CS790 Assignment 1

The Diamond Shield
Report

Kavya Gupta (22B1053)
Department of Computer Science,
Indian Institute of Technology Bombay

Task 1

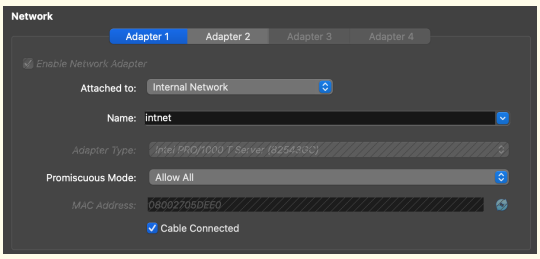
Setup

- Used Oracle VirtualBox
- All 3 VMs are FreeBSD 13.4
- VM1 and 3 were assigned 1024MB RAM and 1 CPU and VM2 was assigned 2048MB RAM and 2 CPUs

Part (a): Configuring the Network

The steps followed are:-

1. In the "Network" section of the VM settings, I added a new "Internal Network" for each VM for the inter-VM communication. `intnet` for VM2-3 and `intnet2` for VM1-2.



2. Inside the VM, I changed the name of interfaces (E.g. `em0` to `eth0` in VM1 and 3) and assigned static IP addresses to the interfaces. To achieve this, I added the following lines to the `/etc/rc.conf` file (these are for VM1, similar for VM2 and 3):-

   - `ifconfig_em0_name="eth0"`
   - `ifconfig_eth0="inet 192.168.10.2 netmask 255.255.255.0 up"`

   These are the IP addresses assigned to the interfaces:-

- VM1 `eth0`: `192.168.10.2`
- VM2 `eth0`: `192.168.10.1`, `eth1`: `192.168.20.1`
- VM3 `eth0`: `192.168.20.2`

Hence VM1 and 2 belong to the `192.168.10.0/24` subnet and VM2 and 3 belong to the `192.168.20.0/24` subnet.

Part (b): Enabling IP Forwarding

The steps followed are:-

1. Added `gateway_enable="YES"` to the `/etc/rc.conf` file of VM2.
2. Added `net.inet.ip.forwarding=1` to the `/etc/sysctl.conf` file of VM2.
3. Made a new static route (to VM3) in VM1 by adding the following line to the `/etc/rc.conf` file of VM1:-
   - `static_routes="net20"`
   - `route_net20="-net 192.168.20.0/24 192.168.10.1"`
4. Similarly made a new static route (to VM1) in VM3.

Note: Routing table can be checked using the `netstat -r` command.
Traceroute from VM1 to VM3 shows that it goes via VM2:-

```
root@vm1:~ # traceroute 192.168.20.2
traceroute to 192.168.20.2 (192.168.20.2), 64 hops max, 40 byte packets
 1  192.168.10.1 (192.168.10.1)  1.251 ms  0.517 ms  0.489 ms
 2  192.168.20.2 (192.168.20.2)  1.674 ms  1.202 ms  1.222 ms
root@vm1:~ #
```

Part (c): Communicating using HTTP

The steps followed are:-

1. To enable Internet access in VM3, I added a "Bridged Adapter" to VM3's "Network" section. Also had to run `dhclient em1` to get an IP address.
2. Installed the Apache web server in VM3 using the command `pkg install apache24`.
3. Started the Apache service using the command `service apache24 start`. Also added `apache24_enable="YES"` to the `/etc/rc.conf` file.
4. Ensured that `LISTEN 80` is there in the `/usr/local/etc/apache24/httpd.conf` file. Also had to change the `ServerName` to `192.168.20.2:80`.
5. Files to be shared are supposed to be in the `/usr/local/www/apache24/data` directory. By default, the `index.html` file is there.
6. Installed the `wget` package in VM1 using the command `pkg install wget`. Then ran the command `wget http://192.168.20.2/index.html`, the results are shown below:-

```
root@vm1:~ # wget http://192.168.20.2/index.html
--2025-02-13 06:58:35--  http://192.168.20.2/index.html
Connecting to 192.168.20.2:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 45 [text/html]
Saving to: 'index.html'

index.html          100%[===================>]      45  --.-KB/s    in 0s

2025-02-13 06:58:35 (7.41 MB/s) - 'index.html' saved [45/45]

root@vm1:~ # cat index.html
<html><body><h1>It works!</h1></body></html>
root@vm1:~ #
```

Part (d): Packet Filtering using `pf`

The steps followed are:-

1. Added the following lines to the `/etc/pf.conf` file of VM2:-

   - `block in on eth0 inet from 192.168.10.2 to 192.168.20.2`
   - `block out on eth1 inet from 192.168.10.2 to 192.168.20.2`
   - `pass in on eth0 inet proto tcp from 192.168.10.2 to 192.168.20.2 port 80 keep state`
   - `pass out on eth1 inet proto tcp from 192.168.10.2 to 192.168.20.2 port 80 keep state`

   The first two lines make sure that any traffic from VM1 to VM3 via VM2 is blocked. The next two lines allow ONLY the HTTP traffic through this route.

2. Enabled the `pf` service using the command `pfctl -e`. Also added `pf_enable="YES"` to the `/etc/rc.conf` file of VM2.

3. Uploaded the rules using the command `pfctl -f /etc/pf.conf`.

   Note: `pf` can be disabled using the command `pfctl -d`.

Below is the output of `ssh` from VM1 to VM3 with `pf` disabled:-

```
root@vm1:~ # ssh root@192.168.20.2
(root@192.168.20.2) Password for root@vm3:
```

And below is the same but with `pf` enabled:-

```
root@vm1:~ # ssh root@192.168.20.2
ssh: connect to host 192.168.20.2 port 22: Operation timed out
root@vm1:~ #
```

Now, the output of `wget` from VM1 to VM3 with `pf` enabled is as in the figure of previous part.

References

- FreeBSD Handbook, Docs. VirtualBox Manual. The book of pf.
- `https://www.nakivo.com/blog/virtualbox-network-setting-guide/`

Task 2

Files Submitted

- `icmp_block.c`
- `Makefile`

Instructions to Execute the Code

Make sure that `Makefile` and `icmp_block.c` are present in the same directory. In that directory:-

1. Run `make` to compile the kernel module.
2. Run `pfctl -d` to ensure that `pf` is disabled.
3. Run `kldload ./icmp_block.ko` to load the kernel module.
4. Run `kldunload icmp_block` to unload the module.

Note: All of this was done as the root user.

Little Overview of the Code

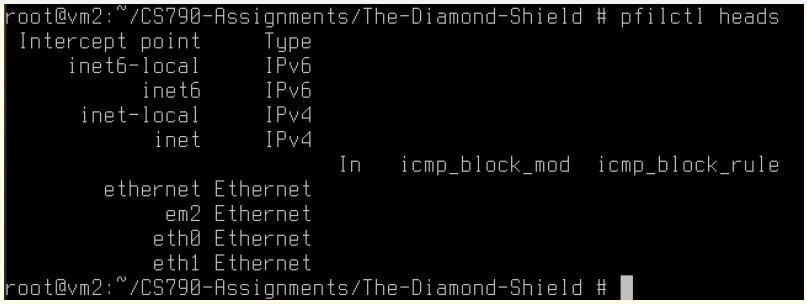
- `icmp_block_hook`: This is the hook function that is ran over each IP packet and blocks those with the ICMP Protocol and Echo Request type and inward direction. `PFIL_IN` is used for it.
- `load_handler`: This function is called when the module is loaded/unloaded. It adds `icmp_block_hook` to the hooks list and links it with the `inet` head in inward direction.

Additional Steps

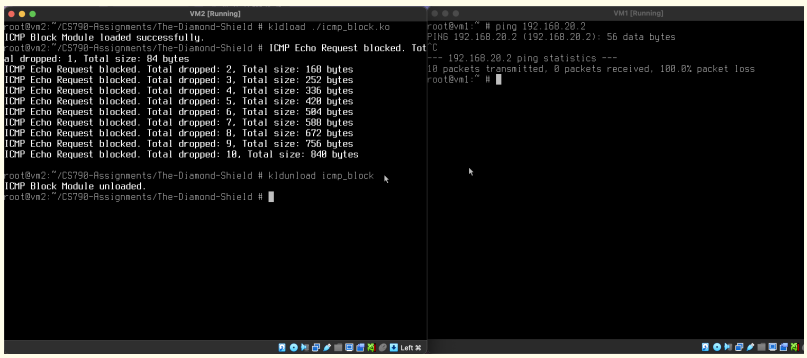
- Ran `wget https://download.freebsd.org/releases/arm64/13.4-RELEASE/src.txz` and un-tared it at `/usr/src` to get the kernel source modules.
- Installed `gcc` and `make` using the command `pkg install gcc gmake`.
- Used `https://tylersguides.com/guides/how-to-increase-the-size-of-a-freebsd-disk/` to increase the disk size of VM2.
- `pfilctl heads` and `pfilctl hooks` give us currently active heads and hooks respectively.

Output

This is the output of `pfilctl heads` after loading the module:-



This is the output of `ping` from VM1 to VM3 with the module loaded:-



As you can see that the packets are successfully being dropped.

Resources Used

- Used a little bit of ChatGPT to get a headstart. Got the idea to use the `net/pfil.h` library.
- Used the man pages of `pfil(9)` to get idea of the functions to be used.
- At `http://fxr.watson.org/fxr/source/net/pfil.h?v=FREEBSD-13-0`, found the source code of `pfil.h` (and of `pfil.c` similarly).
- `https://medium.com/rossdotpink/writing-a-simple-freebsd-kernel-module-9302bd4cfae1`