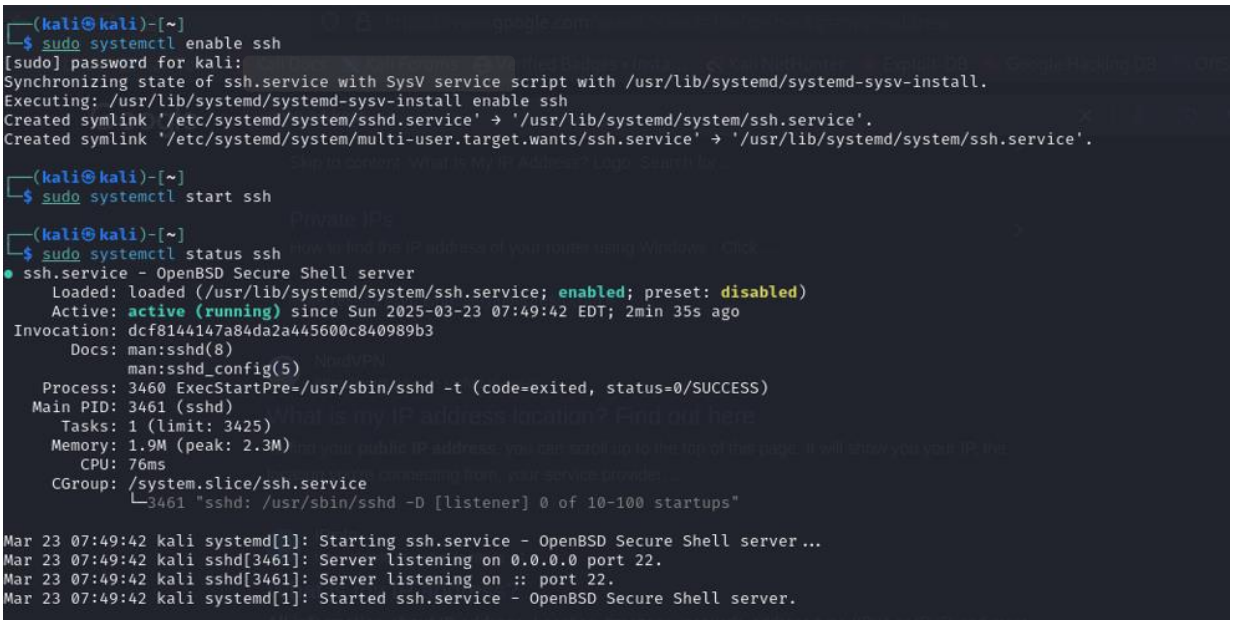
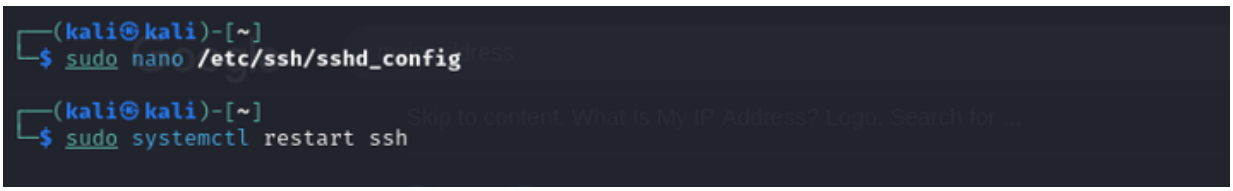
Task 2 : Remote Access & SSH Hardening

Setup: Enabling SSH & Weak Configuration :

1. To initiate the SSH service, we first enable it using `sudo systemctl enable ssh` , then `sudo systemctl start ssh` to ensure it is running and ready for remote access.



2. Next, we modify the SSH configuration to permit root login and enable password authentication by editing the `/etc/ssh/sshd_config` file. Then we restart the ssh service.



3. Find PermitRootLogin and PasswordAuthentication and give yes to this

```
#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no
```

Exploitation: Brute-Forcing SSH

1.Create a wordlist and run hydra to Brute-force ssh

```
┌──(kali㉿kali)-[~]
└─$ hydra -L wordlist.txt -P wordlist.txt ssh://182.74.154.218
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 2025-03-23 08:26:42
[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, 25 login tries (l:5/p:5), ~2 tries per task
```

2. We use **Hydra** to perform a brute-force SSH root login using a customgenerated wordlist, targeting our own machine's IP address. This allows us to test authentication security and assess password strength.

Mitigation:

1. Edit SSH Configuration:

   Open the SSH configuration file and modify the following lines:

   Disable Root Login and PasswordAuthentication

```
# Authentication:

#LoginGraceTime 2m
PermitRootLogin no
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile     .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
```

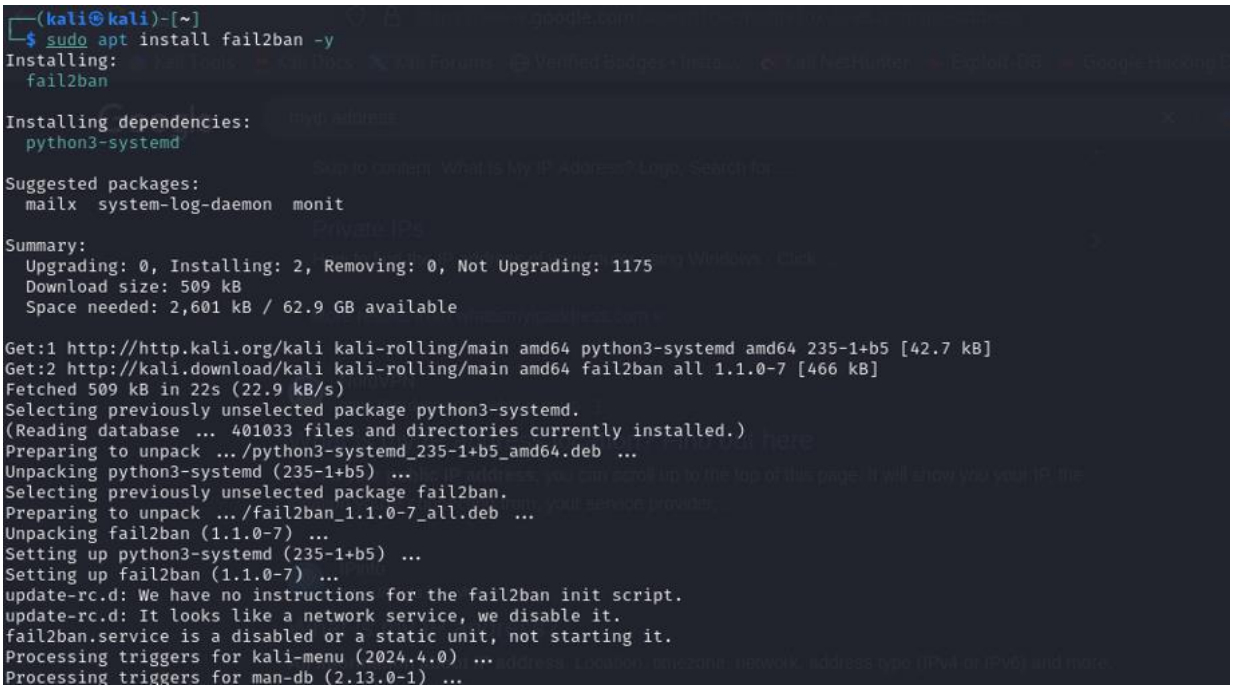
4 . To enhance authentication security, generate an SSH key pair on the client machine using `ssh-keygen -t rsa -b 4096` . Next, copy the public key to the server with `ssh-copy-id user@<server-IP>` , and finally, restart the SSH service using `sudo systemctl restart ssh` to apply the changes.

```
┌──(kali㉿kali)-[~]
└─$ ssh-keygen -t rsa -b 4096

Generating public/private rsa key pair.
Enter file in which to save the key (/home/kali/.ssh/id_rsa): wordlist.txt
wordlist.txt already exists.
Overwrite (y/n)? y
Enter passphrase for "wordlist.txt" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in wordlist.txt
Your public key has been saved in wordlist.txt.pub
The key fingerprint is:
SHA256:0hBG6Q6Sy2CVUlt+SO2RXuKpNeaPpSIWZ6JKhAmxtdw kali@kali
The key's randomart image is:
+---[RSA 4096]----+
|. o..+=..        |
| * ==.o* .       |
|o =oE+=.=        |
|+oo . o@         |
|++ o o* S        |
|. oo +.o .       |
| .. =   =        |
|.. o . o .       |
|o . . .          |
+----[SHA256]-----+
```

Configure Fail2Ban to Prevent Brute-Force Attacks

1. To enhance system security, install **Fail2Ban** by running `sudo apt install fail2ban -y`, which helps protect against brute-force attacks by monitoring and blocking suspicious login attempts.



2. To configure **Fail2Ban**, edit the jail configuration file using `sudo nano /etc/fail2ban/jail.local`, then add the following settings under `[sshd]`: `enabled = true`, `maxretry = 3`, and `bantime = 600`, ensuring protection against repeated failed SSH login attempts.

```
 GNU nano 8.2
[sshd]
enabled = true
port = ssh
maxretry = 3
bantime = 600
```

3.Finally restart fail2ban to avoid ssh attacks.

```
┌──(kali㉿kali)-[~]
└─$ sudo systemctl restart fail2ban
```

7. Conclusion

Successfully set up SSH and performed a brute-force attack.

Implemented security measures including disabling root login, key-based authentication, and Fail2Ban.

Ensured SSH is hardened against unauthorized access attempts.