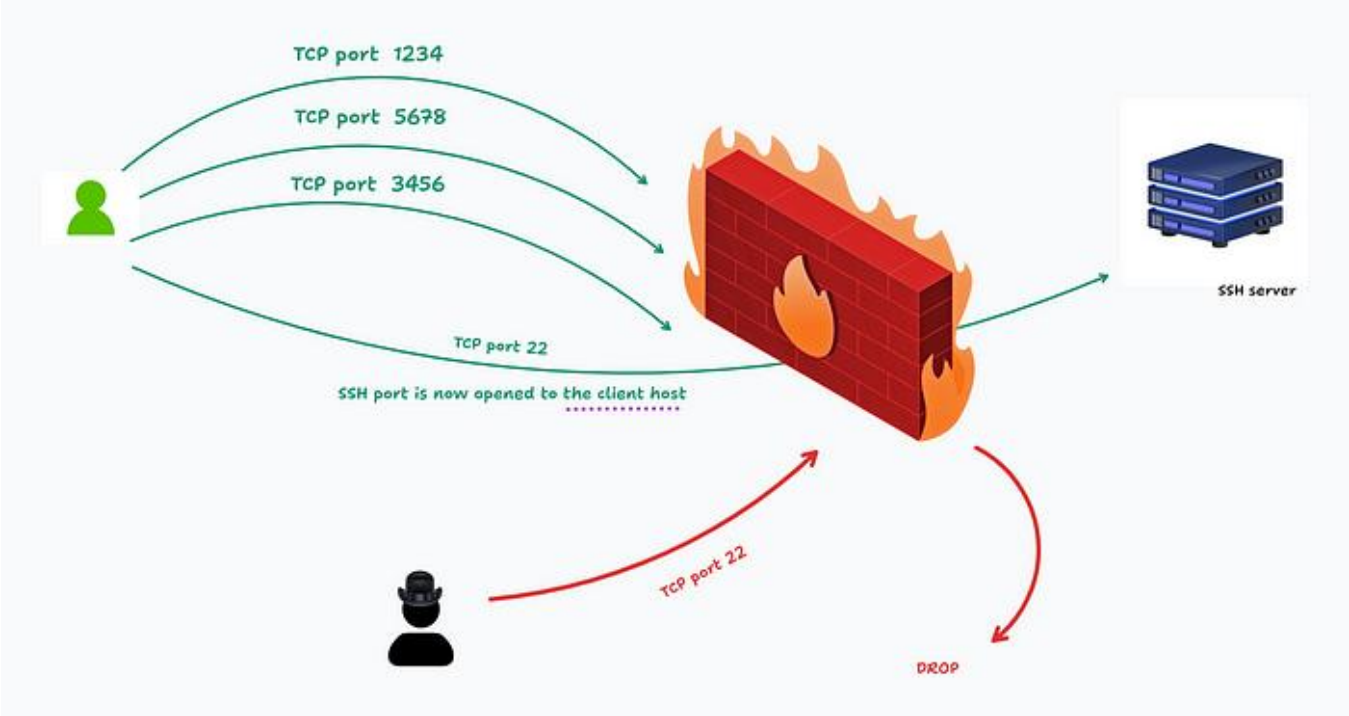
Port knocking authentication system

Abstract

This project demonstrates a lightweight and secure method for protecting SSH services using a port knocking mechanism. By utilizing knockd for monitoring network traffic and iptables with netfilter-persistent for dynamic firewall control, the system restricts SSH access unless a specific port sequence is correctly received. The setup enhances server security by keeping ports invisible to unauthorized users.

Introduction

- **Problem Statement**: SSH brute-force attacks are common. Keeping SSH ports open poses a security risk.
- **Objective**: To implement a secure, hidden authentication mechanism using port knocking to control access to the SSH service.
- **Scope**: Linux-based server setup (Kali Linux) with client-side access (Windows/Any OS).



Components

knockd: Listens for knock sequences on specified ports.

iptables: Manages access control by modifying firewall rules.

netfilter-persistent: Saves and restores iptables rules across reboots.

Flow Diagram

Client ----> Knock Sequence ---> [knockd (Server)]
|
Valid sequence received
↓
[iptables opens SSH port temporarily]
↓
Client connects via SSH
↓
knockd auto-closes port after timeout

Tools & Technologies Used

- **Operating System**: Kali Linux (Server), Windows (Client)
- **knockd**: Daemon to listen for port knock sequences
- **iptables**: Linux firewall for controlling access
- **netfilter-persistent**: To save firewall rules
- **SSH**: Secure Shell service

Installation & Setup

1. Installing knockd

```
root@debian:/home/andrew# apt-get install knockd
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libdnnl3 libflac12t64 libmsgraph-0-1 libopenh264-7 libxnnpack0
Use 'apt autoremove' to remove them.
The following NEW packages will be installed:
  knockd
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 30.0 kB of archives.
After this operation, 113 kB of additional disk space will be used.
Get:1 http://deb.debian.org/debian trixie/main amd64 knockd amd64 0.8-2+b6 [30.0
 kB]
Fetched 30.0 kB in 0s (128 kB/s)
Selecting previously unselected package knockd.
(Reading database ... 134018 files and directories currently installed.)
Preparing to unpack .../knockd_0.8-2+b6_amd64.deb ...
Unpacking knockd (0.8-2+b6) ...
Setting up knockd (0.8-2+b6) ...
Processing triggers for man-db (2.13.0-1) ...
```

2. Installing iptables-persistent

```
root@debian:/home/andrew# apt-get install iptables-persistent
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libdnnl3 libflac12t64 libmsgraph-0-1 libopenh264-7 libxnnpack0
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  iptables libip4tc2 libip6tc2 netfilter-persistent
Suggested packages:
  firewalld
The following NEW packages will be installed:
  iptables iptables-persistent libip4tc2 libip6tc2 netfilter-persistent
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 420 kB of archives.
After this operation, 2,695 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

3. Edit iptables to reject tcp on port 22

```
root@debian:/home/andrew# sudo iptables -A INPUT -p tcp --dport 22 -j REJECT
root@debian:/home/andrew#
```

4. Start the netfilter-persistent

```
root@debian:/home/andrew# sudo systemctl start netfilter-persistent
root@debian:/home/andrew# sudo netfilter-persistent save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables save
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables save
root@debian:/home/andrew# sudo netfilter-persistent reload
run-parts: executing /usr/share/netfilter-persistent/plugins.d/15-ip4tables star
t
run-parts: executing /usr/share/netfilter-persistent/plugins.d/25-ip6tables star
t
```

5. Edit /etc/knockd.conf (configuration file of knockd)

```
[options]
        UseSyslog

[openSSH]
        sequence    = 7000,8000,9000
        seq_timeout = 5
        command     = /sbin/iptables -I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn

[closeSSH]
        sequence    = 9000,8000,7000
        seq_timeout = 5
        command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn

[openHTTPS]
        sequence    = 12345,54321,24680,13579
        seq_timeout = 5
        command     = /usr/local/sbin/knock_add -i -c INPUT -p tcp -d 443 -f %IP%
        tcpflags    = syn

~
~
-- INSERT --                                                    7,33-40       All
```

6. Edit /etc/default/knockd

```
# control if we start knockd at init or not
# 1 = start
# anything else = don't start
# PLEASE EDIT /etc/knockd.conf BEFORE ENABLING
START_KNOCKD=1

# command line options
KNOCKD_OPTS="-i eth1"
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --                                                    8,2           All
```

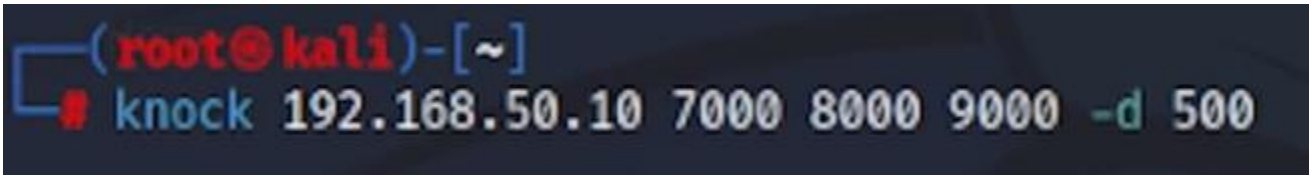
7. Start knockd service

```
root@debian:/home/andrew# systemctl start knockd
root@debian:/home/andrew# systemctl status knockd
● knockd.service - Port-Knock Daemon
     Loaded: loaded (/usr/lib/systemd/system/knockd.service; disabled; preset: enabled)
     Active: active (running) since Thu 2025-04-10 22:22:25 UTC; 5s ago
 Invocation: 71c6f88eae874d578e0ccfd69c07803e
       Docs: man:knockd(1)
   Main PID: 3736 (knockd)
      Tasks: 1 (limit: 2281)
     Memory: 864K (peak: 1.7M)
        CPU: 25ms
     CGroup: /system.slice/knockd.service
             └─3736 /usr/sbin/knockd -i enp0s3

Apr 10 22:22:25 debian systemd[1]: Started knockd.service - Port-Knock Daemon.
Apr 10 22:22:25 debian knockd[3736]: starting up, listening on enp0s3
root@debian:/home/andrew# systemctl enable knockd
Synchronizing state of knockd.service with SysV service script with /usr/lib/systemd/systemd-sy
sv-install.
Executing: /usr/lib/systemd/systemd-sysv-install enable knockd
Created symlink '/etc/systemd/system/multi-user.target.wants/knockd.service' → '/usr/lib/system
d/system/knockd.service'.
```

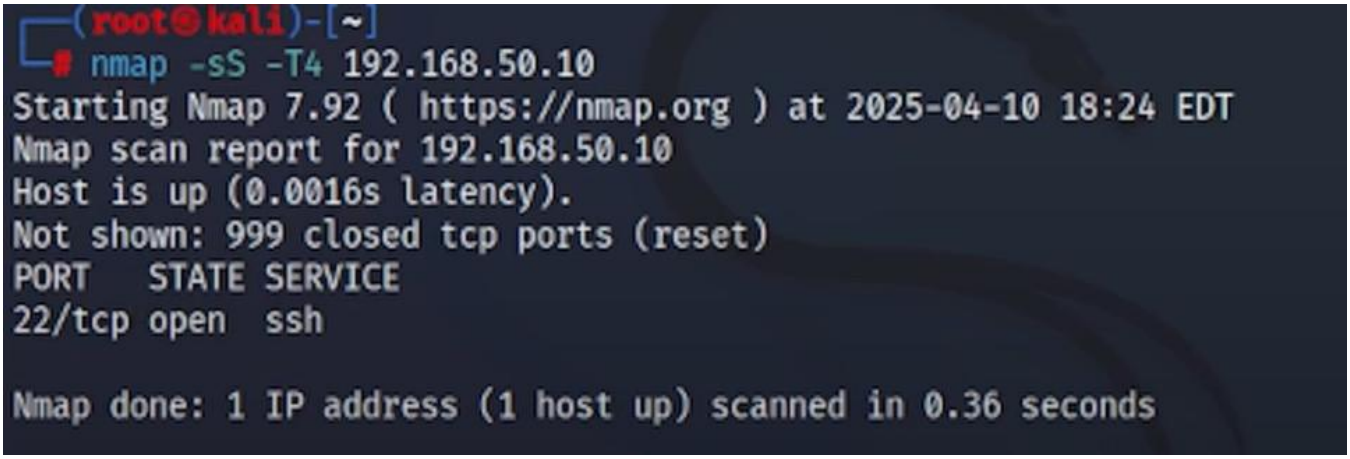
8. Now the server is ready to receive the connections , lets perform the connection from another machine. First, lets run a port scan on the server (ip : 192.168.50.10)

```
┌──(root㉿kali)-[~]
└─# nmap -sS -T4 192.168.50.10
Starting Nmap 7.92 ( https://nmap.org ) at 2025-04-10 18:23 EDT
Nmap scan report for 192.168.50.10
Host is up (0.0021s latency).
Not shown: 999 closed tcp ports (reset)
PORT   STATE    SERVICE
22/tcp filtered ssh

Nmap done: 1 IP address (1 host up) scanned in 0.51 seconds
```

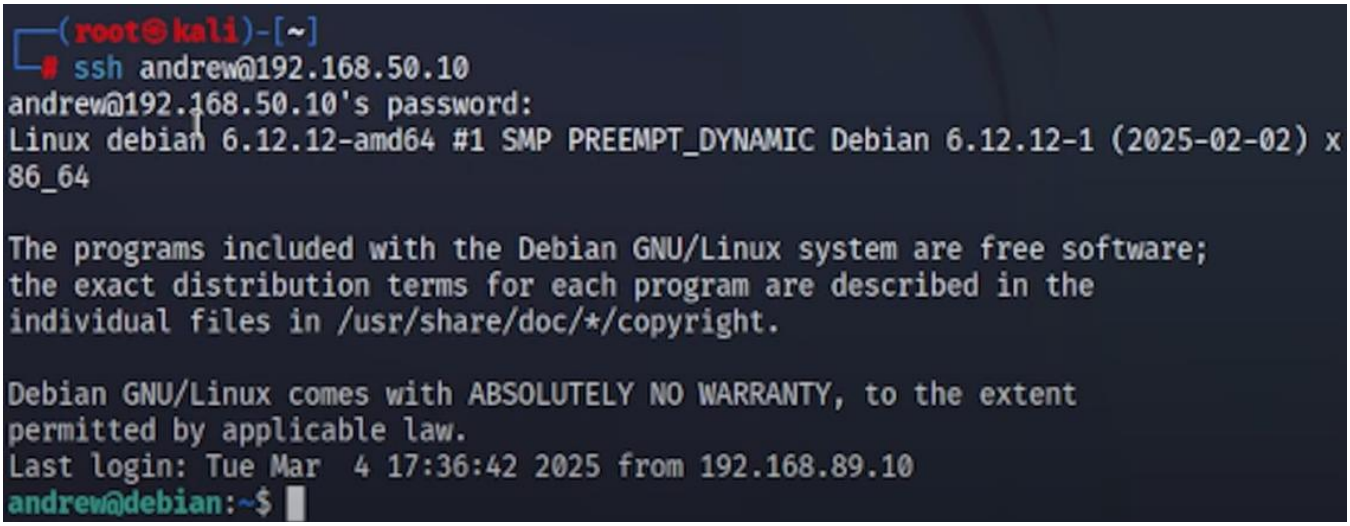
9. Giving the knocks to the server (192.168.50.10) to open the port with the sequence 7000 8000 9000 (these are ports which are closed)



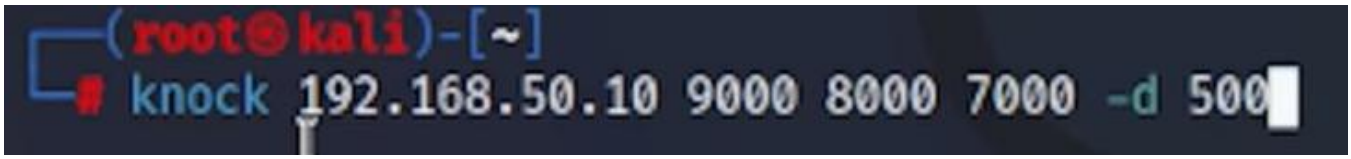
10. Testing the SSH state of the server again.

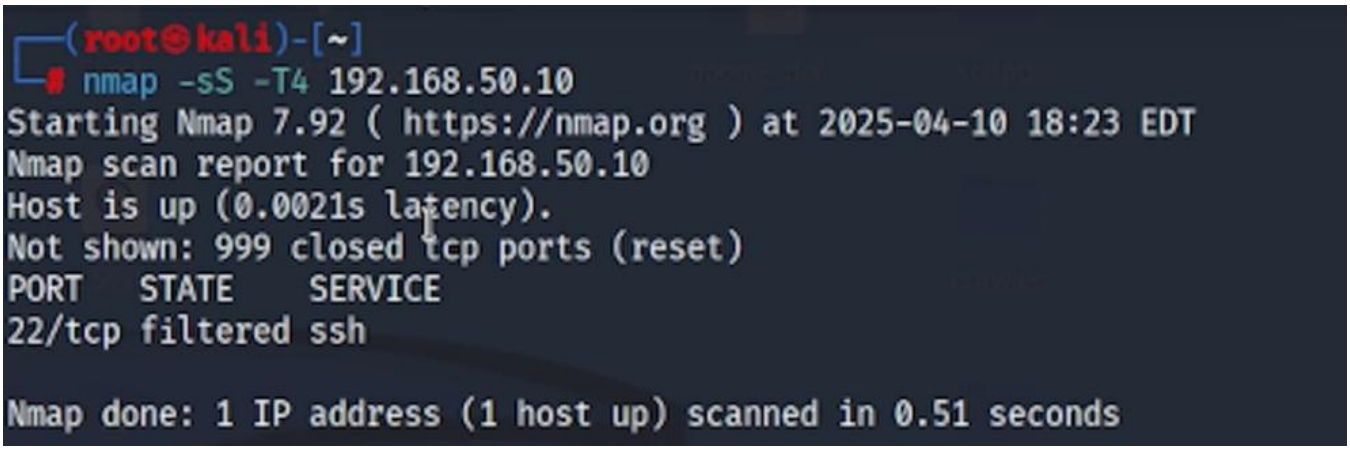


11. Establishing the SSH connection



12. Now the connection has been established successfully. Next, to close the port 22 we have to knock the ports again in reverse order which will close the port 22.





Conclusion

This project demonstrates how legacy Linux tools can be combined to implement an effective security layer. Port knocking is simple yet powerful and adds a form of **security through obscurity**. It is suitable for securing private servers or admin-only services.

Submitted By:

1. Rakesh R K
PES2UG23CS464
B.tech CSE
2nd year 4th sem
PES University (Electronic City Campus)
Campus)

2. Rahul Thushar
PES2UG23CS463
B.tech CSE
2nd year 4th sem
PES University (Electronic City