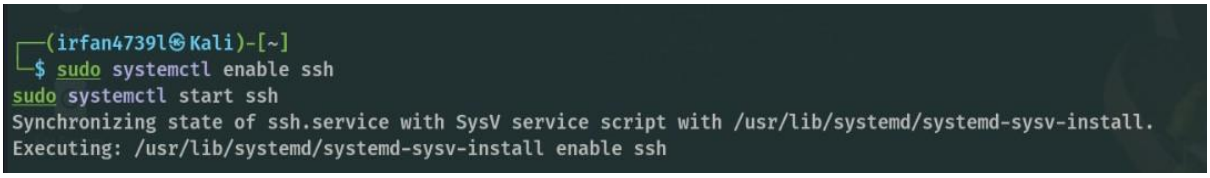
TASK 2: Remote Access & SSH Hardening

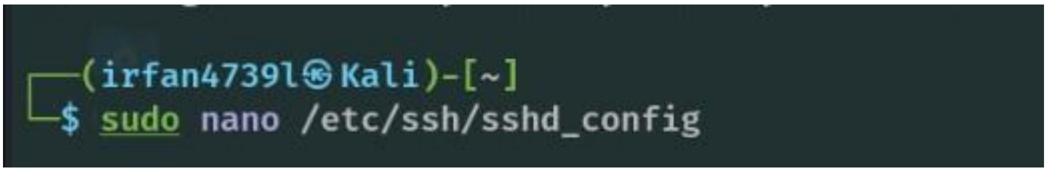
Setup: Enabling SSH & Weak Configuration 🔑 :

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



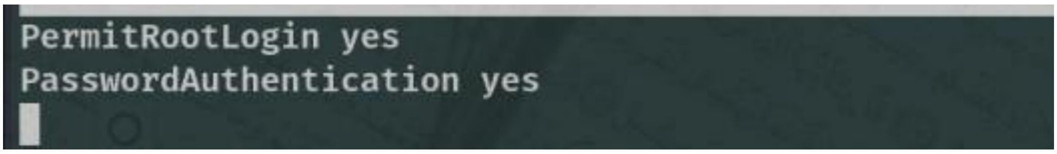
```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo systemctl enable ssh
sudo systemctl start ssh
Synchronizing state of ssh.service with SysV service script with /usr/lib/systemd/systemd-sysv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable ssh
```

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



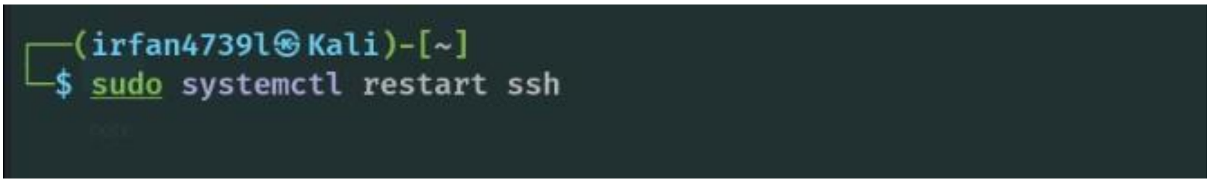
```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo nano /etc/ssh/sshd_config
```

3. Update the **PermitRootLogin** and **PasswordAuthentication** parameters to yes.



```
PermitRootLogin yes
PasswordAuthentication yes
```

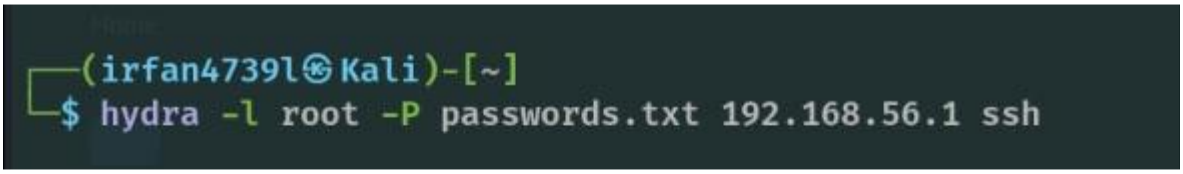
4. Then we restart the ssh service.



```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo systemctl restart ssh
```

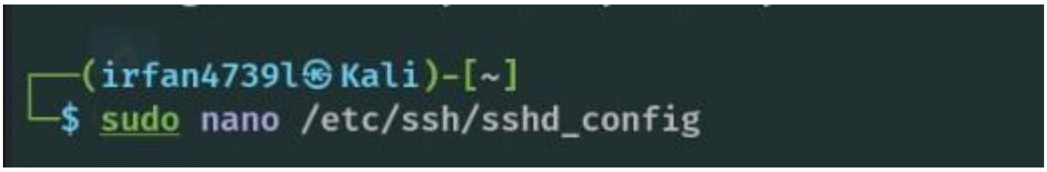
Exploitation: Brute-Forcing SSH 🛠:

1. 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.



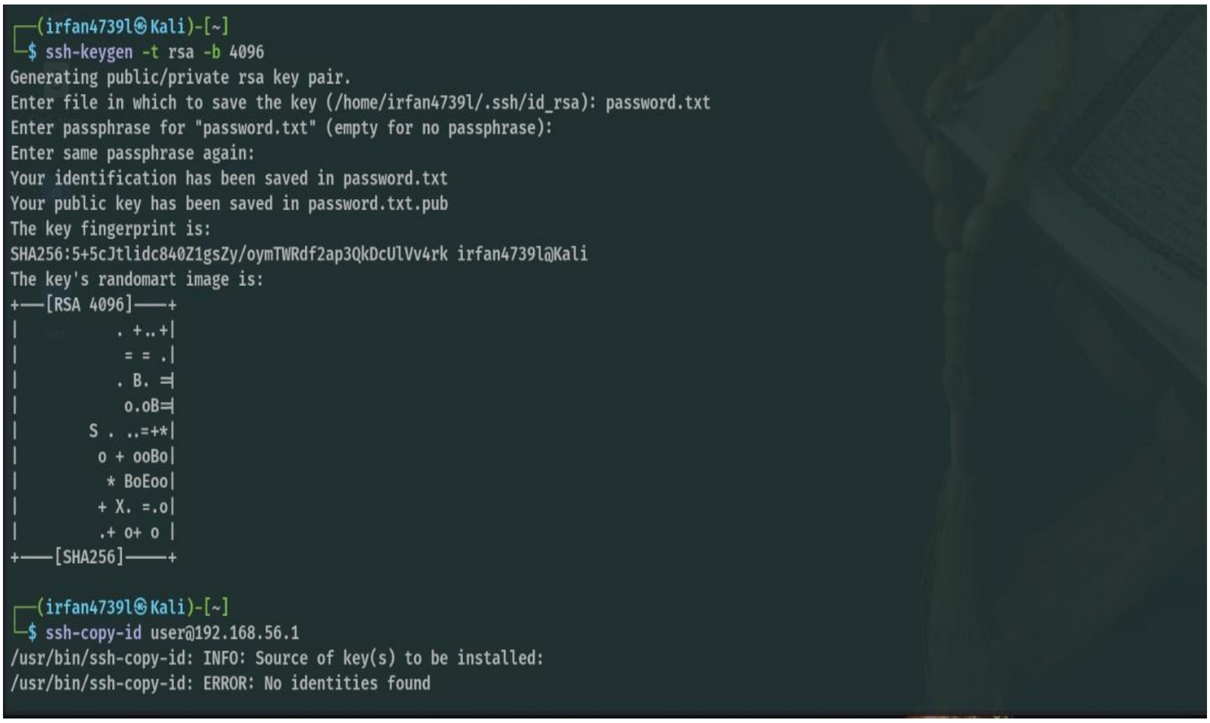
```
┌──(irfan4739l㉿Kali)-[~]
└─$ hydra -l root -P passwords.txt 192.168.56.1 ssh
```

2. To enhance security, root login and password authentication are disabled by setting **PermitRootLogin** no and **PasswordAuthentication** no in the SSH configuration file, followed by restarting the SSH service to apply the changes.



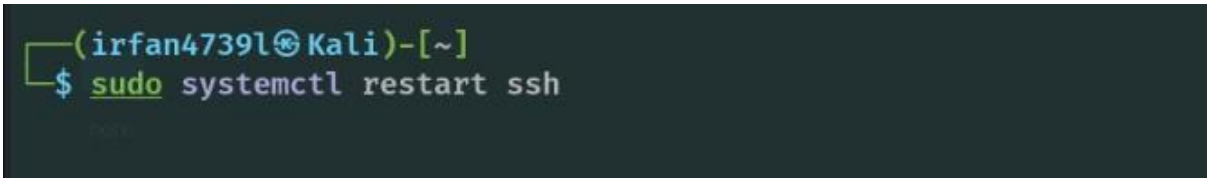
```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo nano /etc/ssh/sshd_config
```

3 . 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



```
┌──(irfan4739l㉿Kali)-[~]
└─$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/irfan4739l/.ssh/id_rsa): password.txt
Enter passphrase for "password.txt" (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in password.txt
Your public key has been saved in password.txt.pub
The key fingerprint is:
SHA256:5+5cJtlidc840Z1gsZy/oymTWRdf2ap3QkDcUlVv4rk irfan4739l@Kali
The key's randomart image is:
+---[RSA 4096]----+
|          . +..+|
|           = = .|
|          . B. =|
|           o.oB=|
|       S . ..=+*|
|        o + ooBo|
|         * BoEoo|
|        + X. =.o|
|        .+ o+ o |
+----[SHA256]-----+

┌──(irfan4739l㉿Kali)-[~]
└─$ ssh-copy-id user@192.168.56.1
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
/usr/bin/ssh-copy-id: ERROR: No identities found
```

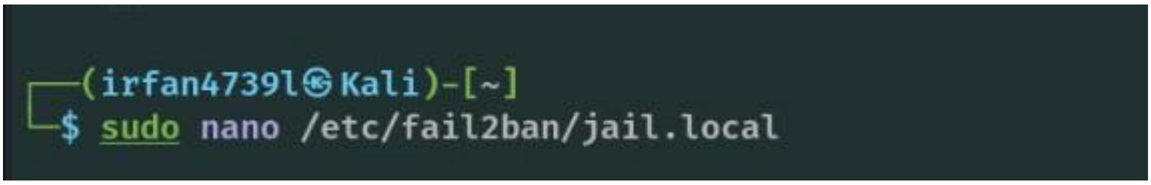


```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo systemctl restart ssh
```

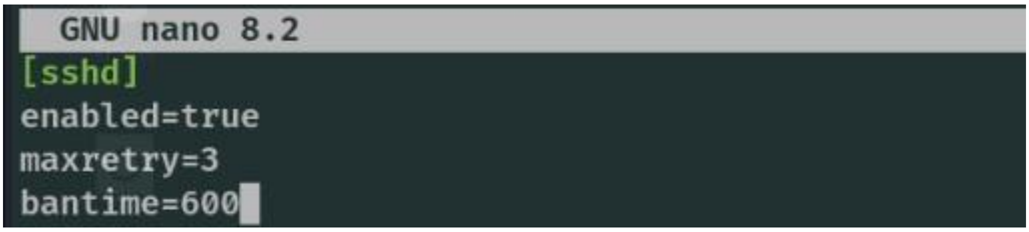
Configure Fail2Ban to Prevent Brute-Force Attacks:

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.

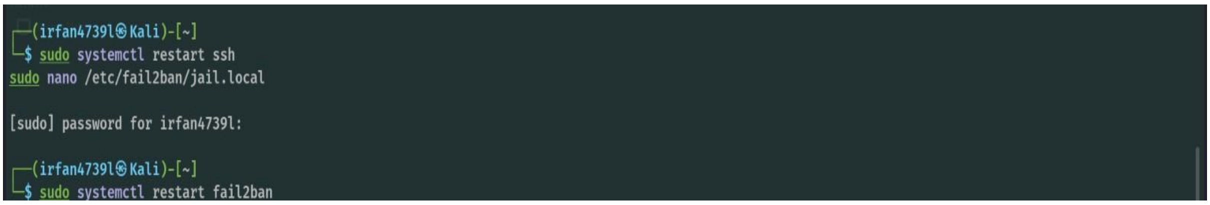


```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo nano /etc/fail2ban/jail.local
```



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

3. Finally restart fail2ban to avoid ssh attacks.



```
┌──(irfan4739l㉿Kali)-[~]
└─$ sudo systemctl restart ssh
sudo nano /etc/fail2ban/jail.local

[sudo] password for irfan4739l:

┌──(irfan4739l㉿Kali)-[~]
└─$ sudo systemctl restart fail2ban
```