How I remotely accessed my Kali Linux machine from my host machine (Windows 11) using SSH.

The goal was to simulate real-world challenges in securing remote administration while testing defense-in-depth techniques against brute-force attacks and access restrictions.

Project Overview

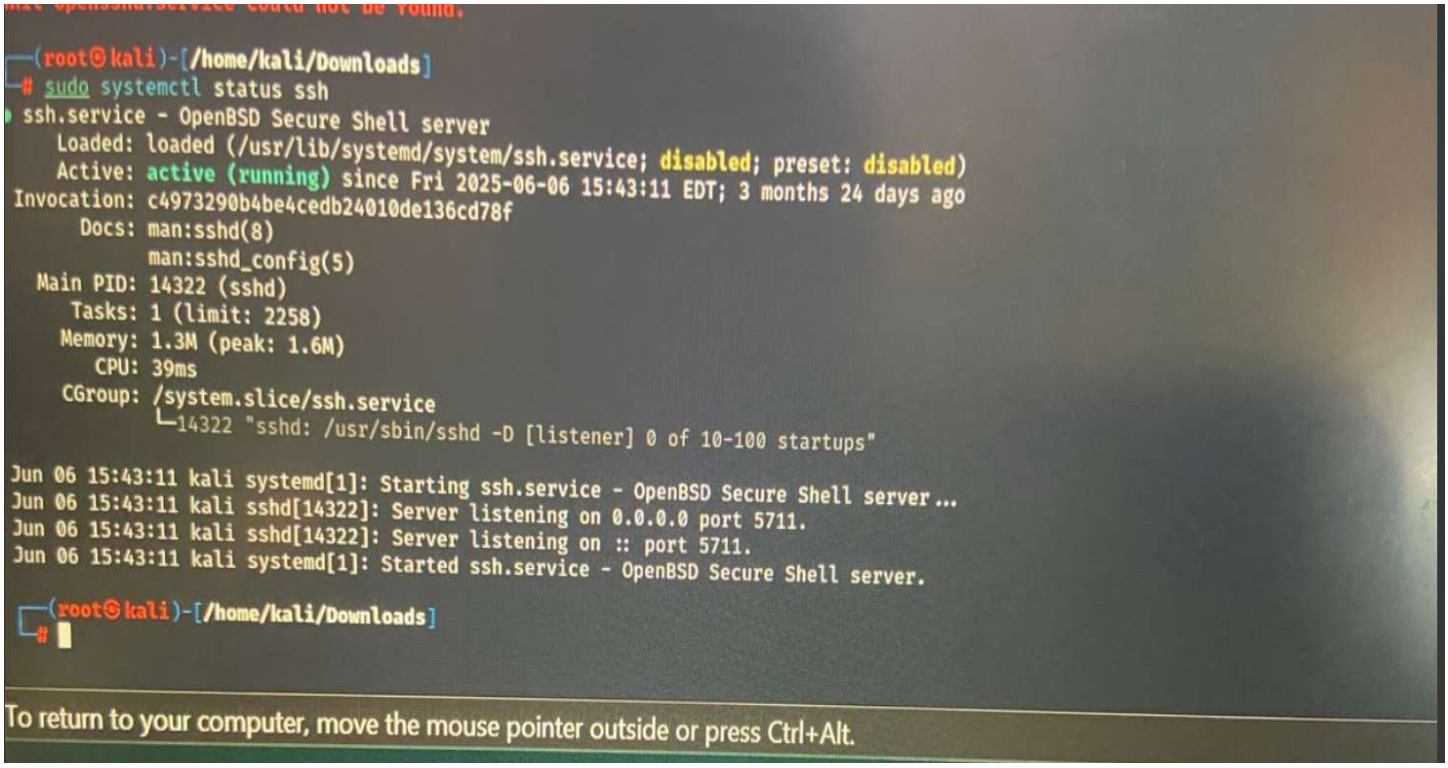
This project demonstrates how to set up and secure SSH access to a Kali Linux machine from a Windows 11 environment. The objective was to simulate **real-world remote access challenges** and apply **defense-in-depth techniques** such as firewall configuration, port obfuscation, intrusion prevention, and tunneling.

The project also explored how attackers attempt brute-force login attempts and how tools like **Fail2Ban** respond by monitoring logs and blocking malicious IPs.

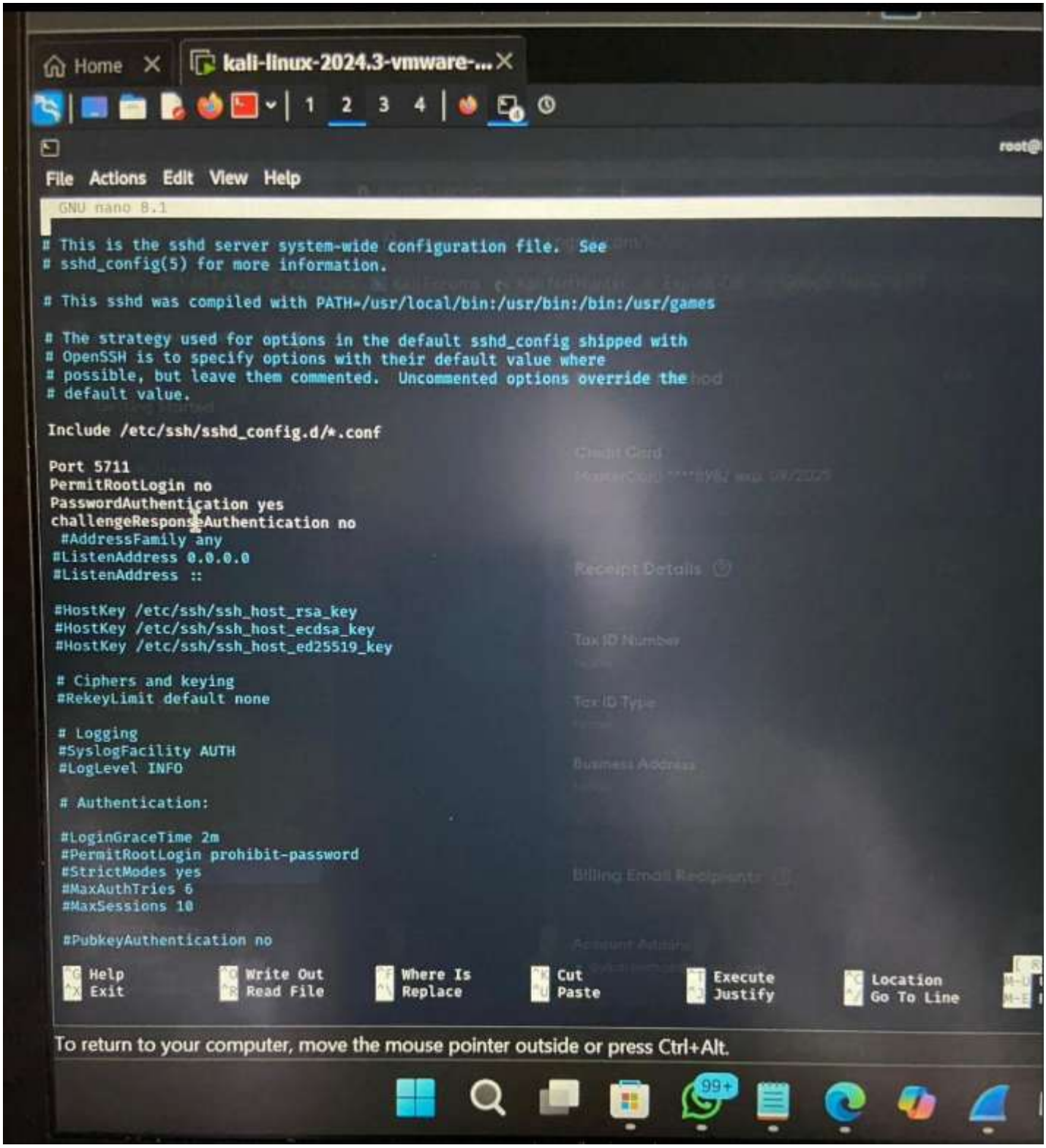
Here is how I was able to do that step by step::

Step 1: OpenSSH Configuration

Installed and enabled OpenSSH server on Kali Linux.

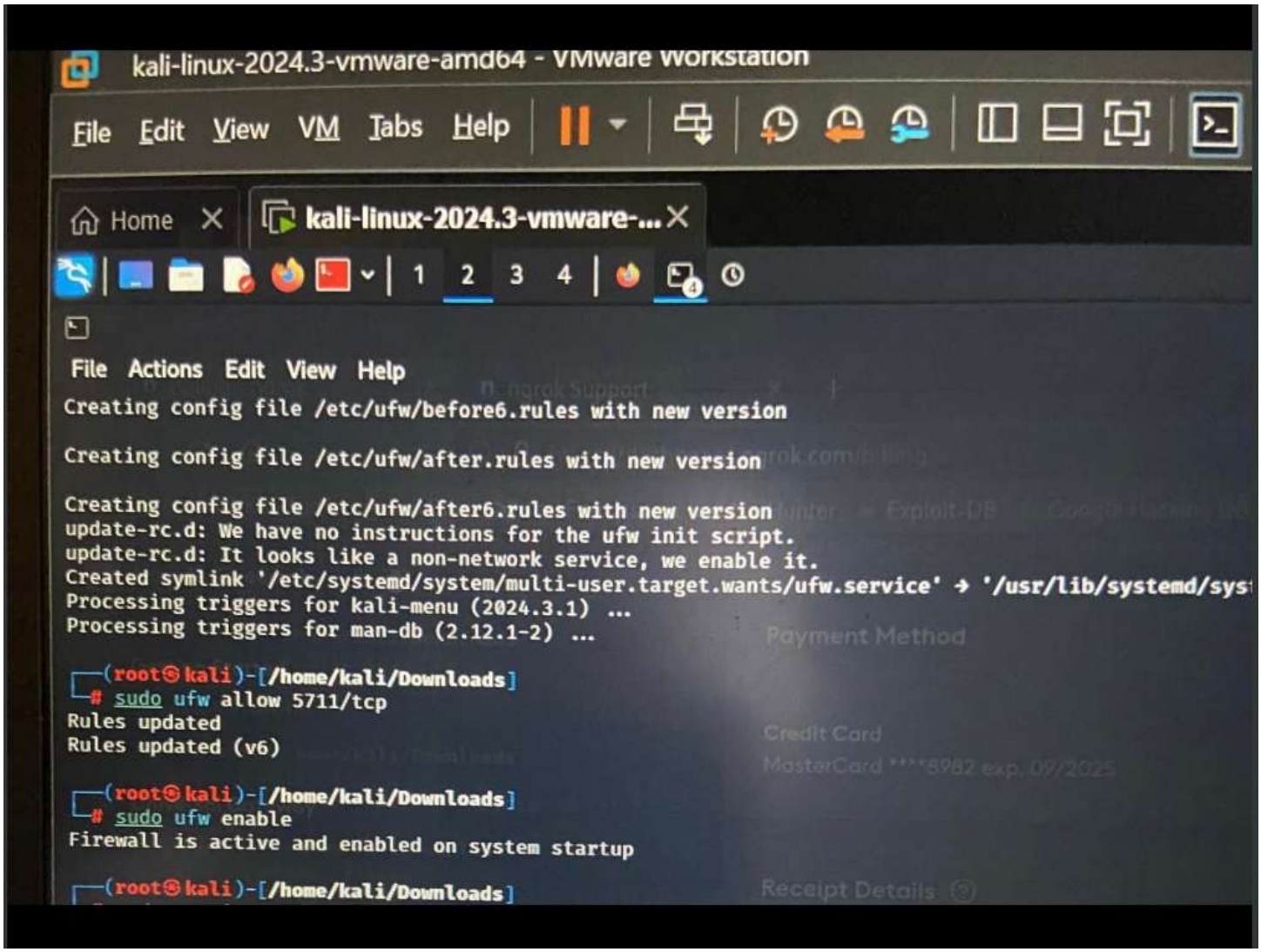


Changed the default SSH port from 22 to 5711 to reduce automated attack exposure.



STEP 2: Firewall Configuration

- Enabled **UFW (Uncomplicated Firewall)** on Kali Linux.
- Allowed only the new SSH port (5711) for inbound connections.



STEP 3:

Installed Fail2Ban to monitor authentication logs and automatically block IPs with repeated failed logins. Configured jails to ban IPs after multiple failed SSH login attempts.

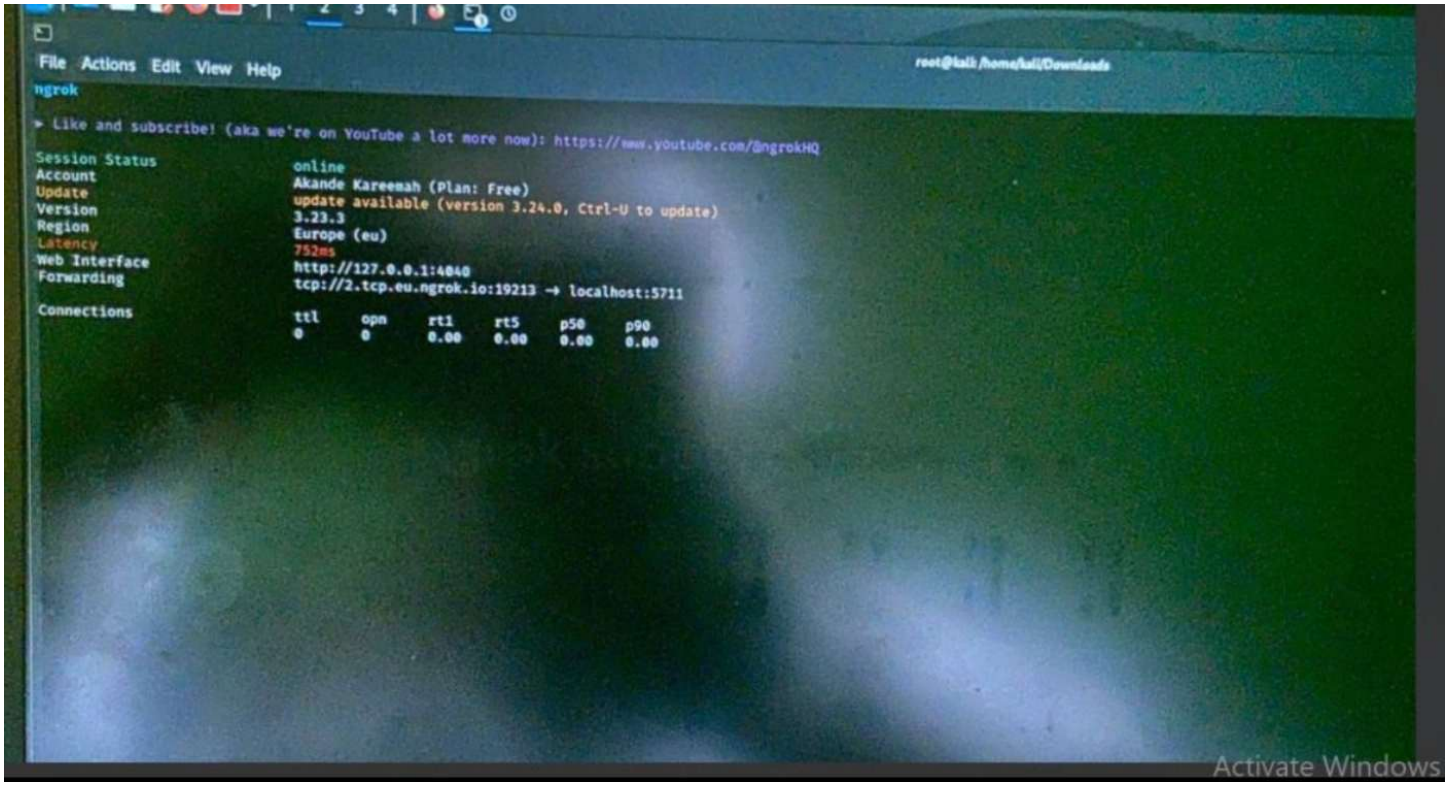
```
$ sudo apt install fail2ban -y
```

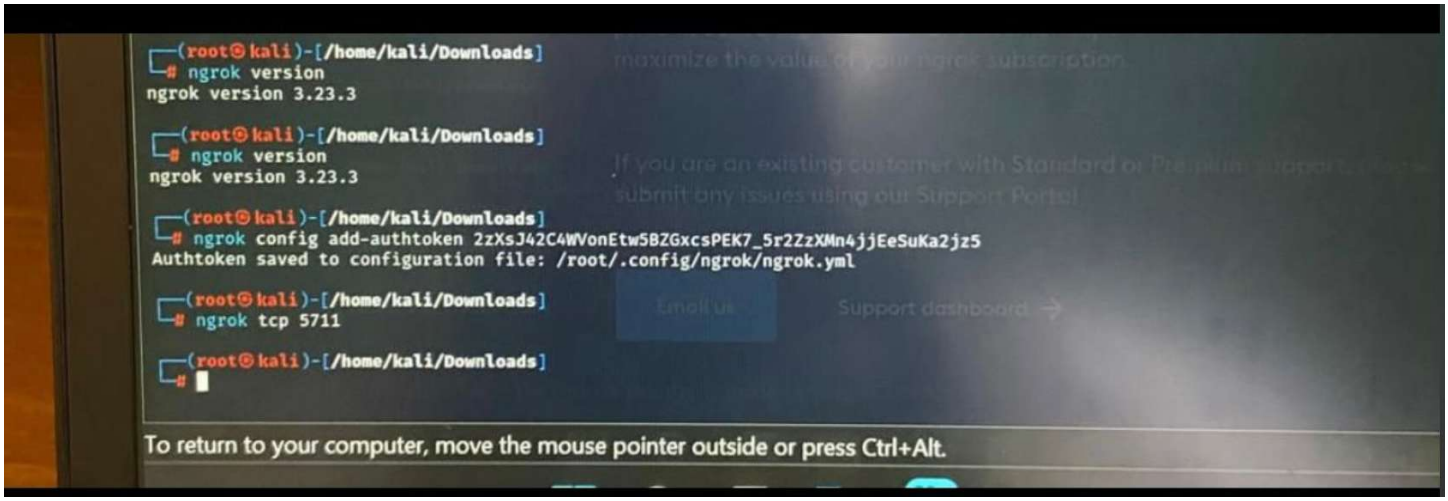
```
Installing:
  fail2ban
```

```
[sshd]
enable = true
port = 5711
maxretry = 3
bantime = 3600
```

STEP 4:

- Attempted direct port forwarding but discovered ISP blocks inbound connections.
- Implemented **Ngrok TCP tunneling** as a secure workaround, enabling external SSH connections despite ISP restrictions.





STEP 5:

- Conducted failed login attempts from Windows terminal using incorrect passwords.
- Observed how the system logged these attempts.

kali@kali: ~/Downloads

```
The authenticity of host '[192.168.91.128]:5711 ([192.168.91.128]:5711)' can't be
established.
ED25519 key fingerprint is SHA256:XDpH9D5bs7dNEOUSNXLkQR81EOqBJwYDw0EZNm/CzTE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? y
Please type 'yes', 'no' or the fingerprint: yes
Warning: Permanently added '[192.168.91.128]:5711' (ED25519) to the list of known
hosts.
kali@192.168.91.128's password:
Linux kali 6.8.11-amd64 #1 SMP PREEMPT_DYNAMIC Kali 6.8.11-1kali2 (2024-05-30) x86
_64

The programs included with the Kali GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Kali GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
┌──(kali㉿kali)-[~]
└─$ cd Downloads

┌──(kali㉿kali)-[~/Downloads]
└─$ ls
echo          ngrok-v3-stable-linux-amd64.tgz
kali-key.asc  ngrok-v3-stable-windows-amd64.zip

┌──(kali㉿kali)-[~/Downloads]
└─$
```

Command Prompt

```
Microsoft Windows [Version 10.0.26100.4351]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Me>ssh -p 5711 kali@192/168.91.128
ssh: Could not resolve hostname 192/168.91.128: No such hos
t is known.

C:\Users\Me>ssh -p 5711 kali@192.168.91.128
kali@192.168.91.128's password:
Permission denied, please try again.
kali@192.168.91.128's password:
Permission denied, please try again.
kali@192.168.91.128's password:
kali@192.168.91.128: Permission denied (publickey,password)
.

C:\Users\Me>
```

Results

- Remote SSH access was successfully established and hardened.
- Firewall and Fail2Ban effectively mitigated brute-force attempts.
- Ngrok provided a reliable solution to ISP connection restrictions.
- Logs captured real-world attack behavior, demonstrating detection and prevention.

Key Learnings

- Importance of **defense-in-depth** (port obfuscation, firewall, intrusion prevention).
- Troubleshooting real-world constraints like **ISP port blocking**.
- Monitoring logs is critical for detecting unauthorized access attempts.
- Balancing usability (password-based access from multiple devices) with security.

Tools Used

- **Kali Linux** (OpenSSH, Fail2Ban, UFW)
- **Ngrok** (TCP tunneling service)
- **Windows 11 Pro** (remote access testing client)

Conclusion

This project provided hands-on experience in **Linux system hardening, secure remote access setup, and intrusion prevention**. By simulating real-world challenges such as brute-force attacks and ISP restrictions, I learned how to apply **defense-in-depth strategies** to maintain secure and reliable remote access.