

# **COL8585/COL862 Assignment I: The Platinum Shield**

Task 1 Report: Configuring a FreeBSD 13.4 L3 Forwarding Firewall  
using pf

**Arindam Sal** (Entry No. 2024JCS2041)  
**Sambit Paul** (Entry No. 2024JCS2038)

## **1 Overview and Problem Statement**

This report documents Task 1 of “The Platinum Shield” assignment. The task requires building a **Layer-3 (L3) forwarding firewall** using **FreeBSD 13.4** and its firewall framework **pf** (Packet Filter).

The firewall must sit between an Ubuntu Client (VM1) and an Ubuntu Server (VM3), and enforce the following traffic policy:

- **Default deny:** block all traffic by default.
- **Allow ICMP:** permit ping for diagnostics.
- **Allow HTTP:** permit TCP port 80 (web access) from client to server.
- **Block SSH:** deny TCP port 22 connectivity after firewall activation.

The assignment also requires evidence:

- Commands used to enable IPv4 forwarding (routing) on VM2.
- Firewall rules in `/etc/pf.conf`.
- Proof of success using screenshots including traceroute from VM1 to VM3.

## **2 Conceptual Background (Theory)**

### **2.1 Why Layer-3 Forwarding is Needed**

VM1 and VM3 are placed in **different subnets**:

- VM1: 10.0.1.0/24
- VM3: 10.0.2.0/24

Hosts in different subnets cannot directly communicate without a router/gateway. Therefore, VM2 must act as an L3 router:

- VM1 sends packets to VM2 (gateway) when destination is outside 10.0.1.0/24.
- VM2 forwards packets out of its other interface toward VM3.
- VM3 replies back via VM2.

This requires **IP forwarding** to be enabled on VM2, otherwise VM2 will receive packets but will not forward them.

## 2.2 Firewalling with pf (Packet Filter)

pf is FreeBSD's stateful firewall system. It processes packets based on rules that match:

- Interface (e.g., em0 or em1)
- Direction (inbound or outbound)
- Protocol (TCP/UDP/ICMP)
- Ports (e.g., 80, 22)
- Source/Destination IP

## 2.3 Default-Deny (Block All) Security Model

A default-deny policy means:

“Everything is blocked unless explicitly allowed.”

This is the standard enterprise security approach because:

- It minimizes attack surface.
- It prevents unexpected services from being reachable.
- It enforces least privilege (only required traffic passes).

## 2.4 Stateful Filtering and `keep state`

When pf uses `keep state`, it remembers a connection (e.g., a TCP session). This is important because:

- TCP uses a handshake (SYN, SYN-ACK, ACK).
- If the firewall only allows SYN packets but not reply packets, connections break.
- With state tracking, pf automatically allows return traffic for approved connections.

Thus, `keep state` is necessary for correct TCP/ICMP operation.

# 3 Network Topology and Addressing

## 3.1 Topology

- **VM1 (Ubuntu Client)**: 10.0.1.2/24 via enp0s3, Gateway 10.0.1.1
- **VM2 (FreeBSD Firewall)**:
  - em0: 10.0.1.1/24 (client side)
  - em1: 10.0.2.1/24 (server side)
- **VM3 (Ubuntu Server)**: 10.0.2.2/24 via enp0s3, Gateway 10.0.2.1

## 3.2 VirtualBox Internal Networks (Isolation)

We use VirtualBox **Internal Network** to isolate traffic from the host machine and internet. Two internal networks are created:

- **intnetA**: connects VM1  $\leftrightarrow$  VM2(em0)
- **intnetB**: connects VM2(em1)  $\leftrightarrow$  VM3

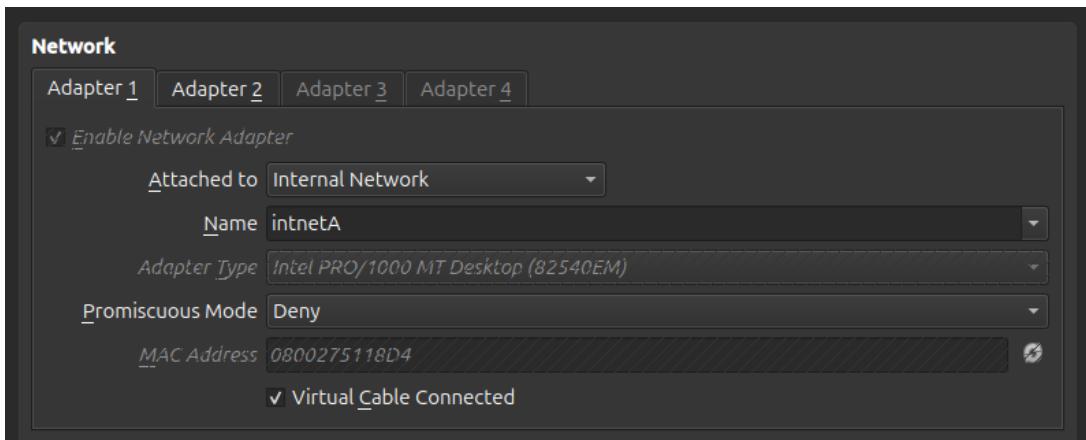


Figure 1: VM2 Firewall Adapter connected to Internal Network `intnetA`

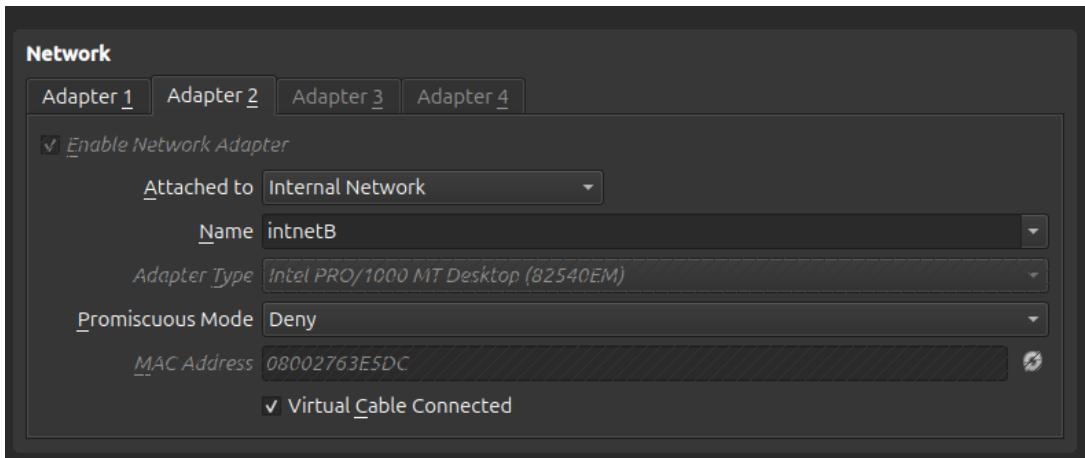


Figure 2: VM2 Firewall Adapter connected to Internal Network intnetB

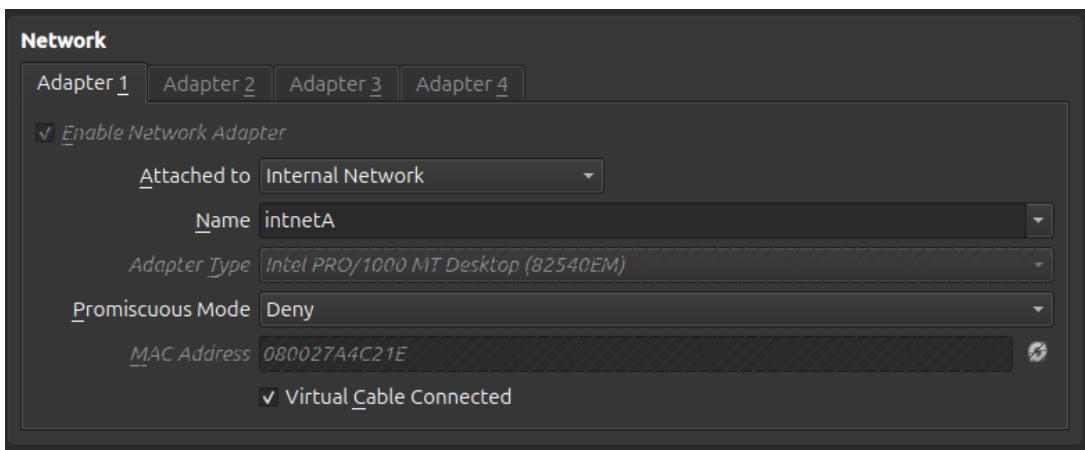


Figure 3: VM1 Client Adapter connected to Internal Network intnetA

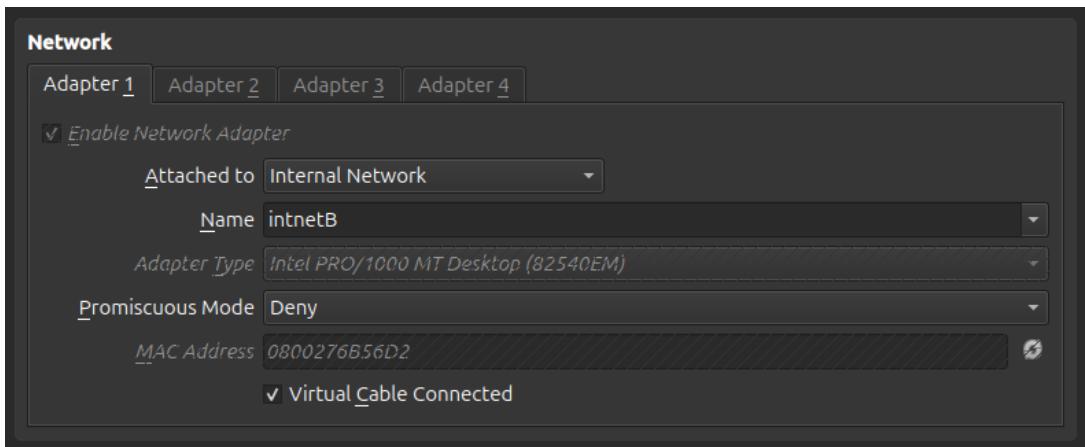


Figure 4: VM3 Server Adapter connected to Internal Network intnetB

## 4 Infrastructure Setup (Static IP and Routes)

### 4.1 Why Static IP?

Static IP ensures:

- predictable addressing (important for firewall rules)
- correct default gateway behavior
- reproducibility of experiments

### 4.2 VM1 Configuration (Ubuntu Client)

VM1 is configured with IP 10.0.1.2/24 and gateway 10.0.1.1.

```
network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 10.0.1.2/24
      routes:
        - to: default
          via: 10.0.1.1
      nameservers:
        addresses:
          - 8.8.8.8
```

Listing 1: Netplan-style configuration intent for VM1 (static IP + gateway)

#### Explanation of fields:

- **addresses**: assigns a fixed IP and prefix
- **routes**: sets default route (gateway) to the firewall
- **nameservers**: DNS server (not mandatory for internal access but standard practice)

```

student@ubuntu-client:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:a4:c2:1e brd ff:ff:ff:ff:ff:ff
        inet 10.0.1.2/24 brd 10.0.1.255 scope global enp0s3
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:fea4:c21e/64 scope link
            valid_lft forever preferred_lft forever
student@ubuntu-client:~$ ip route
default via 10.0.1.1 dev enp0s3 proto static
10.0.1.0/24 dev enp0s3 proto kernel scope link src 10.0.1.2
student@ubuntu-client:~$
```

Figure 5: VM1 verification: `ip a` and `ip route` show IP 10.0.1.2/24 and default route via 10.0.1.1

### 4.3 VM3 Configuration (Ubuntu Server)

VM3 is configured with IP 10.0.2.2/24 and gateway 10.0.2.1.

```

network:
  version: 2
  renderer: networkd
  ethernets:
    enp0s3:
      dhcp4: no
      addresses:
        - 10.0.2.2/24
      routes:
        - to: default
          via: 10.0.2.1
      nameservers:
        addresses:
          - 8.8.8.8
```

Listing 2: Netplan-style configuration intent for VM3 (static IP + gateway)

```

student@ubuntu-server:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:6b:56:d2 brd ff:ff:ff:ff:ff:ff
        inet 10.0.2.2/24 brd 10.0.2.255 scope global enp0s3
            valid_lft forever preferred_lft forever
        inet6 fe80::a00:27ff:fe6b:56d2/64 scope link
            valid_lft forever preferred_lft forever
student@ubuntu-server:~$ ip route
default via 10.0.2.1 dev enp0s3 proto static
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.2
student@ubuntu-server:~$
```

Figure 6: VM3 verification: `ip a` and `ip route` show IP 10.0.2.2/24 and default route via 10.0.2.1

## 5 Objective 1: Enable Routing (IPv4 Forwarding) on VM2

### 5.1 Why IPv4 Forwarding is Required

Even if VM2 has two interfaces, it will not act as a router unless forwarding is enabled. Without forwarding:

- VM2 receives packets on em0
- but drops them instead of forwarding to em1

### 5.2 Commands Used

```

sysctl net.inet.ip.forwarding=1
sysrc gateway_enable="YES"
```

Listing 3: Commands to enable IPv4 forwarding on FreeBSD VM2

#### Explanation:

- `sysctl net.inet.ip.forwarding=1`: enables forwarding immediately (runtime).
- `sysrc gateway_enable = "YES"` : makes forwarding persistent after reboot via `rc.conf`.

```

root@firewall:~# sysctl net.inet.ip.forwarding
net.inet.ip.forwarding: 1
root@firewall:~# grep gateway_enable /etc/rc.conf
gateway_enable="YES"
root@firewall:~# sysrc gateway_enable
gateway_enable: YES
root@firewall:~#

```

Figure 7: VM2 verification: `net.inet.ip.forwarding=1` and `gateway_enable="YES"`

## 6 Objective 2: Validate Basic L3 Connectivity (Before Firewall)

### 6.1 ICMP Testing (Ping)

Ping uses ICMP Echo Request/Reply. Allowing ping is useful because:

- It confirms that routing works.
- It helps isolate issues (IP, gateway, forwarding, firewall).

```

student@ubuntu-client:~$ ping 10.0.1.1
PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data.
64 bytes from 10.0.1.1: icmp_seq=1 ttl=64 time=1.19 ms
64 bytes from 10.0.1.1: icmp_seq=2 ttl=64 time=1.28 ms
64 bytes from 10.0.1.1: icmp_seq=3 ttl=64 time=1.13 ms
^C
--- 10.0.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.128/1.197/1.279/0.062 ms
student@ubuntu-client:~$ ping 10.0.2.1
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=64 time=0.996 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=64 time=1.15 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=64 time=1.40 ms
^C
--- 10.0.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 0.996/1.180/1.397/0.165 ms
student@ubuntu-client:~$ ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2) 56(84) bytes of data.
64 bytes from 10.0.2.2: icmp_seq=1 ttl=63 time=1.97 ms
64 bytes from 10.0.2.2: icmp_seq=2 ttl=63 time=1.87 ms
64 bytes from 10.0.2.2: icmp_seq=3 ttl=63 time=1.98 ms
64 bytes from 10.0.2.2: icmp_seq=4 ttl=63 time=1.95 ms
^C
--- 10.0.2.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 1.870/1.941/1.977/0.042 ms
student@ubuntu-client:~$

```

Figure 8: VM1 ping tests to 10.0.1.1 (firewall em0), 10.0.2.1 (firewall em1), and 10.0.2.2 (server)

```

student@ubuntu-server:~$ ping 10.0.2.1
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=64 time=1.05 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=64 time=1.16 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=64 time=1.03 ms
^C
--- 10.0.2.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 1.027/1.078/1.159/0.057 ms
student@ubuntu-server:~$ ping 10.0.1.1
PING 10.0.1.1 (10.0.1.1) 56(84) bytes of data.
64 bytes from 10.0.1.1: icmp_seq=1 ttl=64 time=1.09 ms
64 bytes from 10.0.1.1: icmp_seq=2 ttl=64 time=3.02 ms
64 bytes from 10.0.1.1: icmp_seq=3 ttl=64 time=1.21 ms
^C
--- 10.0.1.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2002ms
rtt min/avg/max/mdev = 1.094/1.773/3.017/0.880 ms
student@ubuntu-server:~$ ping 10.0.1.2
PING 10.0.1.2 (10.0.1.2) 56(84) bytes of data.
64 bytes from 10.0.1.2: icmp_seq=1 ttl=63 time=1.15 ms
64 bytes from 10.0.1.2: icmp_seq=2 ttl=63 time=1.93 ms
^C
--- 10.0.1.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 1.152/1.539/1.927/0.387 ms
student@ubuntu-server:~$
```

Figure 9: VM3 ping tests to 10.0.2.1 (firewall em1), 10.0.1.1 (firewall em0), and 10.0.1.2 (client)

## 6.2 Traceroute Requirement (Mandatory Evidence)

Traceroute helps confirm the path taken by packets. In a multi-hop path, traceroute reveals intermediate routers by using increasing TTL values.

**How it works (simple explanation):**

- TTL starts at 1. First router reduces TTL to 0 and sends ICMP Time Exceeded.
- TTL increases to 2, reaching the next hop, and so on.

```

student@ubuntu-client:~$ traceroute 10.0.2.2
traceroute to 10.0.2.2 (10.0.2.2), 30 hops max, 60 byte packets
 1  10.0.1.1 (10.0.1.1)  1.118 ms  1.025 ms  0.996 ms
 2  10.0.2.2 (10.0.2.2)  1.978 ms  1.942 ms  1.928 ms
student@ubuntu-client:~$
```

Figure 10: Traceroute from VM1 to VM3: hop 1 is firewall (10.0.1.1), hop 2 is server (10.0.2.2)

**Inference:** The traceroute output confirms VM2 is functioning as the intermediate L3 device.

## 7 Objective 3: Deploy Application Service (HTTP) on VM3

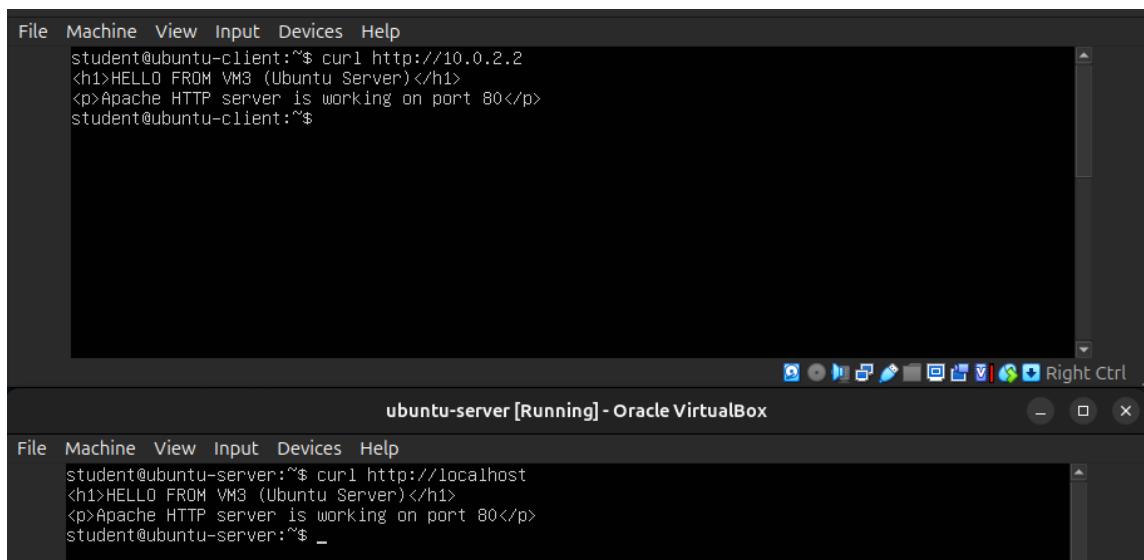
### 7.1 Why HTTP is Chosen

HTTP (port 80) is a realistic service that firewalls commonly allow for controlled access. We run Apache on VM3 so we can verify that the firewall allows port 80 traffic end-to-end.

```
student@ubuntu-server:~$ sudo systemctl status apache2
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2026-02-01 12:09:53 UTC; 32min ago
     Docs: https://httpd.apache.org/docs/2.4/
      Main PID: 1644 (apache2)
        Tasks: 55 (limit: 2218)
       Memory: 5.2M
          CPU: 170ms
        CGroup: /system.slice/apache2.service
                  ├─1644 /usr/sbin/apache2 -k start
                  ├─1646 /usr/sbin/apache2 -k start
                  └─1647 /usr/sbin/apache2 -k start

Feb 01 12:09:53 ubuntu-server systemd[1]: Starting The Apache HTTP Server...
Feb 01 12:09:53 ubuntu-server apachectl[1643]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 10.0.2.2.
Feb 01 12:09:53 ubuntu-server systemd[1]: Started The Apache HTTP Server.
student@ubuntu-server:~$ ss -tuln | grep :80
tcp  LISTEN  0      511           *:80              *:*
student@ubuntu-server:~$ _
```

Figure 11: Apache2 status is active and `ss -tuln` confirms port 80 listener on VM3



```
File Machine View Input Devices Help
student@ubuntu-client:~$ curl http://10.0.2.2
<h1>HELLO FROM VM3 (Ubuntu Server)</h1>
<p>Apache HTTP server is working on port 80</p>
student@ubuntu-client:~$ _
```

```
ubuntu-server [Running] - Oracle VirtualBox
File Machine View Input Devices Help
student@ubuntu-server:~$ curl http://localhost
<h1>HELLO FROM VM3 (Ubuntu Server)</h1>
<p>Apache HTTP server is working on port 80</p>
student@ubuntu-server:~$ _
```

Figure 12: curl from VM1 successfully fetches the HTTP page from VM3 (10.0.2.2)

## 8 Objective 4: Show SSH Works Before Firewall (Baseline)

Before enforcing firewall rules, we establish a baseline showing SSH connectivity exists.

### Why this is important:

- It proves the network path is correct.
- Later, if SSH fails, we can attribute it to firewall enforcement (not routing misconfig).

```
student@ubuntu-client:~$ ssh student@10.0.2.2
The authenticity of host '10.0.2.2 (10.0.2.2)' can't be established.
ED25519 key fingerprint is SHA256:up4e1Np5K7bE2x3+adYCD3N/A6IL1K2wQNhz1s7w8Wk.
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 '10.0.2.2' (ED25519) to the list of known hosts.
student@10.0.2.2's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-164-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sun Feb  1 12:46:32 PM UTC 2026

 System load:  0.0          Processes:           111
 Usage of /:   46.1% of 11.21GB  Users logged in:     1
 Memory usage: 12%          IPv4 address for enp0s3: 10.0.2.2
 Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection
or proxy settings

Last login: Sun Feb  1 11:54:39 2026
student@ubuntu-server:~$
```

Figure 13: SSH connection from VM1 to VM3 successfully established (baseline before firewall)

```

student@ubuntu-client:~$ ssh student@10.0.2.2
student@10.0.2.2's password:
Welcome to Ubuntu 22.04.5 LTS (GNU/Linux 5.15.0-164-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Sun Feb  1 12:48:39 PM UTC 2026

  System load:  0.03           Processes:          111
  Usage of /:   46.1% of 11.21GB  Users logged in:     1
  Memory usage: 12%            IPv4 address for enp0s3: 10.0.2.2
  Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

1 additional security update can be applied with ESM Apps.
Learn more about enabling ESM Apps service at https://ubuntu.com/esm

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection
or proxy settings

Last login: Sun Feb  1 12:46:33 2026 from 10.0.1.2
student@ubuntu-server:~$ exit
logout
Connection to 10.0.2.2 closed.
student@ubuntu-client:~$
```

Figure 14: Successful SSH login session and exit (shows SSH works pre-firewall)

## 9 Objective 5: Implement Firewall Rules using pf on VM2

### 9.1 Firewall Policy (Default Block + Allow Only Required)

We implement:

- block all by default
- allow ICMP (ping)
- allow TCP/80 (HTTP)
- do not allow TCP/22 (SSH remains blocked)

### 9.2 pf Rule-set

```

block all

pass inet proto icmp all keep state
```

```

pass in on em0 inet proto tcp from any to 10.0.2.2 port 80 flags
    S/SA keep state
pass out on em1 inet proto tcp from any to 10.0.2.2 port 80 flags
    S/SA keep state

```

Listing 4: /etc/pf.conf rules used in Task 1

### 9.3 Explanation of Each Rule in Detail

- `block all`: denies everything unless a later rule allows it (default deny).
- `pass inet proto icmp all keep state`: allows ICMP in IPv4 for diagnostics.
- `pass in on em0 ... port 80`: allows client-side incoming HTTP requests towards VM3.
- `pass out on em1 ... port 80`: allows those HTTP packets to leave toward server-side network.
- `flags S/SA`: ensures the rule matches TCP connection initiation correctly.
- `keep state`: ensures reply traffic is allowed automatically.

### 9.4 Enabling pf (Commands and Verification)

```

kldload pf
sysrc pf_enable="YES"
pfctl -nf /etc/pf.conf
pfctl -f /etc/pf.conf
pfctl -e
pfctl -sr

```

Listing 5: Commands to load and enable pf

#### What these commands do:

- `kldload pf`: loads pf kernel module.
- `sysrc pf_enable = "YES"` : enables pf service at boot.
- `pfctl -nf`: syntax check of the `pf.conf` file (safe validation).
- `pfctl -f`: loads the rules into the kernel.
- `pfctl -e`: enables packet filtering (if not already enabled).

- `pfctl -sr`: prints active rule-set (mandatory proof).

```
root@firewall:~ # kldload pf
root@firewall:~ # sysrc pf_enable="YES"
pf_enable: NO -> YES
root@firewall:~ #
```

Figure 15: Loading pf module and enabling pf service on VM2

```
root@firewall:~ # pfctl -nf /etc/pf.conf
root@firewall:~ # pfctl -f /etc/pf.conf
root@firewall:~ # pfctl -e
pfctl: pf already enabled
root@firewall:~ # pfctl -sr
block return all
pass in on em0 inet proto tcp from any to 10.0.2.2 port = http flags S/SR keep state
pass out on em1 inet proto tcp from any to 10.0.2.2 port = http flags S/SR keep state
pass inet proto icmp all keep state
root@firewall:~ #
```

Figure 16: pf rule-set shown using `pfctl -sr` (proof that rules are active)

## 10 Objective 6: Verification After Firewall Enforcement

Now we test the required outcomes:

- HTTP should pass
- SSH should be blocked
- ICMP should still pass

```
student@ubuntu-client:~$ curl http://10.0.2.2
<h1>HELLO FROM VM3 (Ubuntu Server)</h1>
<p>Apache HTTP server is working on port 80</p>
student@ubuntu-client:~$ ssh student@10.0.2.2
ssh: connect to host 10.0.2.2 port 22: Connection refused
student@ubuntu-client:~$ traceroute -I 10.0.2.2
traceroute to 10.0.2.2 (10.0.2.2), 30 hops max, 60 byte packets
 1  10.0.1.1 (10.0.1.1)  1.183 ms  1.070 ms  1.057 ms
 2  10.0.2.2 (10.0.2.2)  2.063 ms  2.052 ms  2.039 ms
student@ubuntu-client:~$ ping 10.0.2.2
PING 10.0.2.2 (10.0.2.2) 56(84) bytes of data.
 64 bytes from 10.0.2.2: icmp_seq=1 ttl=63 time=2.38 ms
 64 bytes from 10.0.2.2: icmp_seq=2 ttl=63 time=2.15 ms
^C
--- 10.0.2.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 2.147/2.262/2.377/0.115 ms
student@ubuntu-client:~$
```

Figure 17: Verification: curl (HTTP) works; SSH fails; ICMP ping works

## 10.1 Why SSH Fails

SSH uses TCP port 22. Since we never explicitly allowed port 22 in pf.conf:

- `block all` applies
- SSH packets are dropped/blocked
- connection fails

This confirms correct enforcement of least privilege.

## 11 Objective 7: Firewall Logging (pflog) and Evidence

Logging provides visibility and forensic evidence. It shows exactly which packets are blocked and why.

### What is pflog?

- `pflog` is a pseudo-interface used by pf for logging.
- Logs can be viewed via tcpdump on `pflog0`.

```
tcpdump -n -e -ttt -i pflog0
```

Listing 6: Typical command to view logs

The figure shows two terminal windows side-by-side. The left window displays a series of network commands and their results:

```

Machine View Input Devices Help
student@ubuntu-client:~$ ssh student@10.0.2.2
ssh: connect to host 10.0.2.2 port 22: Connection refused
student@ubuntu-client:~$ traceroute 10.0.2.2
traceroute to 10.0.2.2 (10.0.2.2), 30 hops max, 60 byte packets
  1  10.0.1.1 (10.0.1.1)  0.423 ms  0.392 ms  0.383 ms
student@ubuntu-client:~$ traceroute 10.0.2.2
traceroute to 10.0.2.2 (10.0.2.2), 30 hops max, 60 byte packets
  1  10.0.1.1 (10.0.1.1)  1.161 ms  1.364 ms  1.334 ms
student@ubuntu-client:~$ ssh student@10.0.2.2
ssh: connect to host 10.0.2.2 port 22: Connection refused
student@ubuntu-client:~$ _

```

The right window shows live pflog output, listing numerous blocked UDP and TCP packets. The log entries are as follows:

```

02:52:31.634774 rule 0/0(match): block in on em0: 10.0.1.2.51015 > 10.0.2.2.3344
6: UDP, length 32
02:52:31.634784 rule 0/0(match): block in on em0: 10.0.1.2.37781 > 10.0.2.2.3344
7: UDP, length 32
02:52:31.634794 rule 0/0(match): block in on em0: 10.0.1.2.37081 > 10.0.2.2.3344
8: UDP, length 32
02:52:31.634948 rule 0/0(match): block in on em0: 10.0.1.2.59825 > 10.0.2.2.3344
9: UDP, length 32
02:52:31.636482 rule 0/0(match): block in on em0: 10.0.1.2.48244 > 8.8.8.8.53: 3
6280+ PTR? 1.1.0.10.in-addr.arpa. (39)
02:52:31.637235 rule 0/0(match): block in on em0: 10.0.1.2.55036 > 8.8.8.8.53: F
lags [S], seq 4076132382, win 64240, options [mss 1460,sackOK,TS val 3577679434
ecr 0,nop,wscale 7,tfo cookiereq,nop,nopl], length 0
02:52:31.637402 rule 0/0(match): block in on em0: 10.0.1.2.55040 > 8.8.8.8.53: F
lags [S], seq 4157682104, win 64240, options [mss 1460,sackOK,TS val 3577679434
ecr 0,nop,wscale 7,tfo cookiereq,nop,nopl], length 0
02:52:31.637529 rule 0/0(match): block in on em0: 10.0.1.2.55046 > 8.8.8.8.53: F
lags [S], seq 1863499463, win 64240, options [mss 1460,sackOK,TS val 3577679434
ecr 0,nop,wscale 7,tfo cookiereq,nop,nopl], length 0
02:52:31.637764 rule 0/0(match): block in on em0: 10.0.1.2.34028 > 8.8.8.8.53: 4
7647+ PTR? 1.1.0.10.in-addr.arpa. (39)
02:52:47.452356 rule 0/0(match): block in on em0: 10.0.1.2.38658 > 10.0.2.2.22:
Flags [S], seq 3218299710, win 64240, options [mss 1460,sackOK,TS val 1473348808
ecr 0,nop,wscale 7], length 0

```

Figure 18: Live pflog output showing blocked SSH and other packets

The figure shows two screenshots of a terminal window. The top screenshot displays a series of network commands and their outputs. It includes:

```

Machine View Input Devices Help
1 10.0.1.1 (10.0.1.1) 1.451 ms 1.798 ms 1.763 ms
student@ubuntu-client:~$ traceroute 10.0.2.2
traceroute to 10.0.2.2 (10.0.2.2), 30 hops max, 60 byte packets
 1 10.0.1.1 (10.0.1.1) 1.193 ms 1.104 ms 1.330 ms
student@ubuntu-client:~$ ssh student@10.0.2.2
ssh: connect to host 10.0.2.2 port 22: Connection refused
student@ubuntu-client:~$ curl http://10.0.2.2
<h1>HELLO FROM VM3 (Ubuntu Server)</h1>
<p>Apache HTTP server is working on port 80</p>
student@ubuntu-client:~$ curl http://10.0.2.2
<h1>HELLO FROM VM3 (Ubuntu Server)</h1>
<p>Apache HTTP server is working on port 80</p>
student@ubuntu-client:~$
```

The bottom screenshot shows a detailed log of network traffic captured by pflog. The log entries include:

```

8932+ AAAA? ntp.ubuntu.com. (32)
03:01:01.801423 rule 0/0(match): block in on em1: 10.0.2.2.52230 > 8.8.8.8.53: F
lags [S], seq 2502562698, win 64240, options [mss 1460,sackOK,TS val 3057339269
ecr 0,nop,wscale 7,tfo cookiereq,nop,nop], length 0
03:01:01.801566 rule 0/0(match): block in on em1: 10.0.2.2.52244 > 8.8.8.8.53: F
lags [S], seq 991425690, win 64240, options [mss 1460,sackOK,TS val 3057339269 e
cr 0,nop,wscale 7,tfo cookiereq,nop,nop], length 0
03:01:01.801691 rule 0/0(match): block in on em1: 10.0.2.2.52258 > 8.8.8.8.53: F
lags [S], seq 3597243813, win 64240, options [mss 1460,sackOK,TS val 3057339269
ecr 0,nop,wscale 7,tfo cookiereq,nop,nop], length 0
03:01:01.801823 rule 0/0(match): block in on em1: 10.0.2.2.46689 > 8.8.8.8.53: 4
8866+ AAAA? ntp.ubuntu.com. (32)
03:01:01.801830 rule 0/0(match): block in on em1: 10.0.2.2.47615 > 8.8.8.8.53: 6
808+ A? ntp.ubuntu.com. (32)
03:01:01.802045 rule 0/0(match): block in on em1: 10.0.2.2.55885 > 8.8.8.8.53: 1
366+ A? ntp.ubuntu.com. (32)
03:01:01.802123 rule 0/0(match): block in on em1: 10.0.2.2.54556 > 8.8.8.8.53: 5
8274+ AAAA? ntp.ubuntu.com. (32)
03:01:07.284816 rule 1/0(match): pass in on em0: 10.0.1.2.42466 > 10.0.2.2.80: F
lags [S], seq 4082539128, win 64240, options [mss 1460,sackOK,TS val 1473848730
ecr 0,nop,wscale 7], length 0
03:01:07.284821 rule 2/0(match): pass out on em1: 10.0.1.2.42466 > 10.0.2.2.80:
Flags [S], seq 4082539128, win 64240, options [mss 1460,sackOK,TS val 1473848730
ecr 0,nop,wscale 7], length 0
```

Figure 19: Additional pflog evidence: blocked traffic and allowed HTTP traffic

## 12 Conclusion

Task 1 was completed successfully by configuring VM2 (FreeBSD 13.4) as an L3 forwarding firewall using pf.

### Final outcomes:

- L3 forwarding enabled and verified (`net.inet.ip.forwarding=1`)
- Routing path verified using traceroute (VM1 → 10.0.1.1 → VM3)
- pf deployed with a default deny posture

- ICMP allowed for diagnostics
- HTTP (TCP/80) allowed and verified using curl
- SSH (TCP/22) blocked after firewall activation
- Logging confirmed using pflog evidence

This setup reflects a real-world secure gateway firewall design enforcing least privilege.