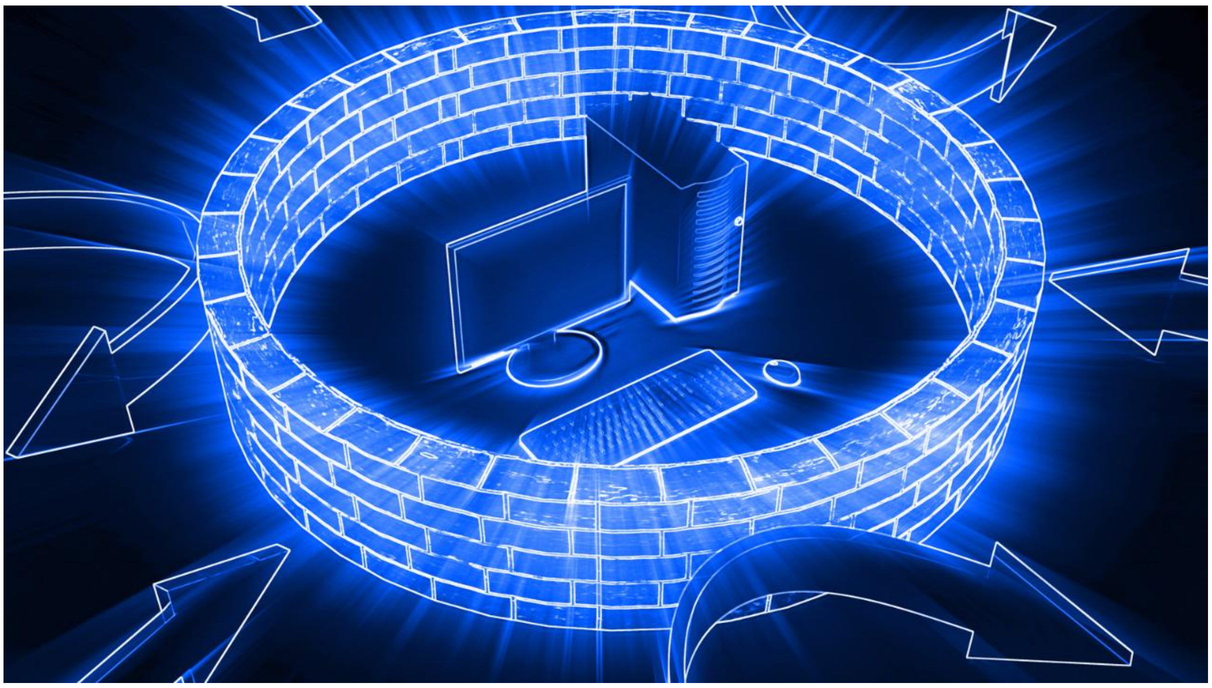
Building a Firewall Using iptables for Network Security



AUTHOR: MD UMER THOUFIQ

Table of Contents

1. Novelty of work

The following is an executive description of the Firewall Implementation and Testing project performed using iptables on a Linux-based system. The purpose of this project was to enhance the system's network security by implementing a custom firewall.For this project,let’s focus on custom rules and logging as the novelty.

1.1 Introduction

This project demonstrates the implementation of a firewall using iptables to enhance network security. The firewall is configured to block all incoming traffic by default and allows only specific services such as SSH, HTTP, and HTTPS.

1.2 Objectives

- Implement a custom firewall to block unauthorized traffic and allow essential services.
- Introduce logging to monitor dropped packets for auditing and security analysis.
- Test and verify the functionality of the firewall in a practical environment.

1.3 Tools and Technologies

- **iptables:** Core tool for firewall implementation.
- **journalctl:** For real-time monitoring and analysis of logged traffic.
- **Linux OS:** Ubuntu/Debian-based system for testing.

2. Firewall Implementation

2.1 Initial Configuration

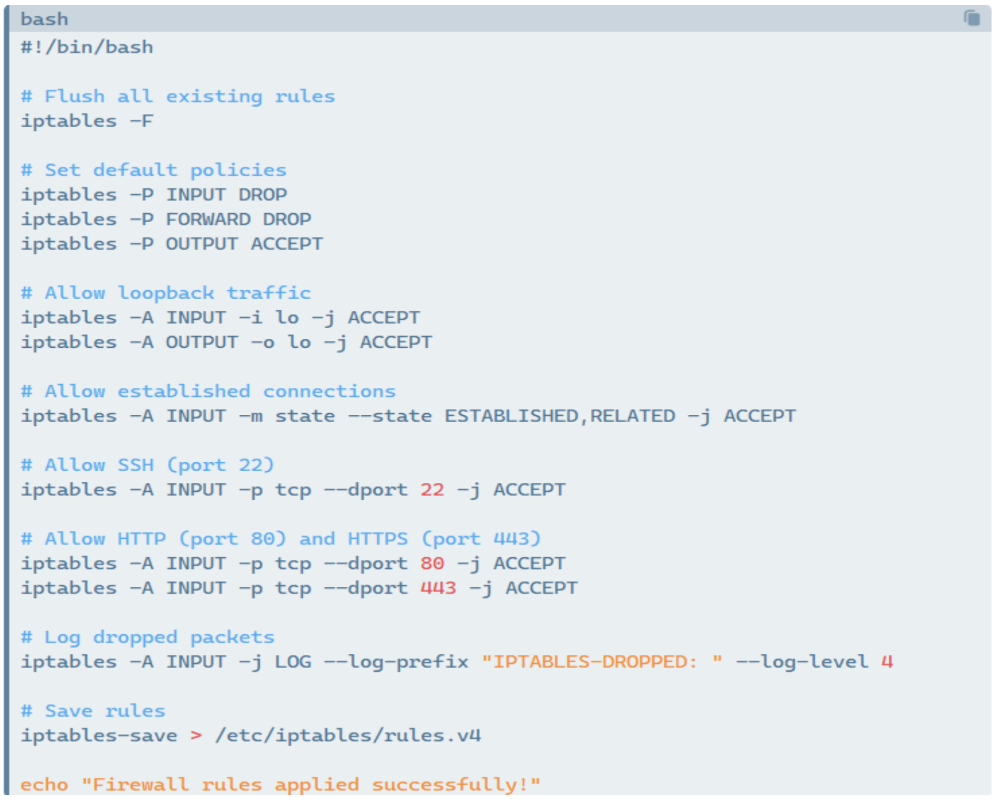
The initial setup involved flushing existing firewall rules and configuring default policies:

- All incoming and forwarding traffic was set to **DROP**.
- Outgoing traffic was allowed for seamless communication.

iptables -F # Flush all existing rules
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

2.2 Code

Code:iptables Firewall Script



```bash
#!/bin/bash

# Flush all existing rules
iptables -F

# Set default policies
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Allow loopback traffic
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Allow established connections
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Allow SSH (port 22)
iptables -A INPUT -p tcp --dport 22 -j ACCEPT

# Allow HTTP (port 80) and HTTPS (port 443)
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# Log dropped packets
iptables -A INPUT -j LOG --log-prefix "IPTABLES-DROPPED: " --log-level 4

# Save rules
iptables-save > /etc/iptables/rules.v4

echo "Firewall rules applied successfully!"
```

Fig:firewall.sh

Specific rules were added to allow only the following services:

- **Loopback Traffic:** Ensures internal system communication.
- **SSH (Port 22):** For remote administration.
- **HTTP (Port 80) and HTTPS (Port 443):** For web traffic.

2.3 Steps to Implement

1. Install iptables:

sudo apt-get update

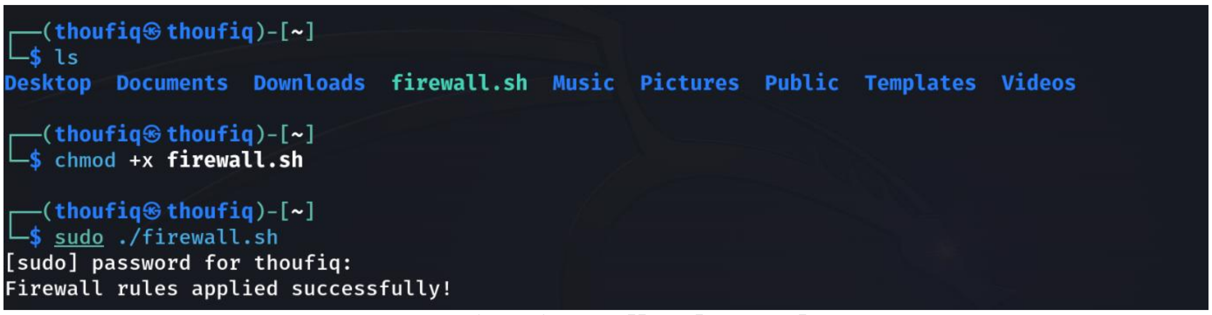
sudo apt-get install iptables

2. Save the provided script to a file (e.g., firewall.sh).
3. Make the script executable:

chmod +x firewall.sh

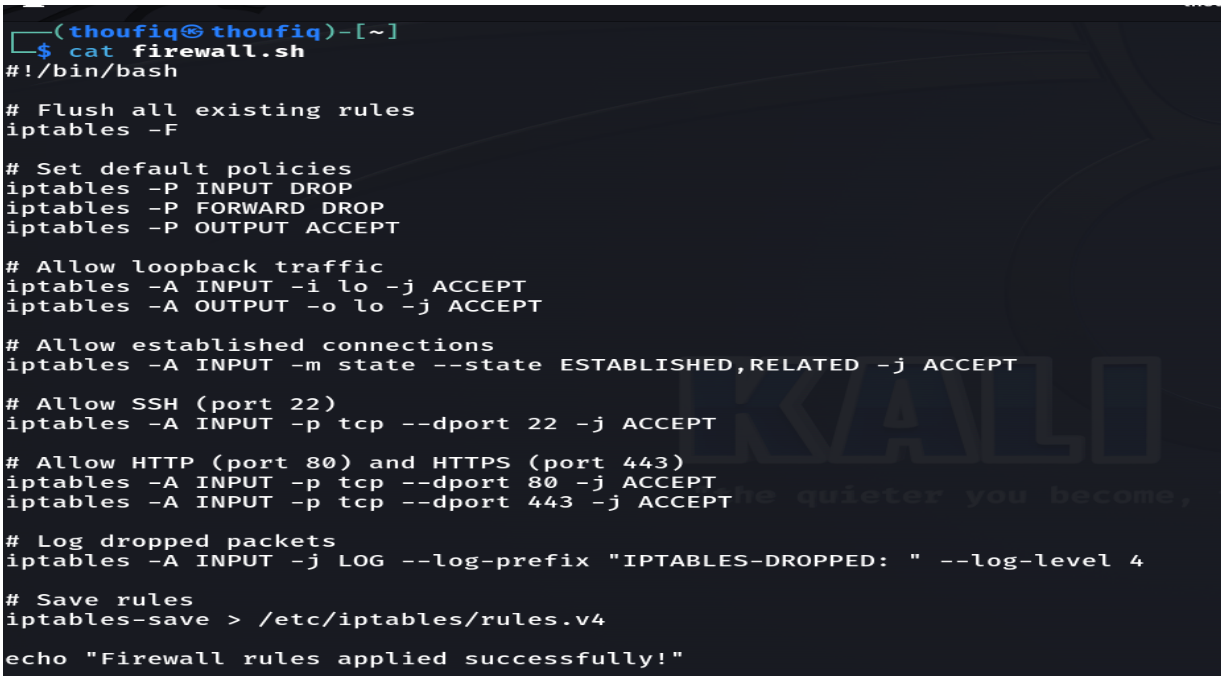
4. Run the script as root:

Sudo ./firewall.sh



```
┌──(thoufiq㉿thoufiq)-[~]
└─$ ls
Desktop  Documents  Downloads  firewall.sh  Music  Pictures  Public  Templates  Videos

┌──(thoufiq㉿thoufiq)-[~]
└─$ chmod +x firewall.sh

┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo ./firewall.sh
[sudo] password for thoufiq:
Firewall rules applied successfully!
```

Fig:Firewall rules update



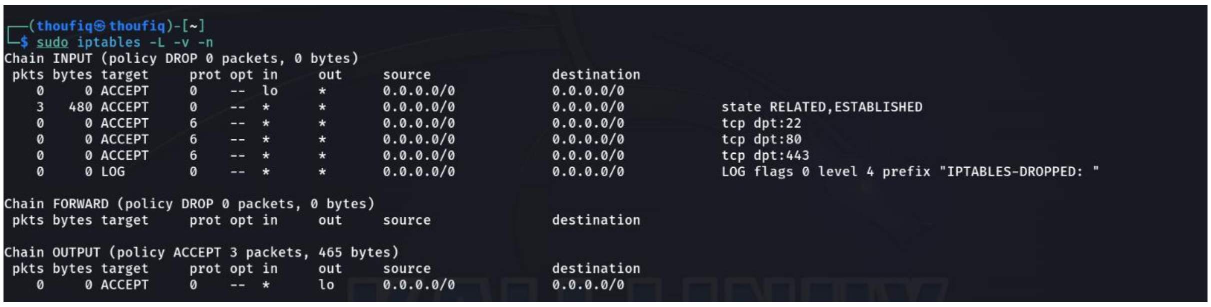
```
┌──(thoufiq㉿thoufiq)-[~]
└─$ cat firewall.sh
#!/bin/bash

# Flush all existing rules
iptables -F

# Set default policies
iptables -P INPUT DROP
iptables -P FORWARD DROP
iptables -P OUTPUT ACCEPT

# Allow loopback traffic
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# Allow established connections
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Allow SSH (port 22)
iptables -A INPUT -p tcp --dport 22 -j ACCEPT

# Allow HTTP (port 80) and HTTPS (port 443)
iptables -A INPUT -p tcp --dport 80 -j ACCEPT
iptables -A INPUT -p tcp --dport 443 -j ACCEPT

# Log dropped packets
iptables -A INPUT -j LOG --log-prefix "IPTABLES-DROPPED: " --log-level 4

# Save rules
iptables-save > /etc/iptables/rules.v4

echo "Firewall rules applied successfully!"
```

Fig:Inside the .sh file

3. Testing and Results

Iptables: After executing firewall.sh the rules are configured successfully. Verify it using the below command:

Sudo iptables -L -v -n



```
┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo iptables -L -v -n
Chain INPUT (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  lo     *       0.0.0.0/0            0.0.0.0/0
    3   480 ACCEPT     0    --  *      *       0.0.0.0/0            0.0.0.0/0            state RELATED,ESTABLISHED
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:22
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:80
    0     0 ACCEPT     6    --  *      *       0.0.0.0/0            0.0.0.0/0            tcp dpt:443
    0     0 LOG        0    --  *      *       0.0.0.0/0            0.0.0.0/0            LOG flags 0 level 4 prefix "IPTABLES-DROPPED: "

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 3 packets, 465 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     0    --  *      lo      0.0.0.0/0            0.0.0.0/0
```

Fig:Configured firewall rules

3.1 Allowed Traffic Verification

Tests were conducted to ensure the following ports were accessible:

Port 22 (SSH): Let’s try to access the same machine using ssh (22) .

ssh localhost

The command **ssh localhost** is used to establish a secure shell (SSH) connection to your own machine (localhost). It allows you to log in to the same system you're currently using, often for testing SSH configurations or practicing SSH commands.

We can also access the same system using other machine the command is

ssh <machine ip>

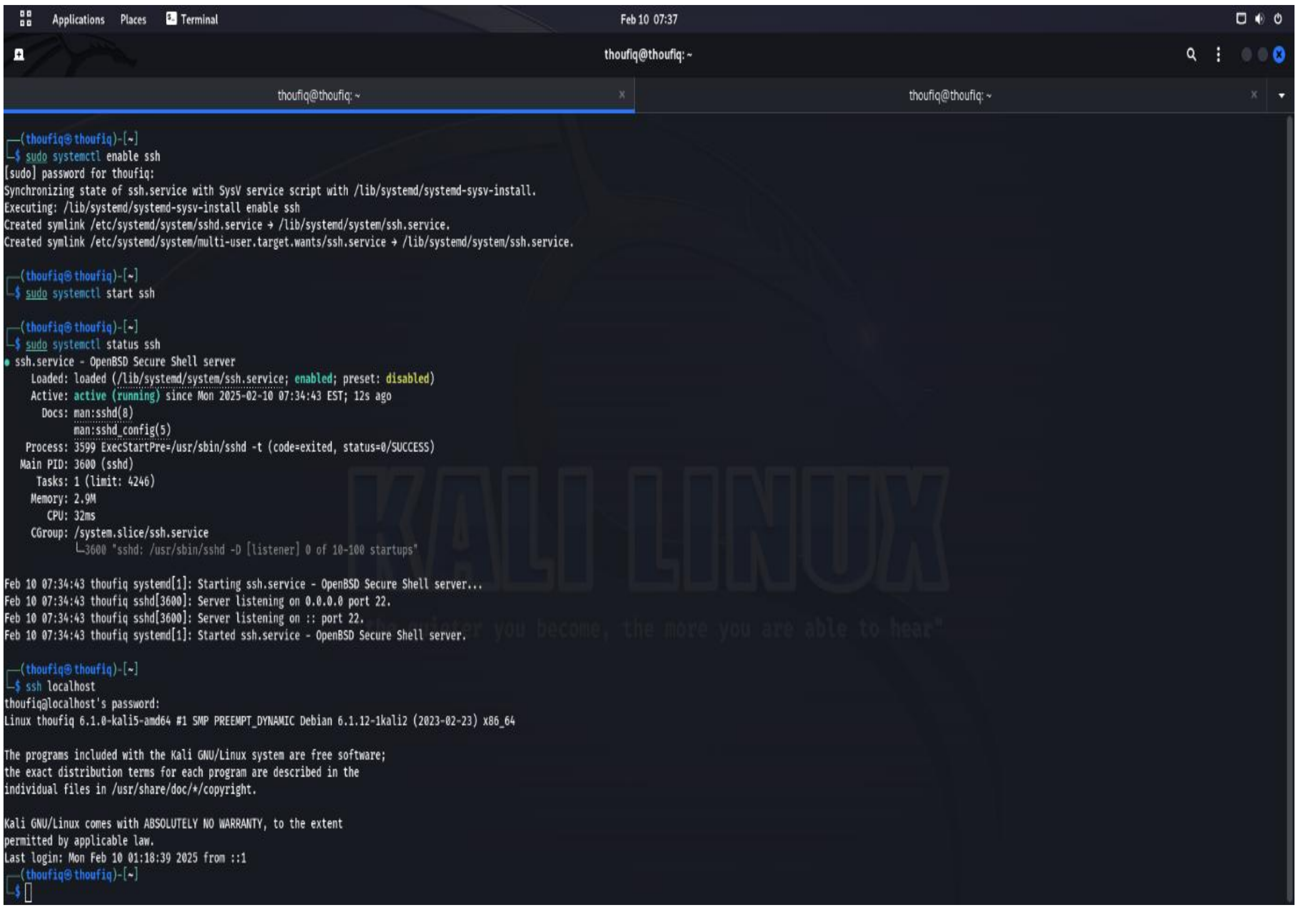
We will get access to the system , as firewall is configured in such a way that it can allow ssh (22) traffic . Below are the images that justifies the traffic verification on ssh (22)

Note:To start ssh service use:

sudo systemctl start ssh

To check ssh status :

sudo systemctl status ssh



```
┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo systemctl enable ssh
[sudo] password for thoufiq:
Synchronizing state of ssh.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable ssh
Created symlink /etc/systemd/system/sshd.service → /lib/systemd/system/ssh.service.
Created symlink /etc/systemd/system/multi-user.target.wants/ssh.service → /lib/systemd/system/ssh.service.

┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo systemctl start ssh

┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; preset: disabled)
     Active: active (running) since Mon 2025-02-10 07:34:43 EST; 12s ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 3599 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 3600 (sshd)
      Tasks: 1 (limit: 4246)
     Memory: 2.9M
        CPU: 32ms
     CGroup: /system.slice/ssh.service
             └─3600 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Feb 10 07:34:43 thoufiq systemd[1]: Starting ssh.service - OpenBSD Secure Shell server...
Feb 10 07:34:43 thoufiq sshd[3600]: Server listening on 0.0.0.0 port 22.
Feb 10 07:34:43 thoufiq sshd[3600]: Server listening on :: port 22.
Feb 10 07:34:43 thoufiq systemd[1]: Started ssh.service - OpenBSD Secure Shell server.

┌──(thoufiq㉿thoufiq)-[~]
└─$ ssh localhost
thoufiq@localhost's password:
Linux thoufiq 6.1.0-kali5-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.12-1kali2 (2023-02-23) 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.
Last login: Mon Feb 10 01:18:39 2025 from ::1
┌──(thoufiq㉿thoufiq)-[~]
└─$ 
```

Fig:Testing ssh (22)

Ports 80/443 (HTTP/HTTPS):

Verified using a web browser to access a test web server.

Let's host the local host website to check , traffic is allowed or blocked from HTTP (80)

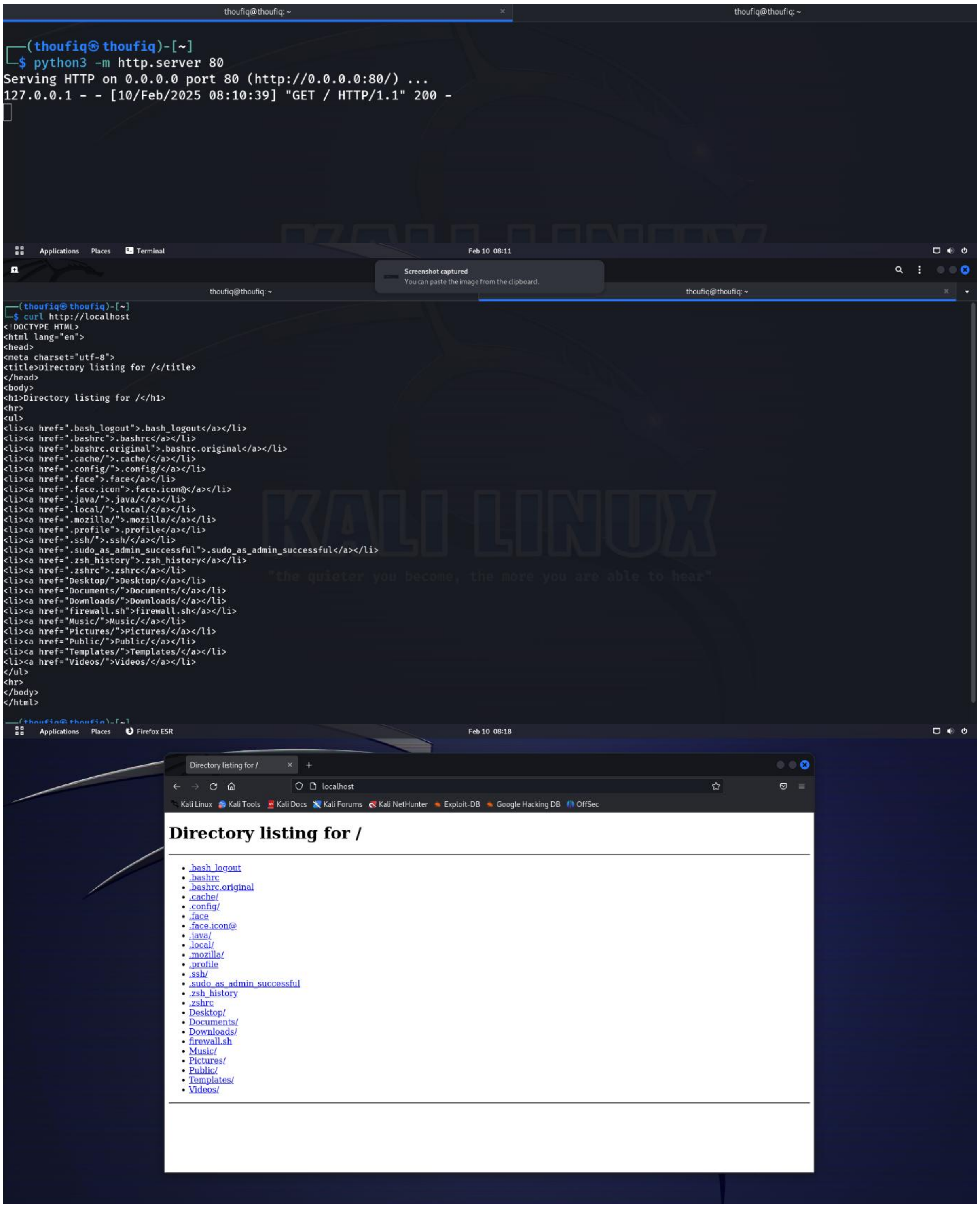
python3 -m http.server 80

This command starts a simple HTTP server using Python on port 80 (the default port for HTTP). It serves the contents of the current directory, allowing you to access files via a web browser or HTTP client.

curl http://localhost

This command uses curl (a command-line HTTP client) to send an HTTP request to localhost (your own machine) on the default HTTP port (80).

If the machine allows the traffic , the output obtained will be html code in the terminal and in browser it will be directories. Below are the images that justifies the traffic verification on HTTP (80).



```
┌──(thoufiq㉿thoufiq)-[~]
└─$ python3 -m http.server 80
Serving HTTP on 0.0.0.0 port 80 (http://0.0.0.0:80/) ...
127.0.0.1 - - [10/Feb/2025 08:10:39] "GET / HTTP/1.1" 200 -
```

```
┌──(thoufiq㉿thoufiq)-[~]
└─$ curl http://localhost
<!DOCTYPE HTML>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Directory listing for /</title>
</head>
<body>
<h1>Directory listing for /</h1>
<hr>
<ul>
<li><a href=".bash_logout">.bash_logout</a></li>
<li><a href=".bashrc">.bashrc</a></li>
<li><a href=".bashrc.original">.bashrc.original</a></li>
<li><a href=".cache/">.cache/</a></li>
<li><a href=".config/">.config/</a></li>
<li><a href=".face">.face</a></li>
<li><a href=".face.icon">.face.icon@</a></li>
<li><a href=".java/">.java/</a></li>
<li><a href=".local/">.local/</a></li>
<li><a href=".mozilla/">.mozilla/</a></li>
<li><a href=".profile">.profile</a></li>
<li><a href=".ssh/">.ssh/</a></li>
<li><a href=".sudo_as_admin_successful">.sudo_as_admin_successful</a></li>
<li><a href=".zsh_history">.zsh_history</a></li>
<li><a href=".zshrc">.zshrc</a></li>
<li><a href="Desktop/">Desktop/</a></li>
<li><a href="Documents/">Documents/</a></li>
<li><a href="Downloads/">Downloads/</a></li>
<li><a href="firewall.sh">firewall.sh</a></li>
<li><a href="Music/">Music/</a></li>
<li><a href="Pictures/">Pictures/</a></li>
<li><a href="Public/">Public/</a></li>
<li><a href="Templates/">Templates/</a></li>
<li><a href="Videos/">Videos/</a></li>
</ul>
<hr>
</body>
</html>
```

Directory listing for /

- .bash_logout
- .bashrc
- .bashrc.original
- .cache/
- .config/
- .face
- .face.icon@
- .java/
- .local/
- .mozilla/
- .profile
- .ssh/
- .sudo_as_admin_successful
- .zsh_history
- .zshrc
- Desktop/
- Documents/
- Downloads/
- firewall.sh
- Music/
- Pictures/
- Public/
- Templates/
- Videos/

Fig:Testing HTTP (80)

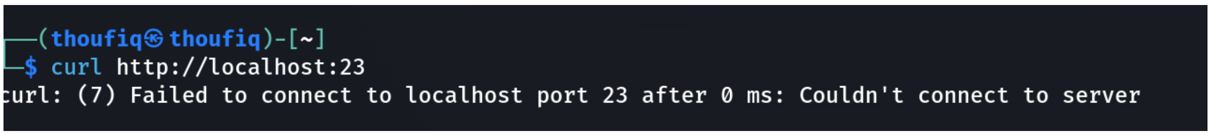
3.2 Blocked Traffic Testing

Ports such as Telnet (Port 23) were intentionally blocked and tested. Attempts to connect resulted in failure, confirming the firewall's functionality.

curl http://localhost:23

The command curl http://localhost:23 sends an HTTP request to your local machine (localhost) on port 23.

Port 23 is typically used for Telnet and is often configured to block traffic on that port.

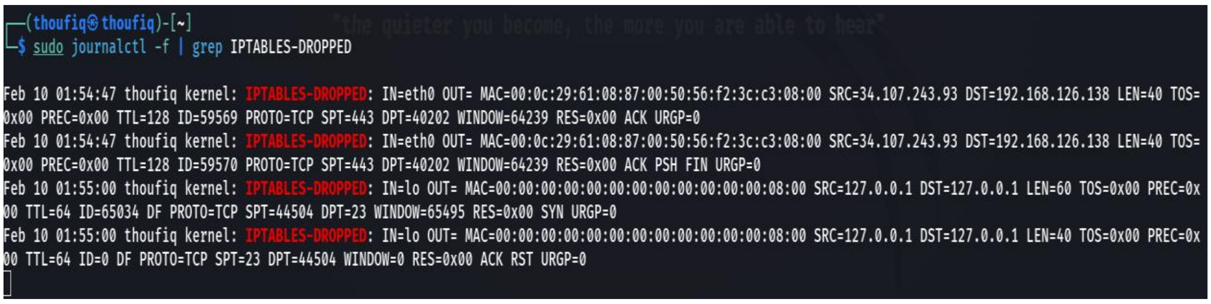


```
┌──(thoufiq㉿thoufiq)-[~]
└─$ curl http://localhost:23
curl: (7) Failed to connect to localhost port 23 after 0 ms: Couldn't connect to server
```

Fig:Blocked port 23 traffic

Logged packets were verified using the monitoring command.

sudo journalctl -f | grep IPTABLES-DOMAIN



```
┌──(thoufiq㉿thoufiq)-[~]
└─$ sudo journalctl -f | grep IPTABLES-DROPPED

Feb 10 01:54:47 thoufiq kernel: IPTABLES-DROPPED: IN=eth0 OUT= MAC=00:0c:29:61:08:87:00:50:56:f2:3c:c3:08:00 SRC=34.107.243.93 DST=192.168.126.138 LEN=40 TOS=
0x00 PREC=0x00 TTL=128 ID=59569 PROTO=TCP SPT=443 DPT=40202 WINDOW=64239 RES=0x00 ACK URGP=0
Feb 10 01:54:47 thoufiq kernel: IPTABLES-DROPPED: IN=eth0 OUT= MAC=00:0c:29:61:08:87:00:50:56:f2:3c:c3:08:00 SRC=34.107.243.93 DST=192.168.126.138 LEN=40 TOS=
0x00 PREC=0x00 TTL=128 ID=59570 PROTO=TCP SPT=443 DPT=40202 WINDOW=64239 RES=0x00 ACK PSH FIN URGP=0
Feb 10 01:55:00 thoufiq kernel: IPTABLES-DROPPED: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=127.0.0.1 DST=127.0.0.1 LEN=60 TOS=0x00 PREC=0x
00 TTL=64 ID=65034 DF PROTO=TCP SPT=44504 DPT=23 WINDOW=65495 RES=0x00 SYN URGP=0
Feb 10 01:55:00 thoufiq kernel: IPTABLES-DROPPED: IN=lo OUT= MAC=00:00:00:00:00:00:00:00:00:00:00:00:08:00 SRC=127.0.0.1 DST=127.0.0.1 LEN=40 TOS=0x00 PREC=0x
00 TTL=64 ID=0 DF PROTO=TCP SPT=23 DPT=44504 WINDOW=0 RES=0x00 ACK RST URGP=0
```

Fig:Blocked traffic form log’s

4. Conclusion

This project demonstrates the implementation of a basic yet effective firewall using iptables to enhance network security. By blocking all incoming traffic by default and allowing only essential services like SSH, HTTP, and HTTPS, we ensure a secure network environment. The addition of logging functionality provides insights into blocked traffic, aiding in monitoring and threat detection.

While this implementation is simple, it serves as a strong foundation for further enhancements, such as integrating a web interface, automating rule updates, or scaling for larger networks. Overall, this project highlights the importance of firewalls and the versatility of iptables in securing modern networks.

5. Appendices and References

- Mihalos, M. G., Nalmpantis, S. I., & Ovaliadis, K. (2019). **Design and Implementation of Firewall Security Policies using Linux Iptables**. *Journal of Engineering Science and Technology Review*, 12(1), 80-86.
- Official iptables Documentation: https://linux.die.net/man/8/iptables
- Linux Journal on iptables: https://www.linuxjournal.com/