettercap screen to continue. A message appears at the bottom of the screen indicating that Unified sniffing has started.

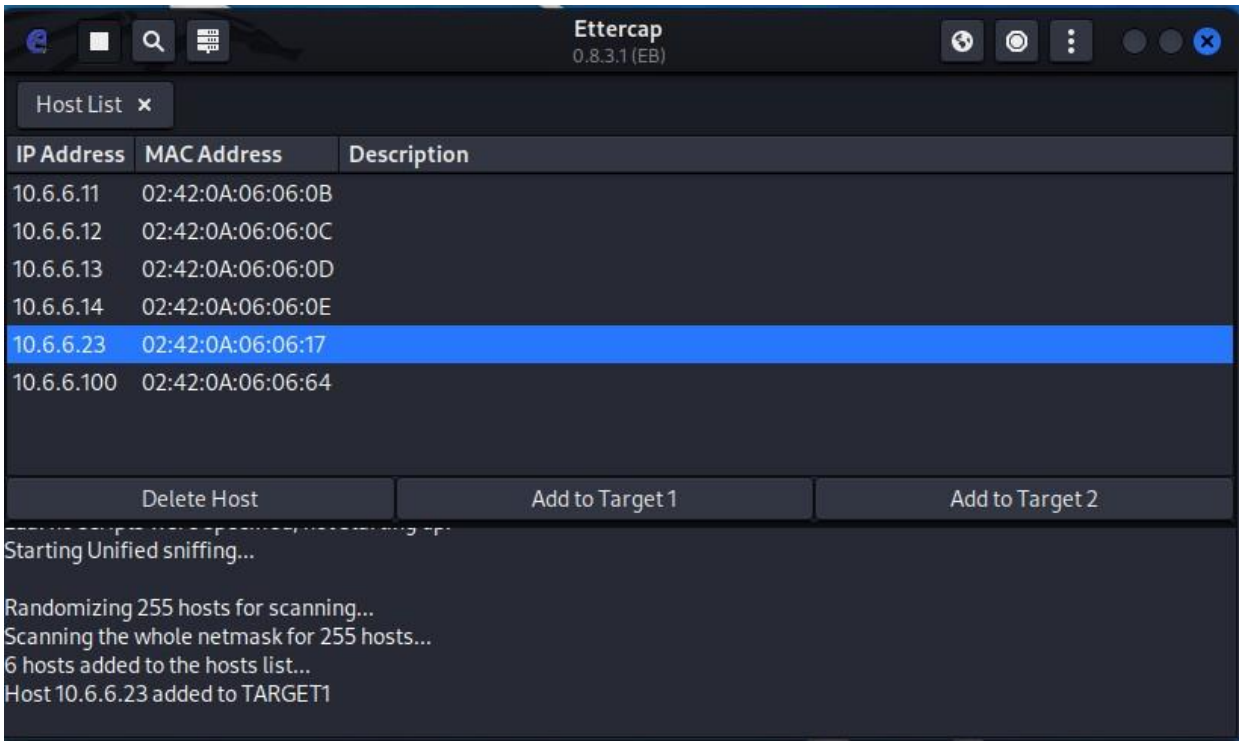
Part 2: Perform the On-Path (MITM) Attack

a. In the **Ettercap GUI** window, open the Hosts List window by clicking the Ettercap menu located at the three dots icon and selecting the Hosts entry, and then Hosts List.
b. Click the Scan for Hosts icon located at the top left in the menu bar. We will see a list of the hosts in the Host List window that were discovered on the **10.6.6.0/24** network.

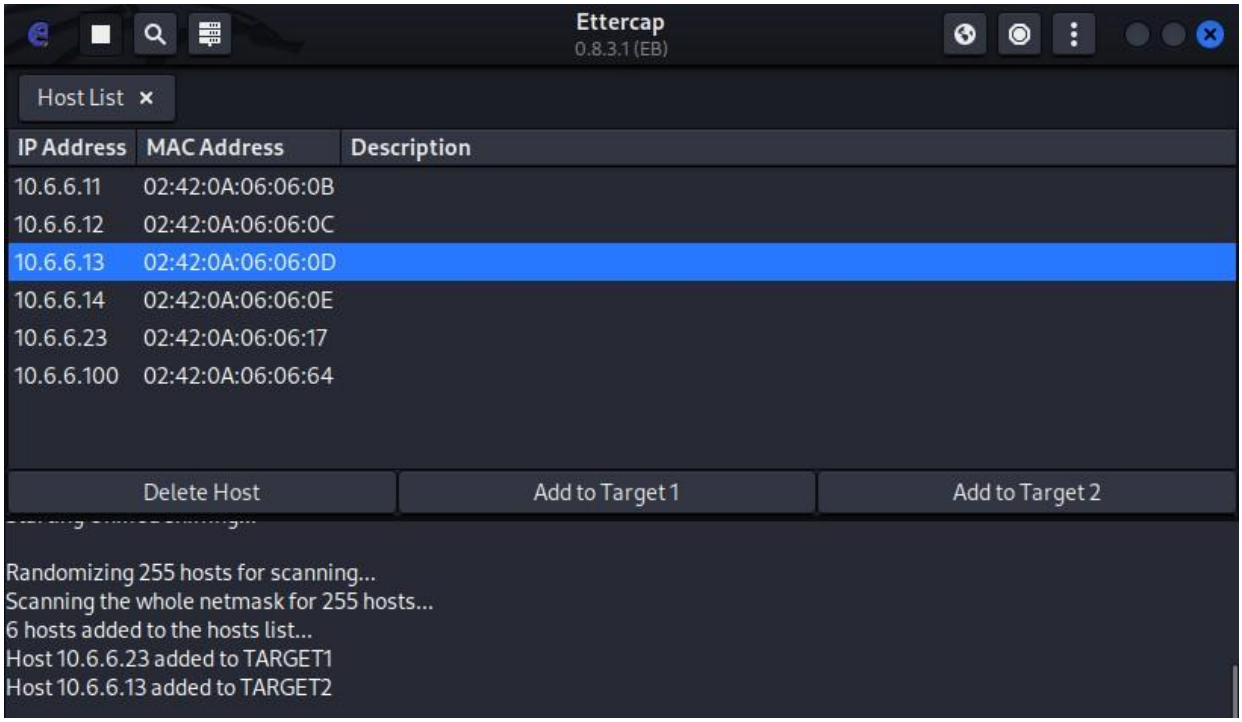


c. Now, we have to define the source and destination devices for the attack. To do so, click the IP address **10.6.6.23** in the window to highlight the target host of the user.

d. Next, click the Add to Target 1 button at the bottom of the Host List window, which defines the user's host as Target 1.

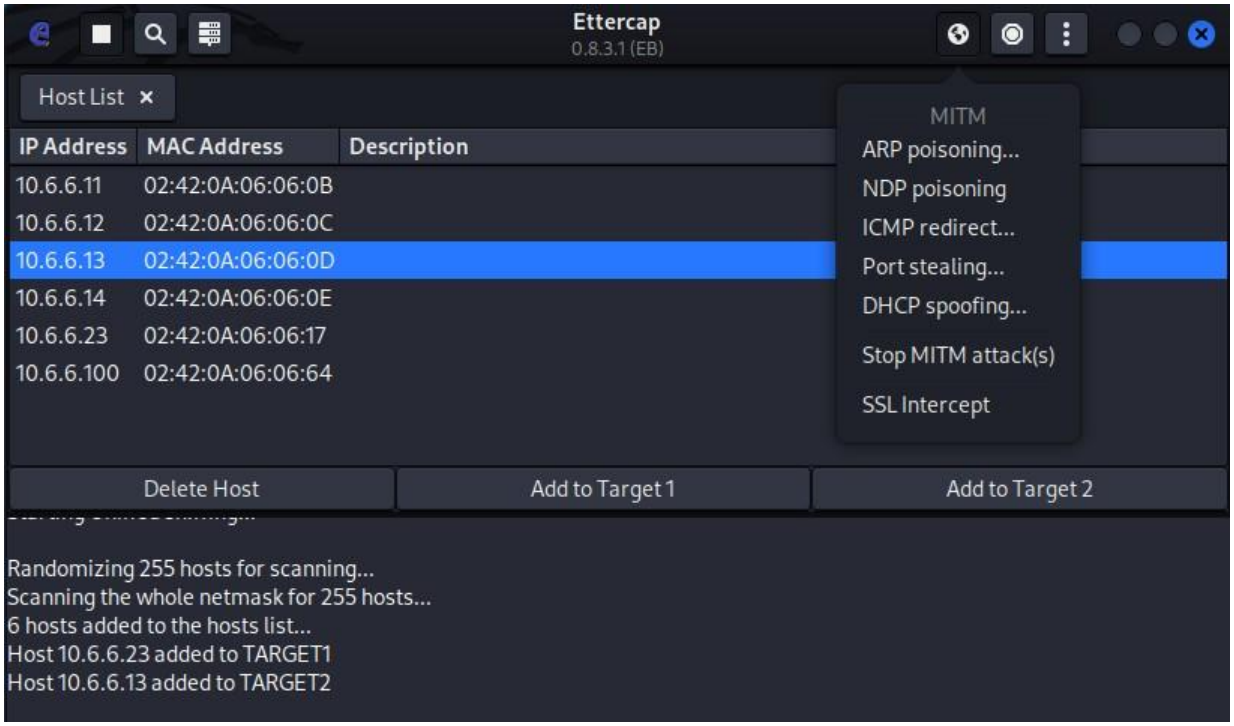


e. Click the IP address of the destination web server at **10.6.6.13** to highlight the line. Then, click the Add to Target 2 button located at the bottom of the host window.



f. Click the MITM icon on the menu bar, which is the first circular icon located on the top right corner.

g. Now, select **"ARP Poisoning"** from the dropdown menu to verify that **Sniff remote connections** are selected. Then, click on the **OK** button.



h. After that, the MITM exploit will start. If sniffing does not start immediately, click the Start option appears as a play button located at the top left of the menu.

i. Next, return to the terminal window that is running the SSH session with the target user host at **10.6.6.23** and repeat the ping to **10.6.6.13**

```
$ ping -c 5 10.6.6.13
```

```
labuser@gravemind:~$ ping -c 5 10.6.6.13
PING 10.6.6.13 (10.6.6.13) 56(84) bytes of data.
64 bytes from 10.6.6.13: icmp_seq=1 ttl=64 time=8.78 ms
64 bytes from 10.6.6.13: icmp_seq=2 ttl=64 time=19.0 ms
64 bytes from 10.6.6.13: icmp_seq=3 ttl=64 time=9.58 ms
64 bytes from 10.6.6.13: icmp_seq=4 ttl=64 time=12.3 ms
64 bytes from 10.6.6.13: icmp_seq=5 ttl=64 time=13.5 ms

--- 10.6.6.13 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 11ms
rtt min/avg/max/mdev = 8.776/12.626/19.007/3.626 ms
labuser@gravemind:~$
```

j. Use the **ip neighbor** command to view the ARP table on 10.6.6.23 again and note the MAC address listed for 10.6.6.13.

```
labuser@gravemind:~$ ip neighbor
10.6.6.13 dev eth0 lladdr 02:42:18:51:81:79 REACHABLE
10.6.6.1 dev eth0 lladdr 02:42:18:51:81:79 DELAY
labuser@gravemind:~$
```

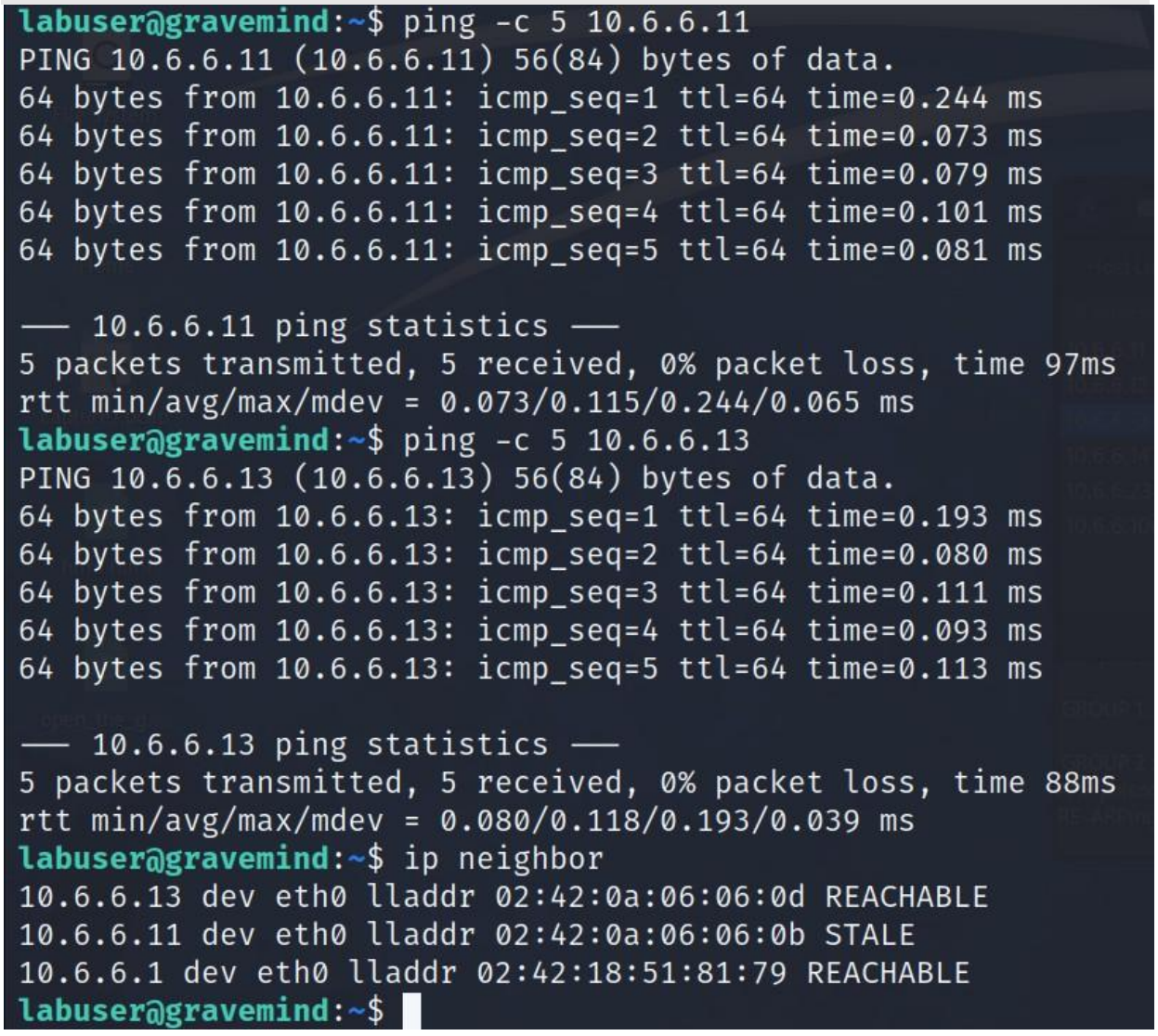
k. Close the Ettercap graphical user interface but leave the SSH connection to 10.6.6.23 active.

Part 3: Use Wireshark to Observe the ARP Spoofing Attack

Now, we will use the command-line interface in **Ettercap** to perform **ARP spoofing** and write a **.pcap** file that can be opened in Wireshark. We can refer to the help information for Ettercap to interpret the options used in the commands.

a. Return to the terminal session that is connected via SSH to 10.6.6.23. Ping the IP addresses 10.6.6.11 and 10.6.6.13. Then, use the **ip neighbor** command to find the MAC addresses associated with the IP addresses of the two systems.

```
$ ping -c 5 10.6.6.11
$ ping -c 5 10.6.6.13
$ ip neighbor
```



b. To find the MAC of **10.6.6.23**, go to the SSH session terminal and enter the **IP address** command.

```
labuser@gravemind:~$ ip address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
29: eth0@if30: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default
    link/ether 02:42:0a:06:06:17 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 10.6.6.23/24 brd 10.6.6.255 scope global eth0
       valid_lft forever preferred_lft forever
labuser@gravemind:~$
```

c. In a terminal window, enter the command as follows to save the pcap file in the current working directory:

```
$sudo ettercap -T -q -i br-internal --write mitm-saved.pcap --mitm arp /10.6.6.23// /10.6.6.13//
```

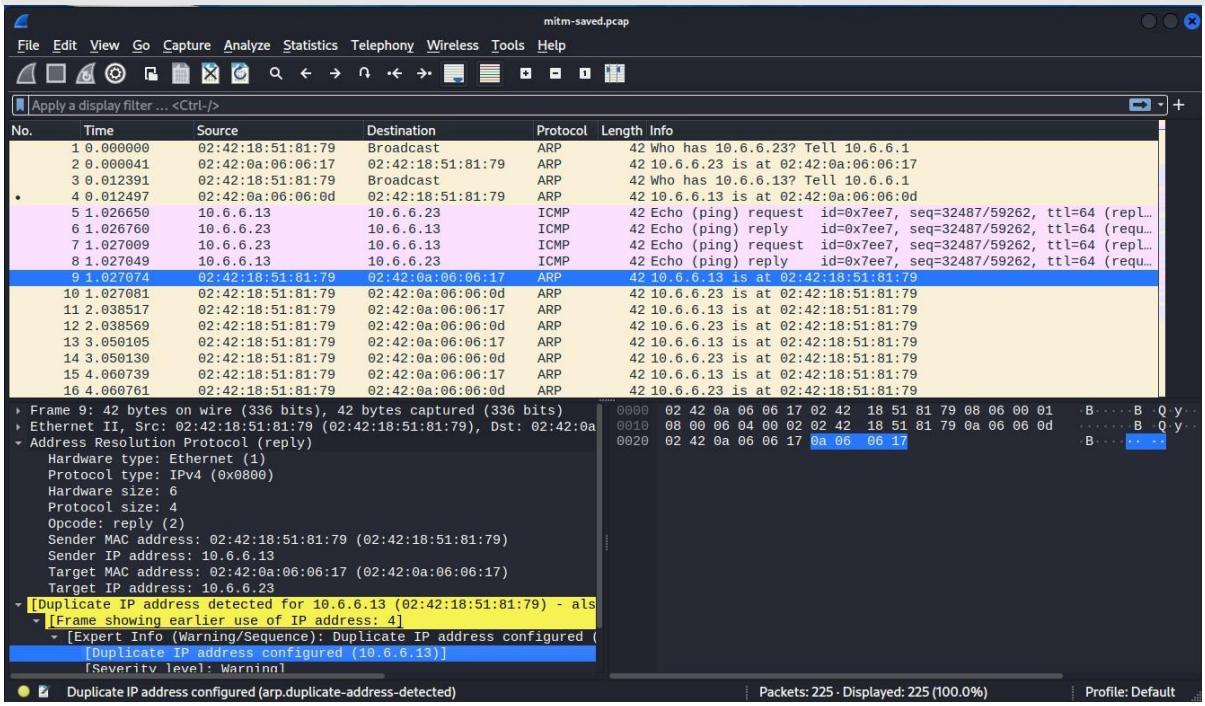
d. Return to the SSH terminal session to **10.6.6.23**. Ping the two IP addresses, **10.6.6.11** and **10.6.6.13**, again and use the ip neighbor command to view the associated MAC addresses.

```
labuser@gravemind:~$ ping -c 5 10.6.6.11
PING 10.6.6.11 (10.6.6.11) 56(84) bytes of data.
64 bytes from 10.6.6.11: icmp_seq=1 ttl=64 time=0.140 ms
64 bytes from 10.6.6.11: icmp_seq=2 ttl=64 time=0.139 ms
64 bytes from 10.6.6.11: icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from 10.6.6.11: icmp_seq=4 ttl=64 time=0.075 ms
64 bytes from 10.6.6.11: icmp_seq=5 ttl=64 time=0.077 ms

—— 10.6.6.11 ping statistics ——
5 packets transmitted, 5 received, 0% packet loss, time 103ms
rtt min/avg/max/mdev = 0.067/0.099/0.140/0.034 ms
labuser@gravemind:~$ ping -c 5 10.6.6.13
PING 10.6.6.13 (10.6.6.13) 56(84) bytes of data.
64 bytes from 10.6.6.13: icmp_seq=1 ttl=64 time=13.8 ms
64 bytes from 10.6.6.13: icmp_seq=2 ttl=64 time=12.0 ms
64 bytes from 10.6.6.13: icmp_seq=3 ttl=64 time=11.1 ms
64 bytes from 10.6.6.13: icmp_seq=4 ttl=64 time=9.25 ms
64 bytes from 10.6.6.13: icmp_seq=5 ttl=64 time=12.0 ms

—— 10.6.6.13 ping statistics ——
5 packets transmitted, 5 received, 0% packet loss, time 10ms
rtt min/avg/max/mdev = 9.252/11.630/13.810/1.481 ms
labuser@gravemind:~$ ip neighbor
10.6.6.13 dev eth0 lladdr 02:42:18:51:81:79 REACHABLE
10.6.6.11 dev eth0 lladdr 02:42:0a:06:06:0b REACHABLE
10.6.6.1 dev eth0 lladdr 02:42:18:51:81:79 REACHABLE
labuser@gravemind:~$
```

e. Close the **SSH terminal** session that is connected to 10.6.6.23 and return to the terminal session running Ettercap in text mode, then press **Ctrl+C** to quit Ettercap.

f. In the Kali terminal window, start Wireshark with the mitm-saved.pcap file that you created with Ettercap.

```
$ wireshark mitm-saved.pcap
```



g. With the Ettercap attack, computer first broadcasts ARP requests to obtain the actual MAC addresses for the two target hosts, 10.6.6.23 and 10.6.6.11. The attacking machine then begins to send ARP responses to both target hosts using its own MAC for both IP addresses. This causes the two target hosts to address the Ethernet frames to the attacker's computer, which enables it to collect data as an on-path attacker.