

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

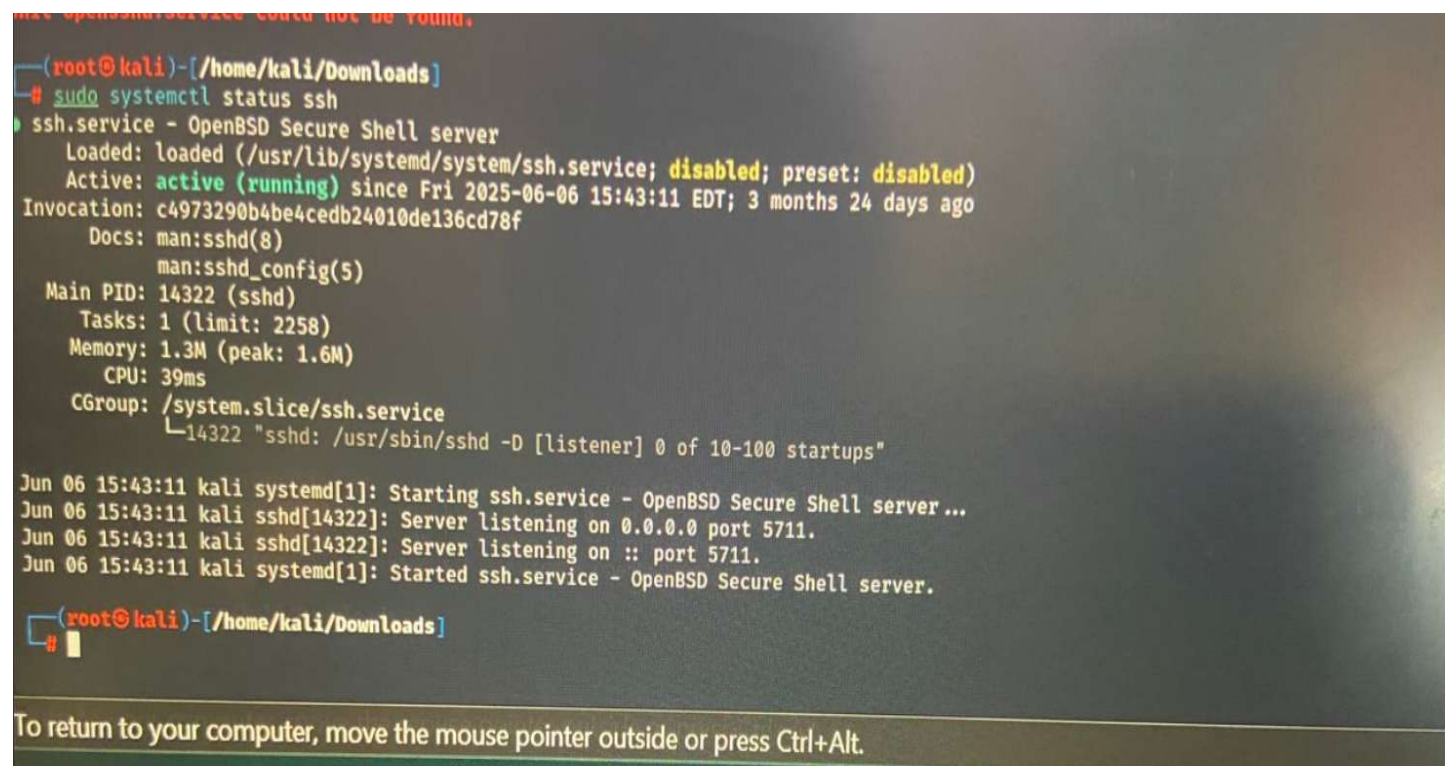
This project demonstrates how to set up and secure SSH access to a Kali Linux machine from a Windows 11 environment. The objective was to simulate **real-world remote access challenges** and apply **defense-in-depth techniques** such as firewall configuration, port obfuscation, intrusion prevention,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



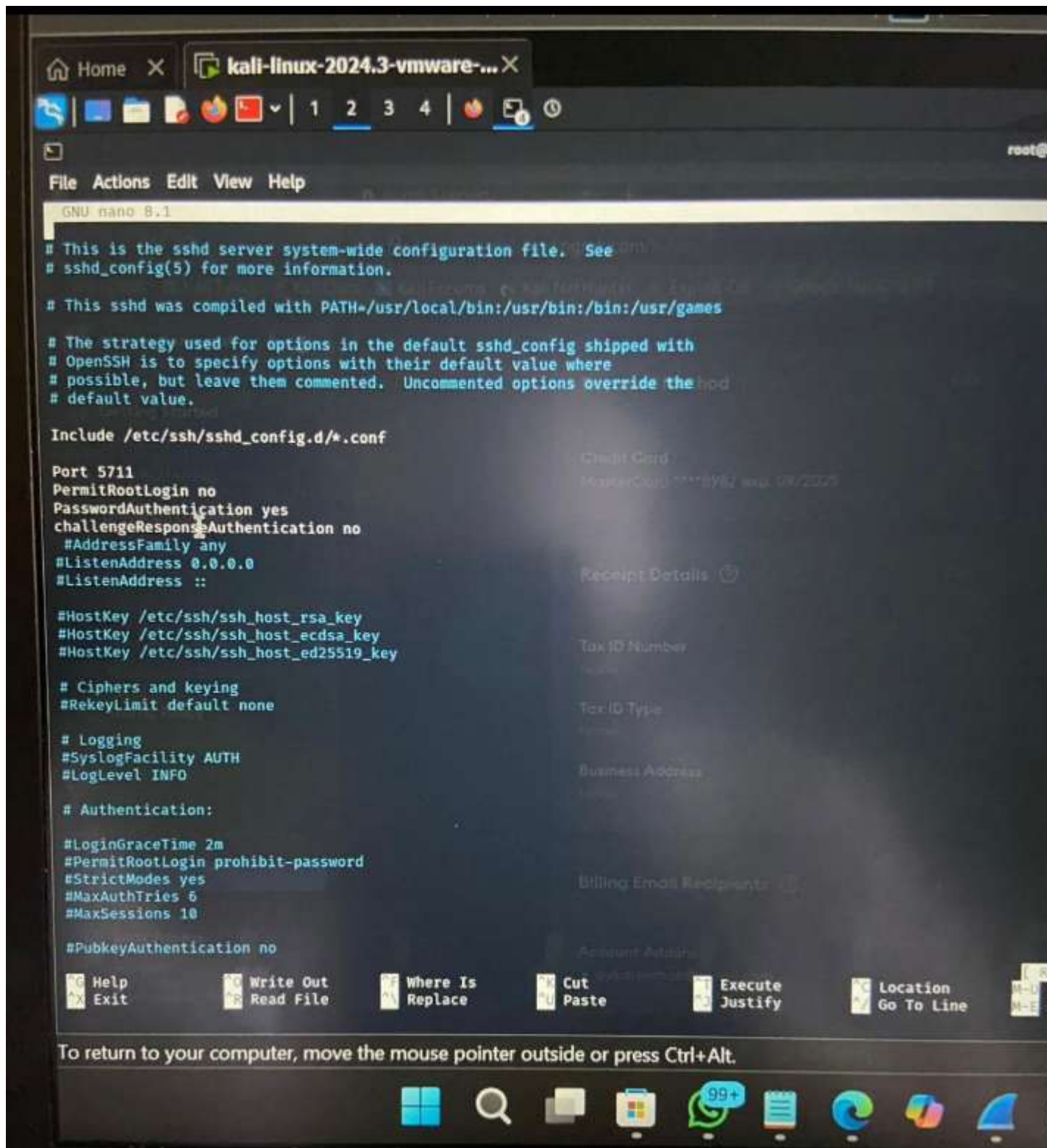
```
(root@kali)-[/home/kali/Downloads]
# sudo systemctl status ssh
ssh.service - OpenBSD Secure Shell server
  Loaded: loaded (/usr/lib/systemd/system/ssh.service; disabled; preset: disabled)
  Active: active (running) since Fri 2025-06-06 15:43:11 EDT; 3 months 24 days ago
  Invocation: c4973290b4be4cedb24010de136cd78f
  Docs: man:sshd(8)
       man:sshd_config(5)
  Main PID: 14322 (sshd)
  Tasks: 1 (limit: 2258)
  Memory: 1.3M (peak: 1.6M)
  CPU: 39ms
  CGroup: /system.slice/ssh.service
          └─14322 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jun 06 15:43:11 kali systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Jun 06 15:43:11 kali sshd[14322]: Server listening on 0.0.0.0 port 5711.
Jun 06 15:43:11 kali sshd[14322]: Server listening on :: port 5711.
Jun 06 15:43:11 kali systemd[1]: Started ssh.service - OpenBSD Secure Shell server.

(root@kali)-[/home/kali/Downloads]
#
```

To return to your computer, move the mouse pointer outside or press Ctrl+Alt.

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



The screenshot shows a terminal window titled "kali-linux-2024.3-vmware-..." with a root prompt. The GNU nano 8.1 editor is open, displaying the /etc/ssh/sshd\_config file. The configuration is as follows:

```
# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/bin:/usr/bin:/bin:/usr/games

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 5711
PermitRootLogin no
PasswordAuthentication yes
challengeResponseAuthentication no
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

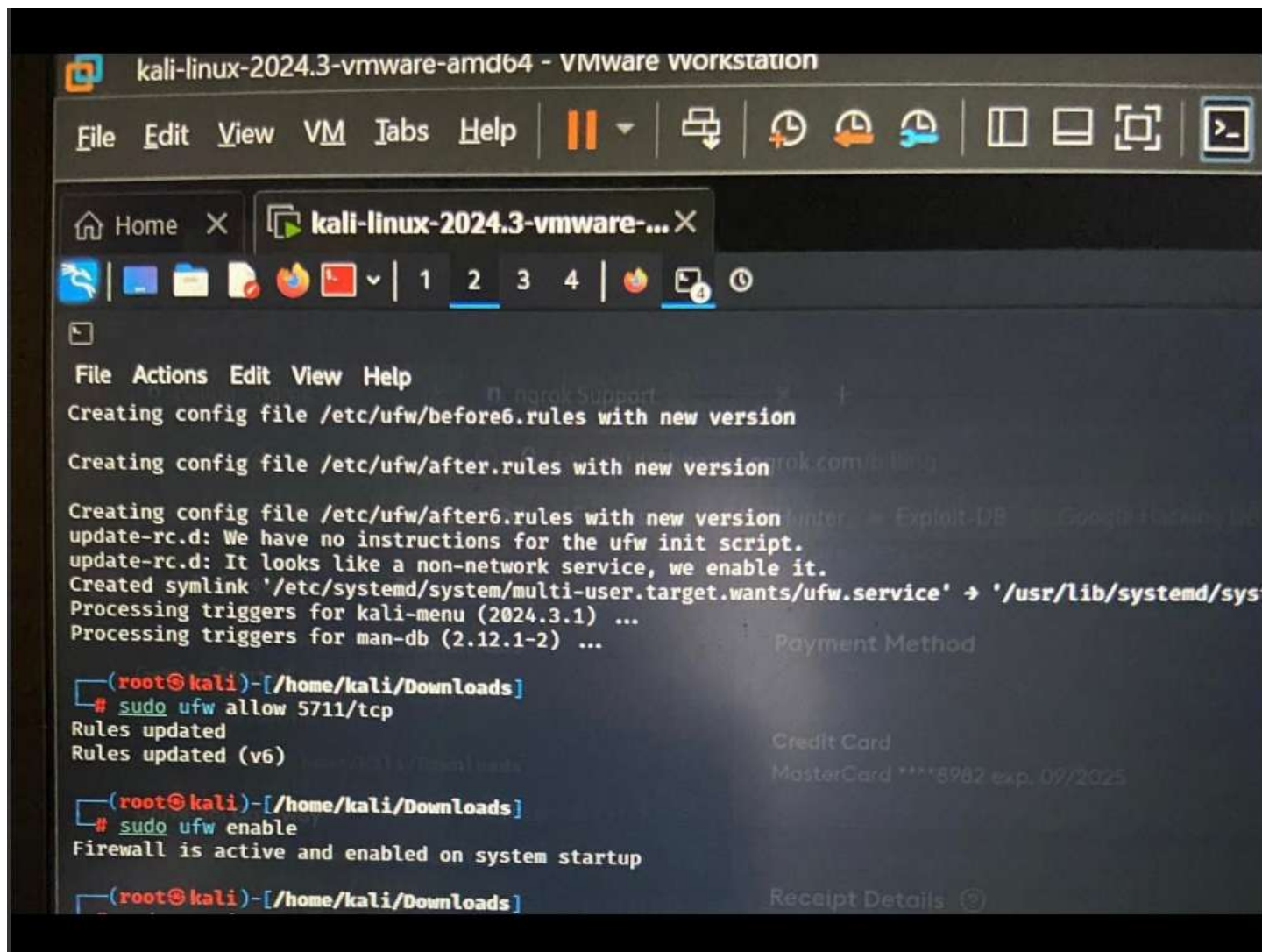
#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication no
```

At the bottom of the terminal, a message reads: "To return to your computer, move the mouse pointer outside or press Ctrl+Alt." The Windows taskbar is visible at the bottom of the screen.

## STEP 2: Firewall Configuration

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



```
kali-linux-2024.3-vmware-amd64 - VMware workstation
File Edit View VM Tabs Help
Home X kali-linux-2024.3-vmware-... X
1 2 3 4
File Actions Edit View Help
Creating config file /etc/ufw/before6.rules with new version
Creating config file /etc/ufw/after.rules with new version
Creating config file /etc/ufw/after6.rules with new version
update-rc.d: We have no instructions for the ufw init script.
update-rc.d: It looks like a non-network service, we enable it.
Created symlink '/etc/systemd/system/multi-user.target.wants/ufw.service' -> '/usr/lib/systemd/sy
Processing triggers for kali-menu (2024.3.1) ...
Processing triggers for man-db (2.12.1-2) ...

(root@kali)-[/home/kali/Downloads]
# sudo ufw allow 5711/tcp
Rules updated
Rules updated (v6)

(root@kali)-[/home/kali/Downloads]
# sudo ufw enable
Firewall is active and enabled on system startup

(root@kali)-[/home/kali/Downloads]
```

## 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
```

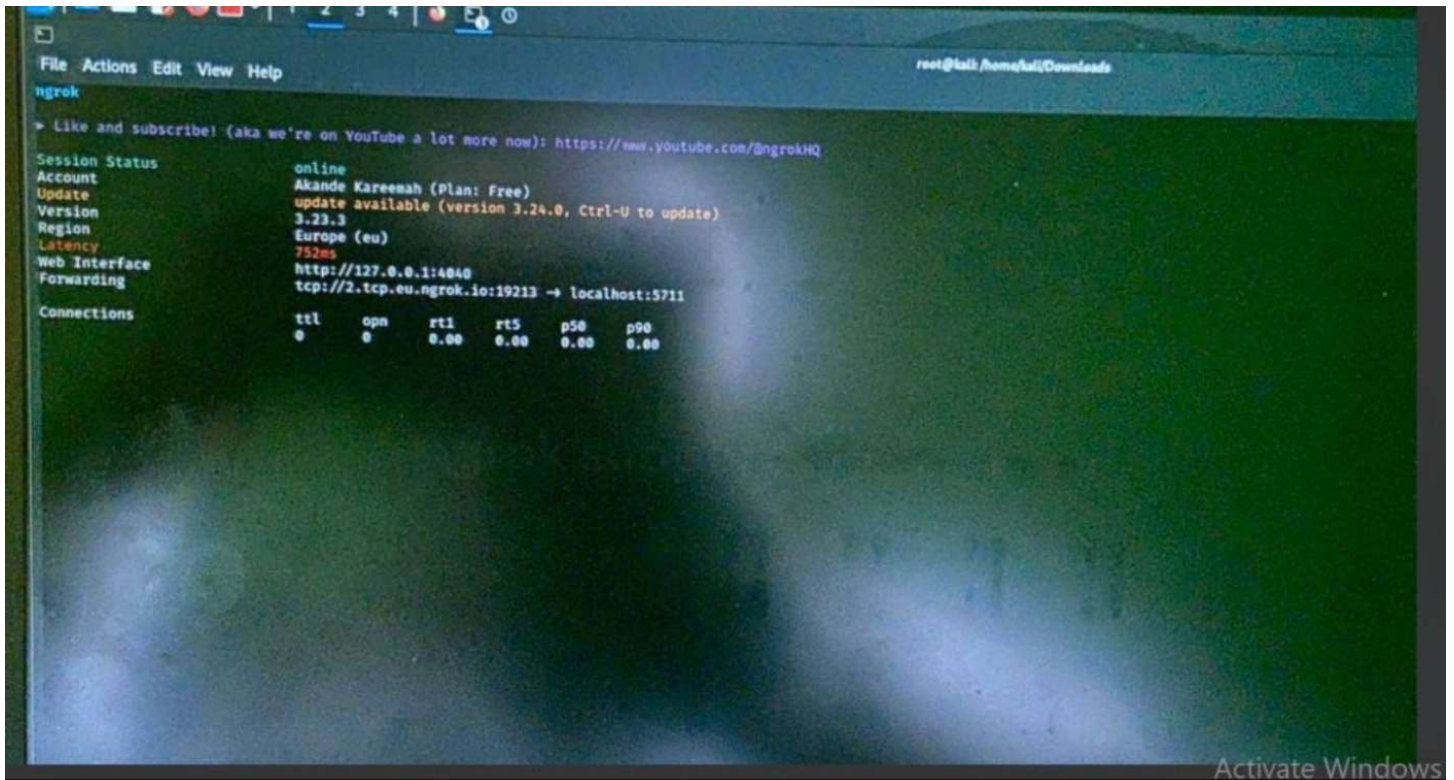
```
Installing:
fail2ban
```

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



#### STEP 4:

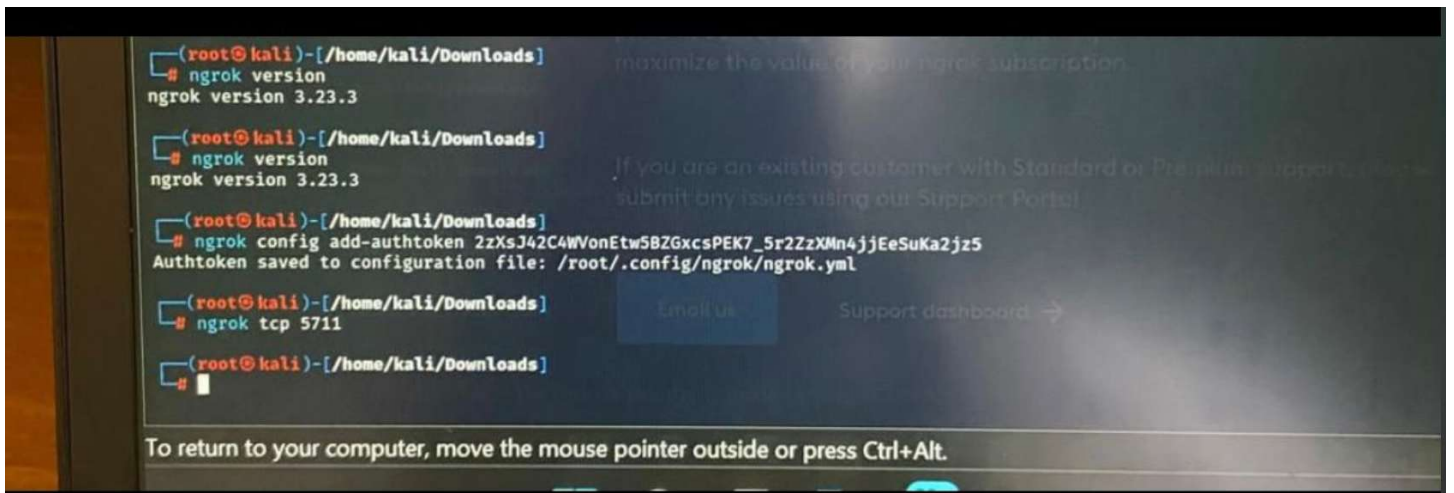
- Attempted direct port forwarding but discovered ISP blocks inbound connections.
- Implemented **Ngrok TCP tunneling** as a secure workaround, enabling external SSH connections despite ISP restrictions.



```
File Actions Edit View Help
ngrok
> Like and subscribe! (aka we're on YouTube a lot more now): https://www.youtube.com/@ngrokHQ

Session Status      online
Account             Akande Kareemah (Plan: Free)
Update              update available (version 3.24.0, Ctrl-U to update)
Version             3.23.3
Region              Europe (eu)
Latency             752ms
Web Interface        http://127.0.0.1:4040
Forwarding           tcp://2.tcp.eu.ngrok.io:19213 -> localhost:5711

Connections
  ttl  opn  rt1  rt5  p50  p90
   0    0    0.00 0.00 0.00 0.00
```



```
(root@kali)-[/home/kali/Downloads]
# ngrok version
ngrok version 3.23.3

(root@kali)-[/home/kali/Downloads]
# ngrok version
ngrok version 3.23.3

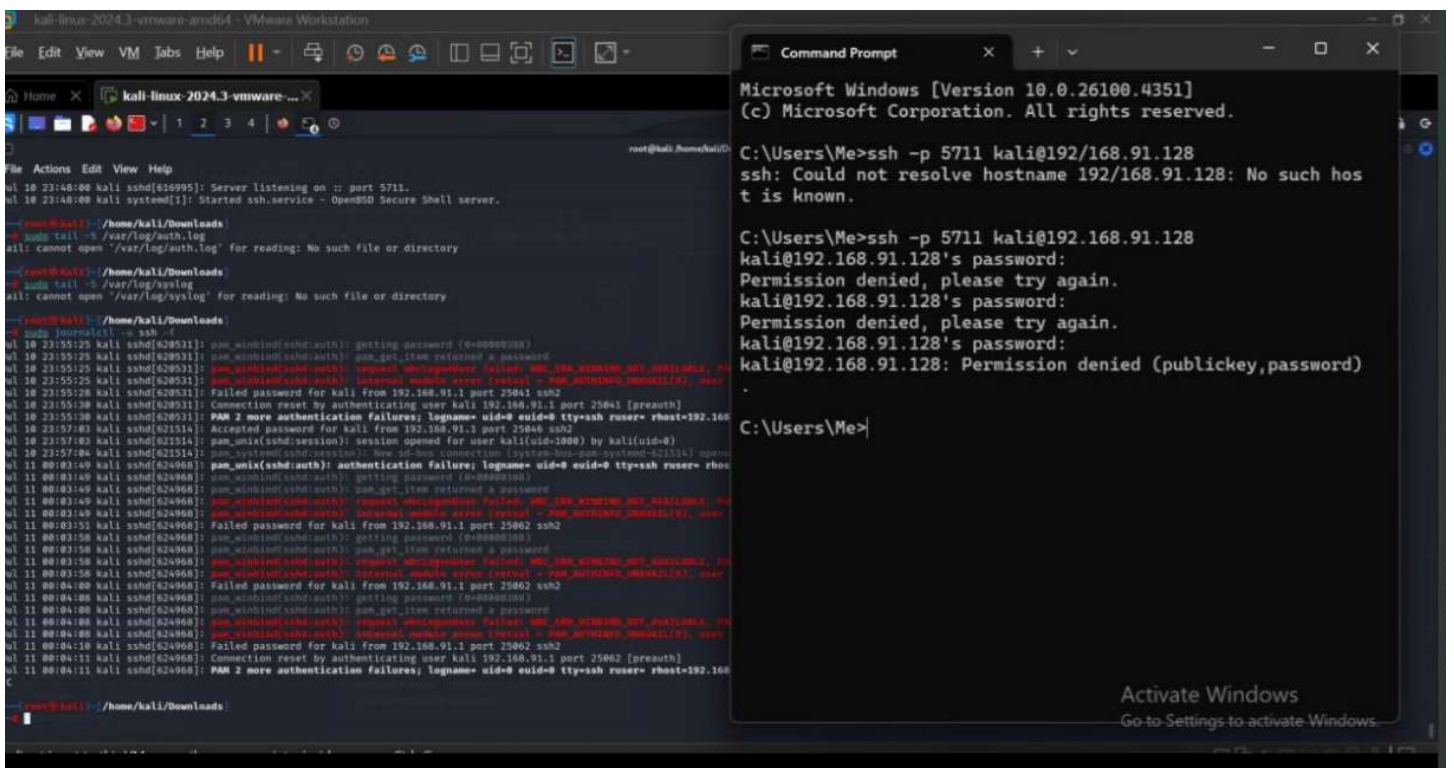
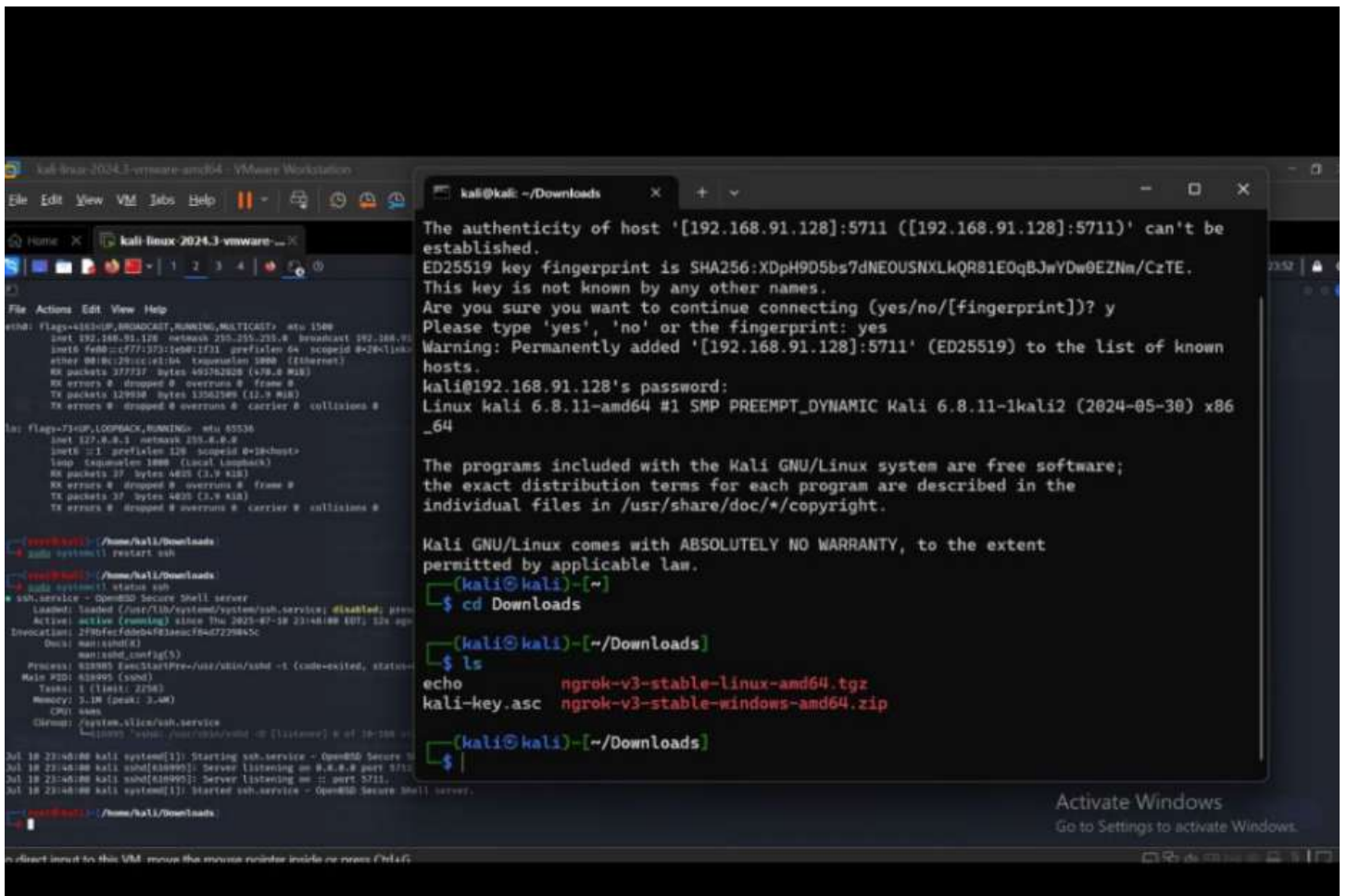
(root@kali)-[/home/kali/Downloads]
# ngrok config add-authtoken 2zXsJ42C4WVonEtw5BZGxcsPEK7_5r2ZzXMn4jjEeSuKa2jz5
Authtoken saved to configuration file: /root/.config/ngrok/ngrok.yml

(root@kali)-[/home/kali/Downloads]
# ngrok tcp 5711

(root@kali)-[/home/kali/Downloads]
#
```

#### STEP 5:

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



## 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.