

# Lab 5: Firewall Exploration

Student ID: 1004885

Name: Guo Yuchen

## Task 1: Implementing a Simple Firewall

### Task 1.A: Implement a Simple Kernel Module

Make and load the hello.ko module:

```
$ sudo insmod hello.ko
$ lsmod | grep hello
$ dmesg
```

```
seed@VM: ~/.../kernel_module
seed@VM: ~/.../kernel_module 141x37
[40024.162386] device vethb87f89f entered promiscuous mode
[40024.163528] br-7a01a710e225: port 2(vethb87f89f) entered blocking state
[40024.163530] br-7a01a710e225: port 2(vethb87f89f) entered forwarding state
[40024.163957] br-7a01a710e225: port 2(vethb87f89f) entered disabled state
[40024.388826] br-be06a02f90b6: port 4(vethb7b9318) entered blocking state
[40024.388828] br-be06a02f90b6: port 4(vethb7b9318) entered disabled state
[40024.389485] device vethb7b9318 entered promiscuous mode
[40024.389621] br-be06a02f90b6: port 4(vethb7b9318) entered blocking state
[40024.389623] br-be06a02f90b6: port 4(vethb7b9318) entered forwarding state
[40024.939323] br-be06a02f90b6: port 4(vethb7b9318) entered disabled state
[40026.105031] eth0: renamed from veth9a63e29
[40026.118283] IPv6: ADDRCONF(NETDEV_CHANGE): vethf5fc4d5: link becomes ready
[40026.118428] br-be06a02f90b6: port 3(vethf5fc4d5) entered blocking state
[40026.118460] br-be06a02f90b6: port 3(vethf5fc4d5) entered forwarding state
[40026.176712] eth0: renamed from veth36a6a76
[40026.186921] IPv6: ADDRCONF(NETDEV_CHANGE): vethd973ebe: link becomes ready
[40026.187721] br-be06a02f90b6: port 2(vethd973ebe) entered blocking state
[40026.187724] br-be06a02f90b6: port 2(vethd973ebe) entered forwarding state
[40026.262702] eth0: renamed from veth33532b3
[40026.302966] IPv6: ADDRCONF(NETDEV_CHANGE): veth9530f24: link becomes ready
[40026.303004] br-7a01a710e225: port 1(veth9530f24) entered blocking state
[40026.303006] br-7a01a710e225: port 1(veth9530f24) entered forwarding state
[40026.330742] eth0: renamed from veth12ee33f
[40026.342566] eth0: renamed from veth3eb1545
[40026.360742] IPv6: ADDRCONF(NETDEV_CHANGE): vethb87f89f: link becomes ready
[40026.360845] br-7a01a710e225: port 2(vethb87f89f) entered blocking state
[40026.360847] br-7a01a710e225: port 2(vethb87f89f) entered forwarding state
[40026.365871] IPv6: ADDRCONF(NETDEV_CHANGE): veth33cf4bb: link becomes ready
[40026.366094] br-be06a02f90b6: port 1(veth33cf4bb) entered blocking state
[40026.366098] br-be06a02f90b6: port 1(veth33cf4bb) entered forwarding state
[40026.391975] eth1: renamed from vethe9f852a
[40026.410500] IPv6: ADDRCONF(NETDEV_CHANGE): vethb7b9318: link becomes ready
[40026.410614] br-be06a02f90b6: port 4(vethb7b9318) entered blocking state
[40026.410616] br-be06a02f90b6: port 4(vethb7b9318) entered forwarding state
[41105.632801] hello: module verification failed: signature and/or required key missing - tainting kernel
[41105.635940] Hello World!
[10/20/23]seed@VM:~/.../kernel_module$
```

## Task 1.B: Implement a Simple Firewall Using Netfilter

### Task 1

```
[40026.410614] br-be06a02f90b6: port 4(vethb7b9318) entered blocking state
[40026.410616] br-be06a02f90b6: port 4(vethb7b9318) entered forwarding state
[41105.632801] hello: module verification failed: signature and/or required key missing
[41105.635940] Hello World!
[42215.143411] Registering filters.
[42222.061445] *** LOCAL_OUT
[42222.061878] 10.0.2.15 --> 104.46.162.224 (TCP)
[42267.121790] *** LOCAL_OUT
[42267.122031] 10.0.2.15 --> 104.46.162.224 (TCP)
[42267.709709] *** LOCAL_OUT
[42267.709713] 127.0.0.1 --> 127.0.0.1 (UDP)
[42267.710581] *** LOCAL_OUT
[42267.710584] 10.0.2.15 --> 8.8.8.8 (UDP)
[42267.710595] *** Dropping 8.8.8.8 (UDP), port 53
[42272.707629] *** LOCAL_OUT
[42272.707633] 10.0.2.15 --> 8.8.8.8 (UDP)
[42272.707650] *** Dropping 8.8.8.8 (UDP), port 53
[10/20/23]seed@VM:~/.../packet_filter$
```

```
[10/20/23]seed@VM:~/.../packet_filter$ dig @8.8.8.8 www.example.com

; <<>> DiG 9.16.1-Ubuntu <<>> @8.8.8.8 www.example.com
; (1 server found)
;; global options: +cmd
;; connection timed out; no servers could be reached
```

Request is blocked, since all DNS packages which goes to 8.8.8.8 are dropped.

### Task2

Apply printInfo function to all hooks:

```
Int registerFilter(void) {
    // omit previous hooks...
    hook3.hook = printInfo;
    hook3.hooknum = NF_INET_PRE_ROUTING;
    hook3.pf = PF_INET;
    hook3.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook3);

    hook4.hook = printInfo;
    hook4.hooknum = NF_INET_LOCAL_IN;
    hook4.pf = PF_INET;
    hook4.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook4);

    hook5.hook = printInfo;
```

```

hook5.hooknum = NF_INET_FORWARD;
hook5.pf = PF_INET;
hook5.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook5);

hook6.hook = printInfo;
hook6.hooknum = NF_INET_POST_ROUTING;
hook6.pf = PF_INET;
hook6.priority = NF_IP_PRI_FIRST;
nf_register_net_hook(&init_net, &hook6);
}

```

After dig @8.8.8.8 www.example.com:

```

[42860.381345] Registering filters.
[42882.527055] *** LOCAL_OUT
[42882.527058] 127.0.0.1 --> 127.0.0.1 (UDP)
[42882.527073] *** POST_ROUTING
[42882.527074] 127.0.0.1 --> 127.0.0.1 (UDP)
[42882.527092] *** PRE_ROUTING
[42882.527093] 127.0.0.1 --> 127.0.0.1 (UDP)
[42882.527095] *** LOCAL_IN
[42882.527096] 127.0.0.1 --> 127.0.0.1 (UDP)
[42882.527672] *** LOCAL_OUT
[42882.527674] 10.0.2.15 --> 8.8.8.8 (UDP)
[42882.527679] *** POST_ROUTING
[42882.527680] 10.0.2.15 --> 8.8.8.8 (UDP)
[42882.527681] *** Dropping 8.8.8.8 (UDP), port 53
[42883.561872] *** LOCAL_OUT
[42883.561876] 10.0.2.15 --> 13.107.42.18 (TCP)
[42883.561888] *** POST_ROUTING
[42883.561889] 10.0.2.15 --> 13.107.42.18 (TCP)
[42883.562213] *** PRE_ROUTING
[42883.562214] 13.107.42.18 --> 10.0.2.15 (TCP)
[42883.562222] *** LOCAL_IN
[42883.562265] 13.107.42.18 --> 10.0.2.15 (TCP)

```

Under what condition will each of the hook function be invoked:

- **NF\_INET\_PRE\_ROUTING**: when a packet is received by the network stack, before the routing decision is made.
- **NF\_INET\_LOCAL\_IN**: when a packet is about to be delivered locally (the packet's destination IP address matches one of the local system's interfaces). It happens after routing decisions have been made but before any local processing takes place.
- **NF\_INET\_FORWARD**: when a packet is to be forwarded through the system.
- **NF\_INET\_LOCAL\_OUT**: when a locally generated packet is about to leave the system.
- **NF\_INET\_POST\_ROUTING**: before a packet is sent out on a network interface.

## Task3

Implement two more hooks

```
// preventing other computers to ping the VM
unsigned int blockPing(void *priv, struct sk_buff *skb,
                        const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct icmphdr *icmph;

    if (!skb)
        return NF_ACCEPT;

    iph = ip_hdr(skb);
    icmph = icmp_hdr(skb);

    if (icmph->type == ICMP_ECHO)
    {
        printk(KERN_WARNING "%* Dropping %pI4 (ICMP_ECHO)\n", &(iph->daddr));
        return NF_DROP;
    }
    return NF_ACCEPT;
}

// preventing other computers to telnet into the VM
// Telnet's default port is TCP port 23.
unsigned int blockTelnet(void *priv, struct sk_buff *skb,
                          const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct tcphdr *tcph;

    u16 port = 23;

    if (!skb)
        return NF_ACCEPT;

    iph = ip_hdr(skb);

    if (iph->protocol == IPPROTO_TCP)
    {
        tcph = tcp_hdr(skb);
```

```

        if (ntohs(tcp->dest) == port)
        {
            printk(KERN_WARNING "* Dropping %pI4 (TCP), port %d\n", &(iph->daddr),
port);
            return NF_DROP;
        }
    }
    return NF_ACCEPT;
}

```

```

Int registerFilter(void) {
    // omit previous hooks...
    // (1) preventing other computers to ping the VM
    hook7.hook = blockPing;
    hook7.hooknum = NF_INET_LOCAL_IN;
    hook7.pf = PF_INET;
    hook7.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook7);

    // (2) preventing other computers to telnet into the VM.
    hook8.hook = blockTelnet;
    hook8.hooknum = NF_INET_LOCAL_IN;
    hook8.pf = PF_INET;
    hook8.priority = NF_IP_PRI_FIRST;
    nf_register_net_hook(&init_net, &hook8);
}

```

ping 10.9.0.1 from 10.9.0.5 -> could not connect

```

[10/21/23]seed@VM:~/.../packet_filter$ dockps
71faf90aa381  host2-192.168.60.6
4cb3636f4752  hostA-10.9.0.5
72812424697c  host3-192.168.60.7
ff8dbac15e29  seed-router
7c007693064a  host1-192.168.60.5
[10/21/23]seed@VM:~/.../packet_filter$ docksh 4c
root@4cb3636f4752:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.

```



```

seed@VM: ~/.../packet_filter 70x18
[45136.346528] 10.0.2.15 --> 52.168.117.170 (TCP)
[45136.346539] *** POST_ROUTING
[45136.346540] 10.0.2.15 --> 52.168.117.170 (TCP)
[45136.347225] *** PRE_ROUTING
[45136.347229] 52.168.117.170 --> 10.0.2.15 (TCP)
[45136.347240] *** LOCAL_IN
[45136.347241] 52.168.117.170 --> 10.0.2.15 (TCP)
[45139.138199] *** PRE_ROUTING
[45139.138202] 10.9.0.5 --> 10.9.0.1 (ICMP)
[45139.138209] *** PRE_ROUTING
[45139.138210] 10.9.0.5 --> 10.9.0.1 (ICMP)
[45139.138212] *** Dropping 10.9.0.1 (ICMP_ECHO)
[45140.153850] *** PRE_ROUTING
[45140.153853] 10.9.0.5 --> 10.9.0.1 (ICMP)
[45140.153872] *** PRE_ROUTING
[45140.153873] 10.9.0.5 --> 10.9.0.1 (ICMP)
[45140.153878] *** Dropping 10.9.0.1 (ICMP_ECHO)
[10/21/23]seed@VM:~/.../packet_filter$

```

telnet 10.9.0.1 from 10.9.0.5 -> could not connect

```

root@4cb3636f4752:/# telnet 10.9.0.1
Trying 10.9.0.1...

```

```

seed@VM: ~/.../packet_filter 70x18
[45194.309812] 127.0.0.53 --> 127.0.0.1 (UDP)
[45194.309814] *** POST_ROUTING
[45194.309814] 127.0.0.53 --> 127.0.0.1 (UDP)
[45194.309818] *** PRE_ROUTING
[45194.309818] 127.0.0.53 --> 127.0.0.1 (UDP)
[45194.309819] *** LOCAL_IN
[45194.309819] 127.0.0.53 --> 127.0.0.1 (UDP)
[45198.888319] *** PRE_ROUTING
[45198.888321] 10.9.0.5 --> 10.9.0.1 (TCP)
[45198.888330] *** PRE_ROUTING
[45198.888330] 10.9.0.5 --> 10.9.0.1 (TCP)
[45198.888334] *** Dropping 10.9.0.1 (TCP), port 23
[45199.895151] *** PRE_ROUTING
[45199.895313] 10.9.0.5 --> 10.9.0.1 (TCP)
[45199.895436] *** PRE_ROUTING
[45199.895588] 10.9.0.5 --> 10.9.0.1 (TCP)
[45199.895706] *** Dropping 10.9.0.1 (TCP), port 23
[10/21/23]seed@VM:~/.../packet_filter$

```

## Task 2: Experimenting with Stateless Firewall Rules

### Task 2.A: Protecting the Router

```
// Append a rule in the INPUT chain to accept all icmp echo-request packets.
iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
// Append a rule in the OUTPUT chain to accept all icmp echo-reply packets.
iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
// Set default policy for OUTPUT
iptables -P OUTPUT DROP
// Set default policy for INPUT
iptables -P INPUT DROP
```

Before applying rule:

```
root@guoyuchen-hostA-10:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.151 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.132 ms
^C
--- 10.9.0.11 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1012ms
rtt min/avg/max/mdev = 0.132/0.141/0.151/0.009 ms
```

```
root@guoyuchen-hostA-10:/# telnet 10.9.0.11
Trying 10.9.0.11...
Connected to 10.9.0.11.
Escape character is '^['.
Ubuntu 20.04.1 LTS
guoyuchen-seed-router login: ^CConnection closed by foreign host.
root@guoyuchen-hostA-10:/#
```

Both ping and telnet can connect.

Implement rules:

```
root@guoyuchen-seed-router:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
root@guoyuchen-seed-router:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@guoyuchen-seed-router:/# iptables -P OUTPUT DROP
root@guoyuchen-seed-router:/# iptables -P INPUT DROP
root@guoyuchen-seed-router:/# iptables -t filter -L -n --line-numbers
Chain INPUT (policy DROP)
num target      prot opt source                destination            icmptype
1  ACCEPT        icmp -- 0.0.0.0/0              0.0.0.0/0              icmp_type 8

Chain FORWARD (policy ACCEPT)
num target      prot opt source                destination

Chain OUTPUT (policy DROP)
num target      prot opt source                destination            icmptype
1  ACCEPT        icmp -- 0.0.0.0/0              0.0.0.0/0              icmp_type 0
```

Can ping

```
root@guoyuchen-hostA-10:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.213 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.192 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.170 ms
^C
--- 10.9.0.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2069ms
rtt min/avg/max/mdev = 0.170/0.191/0.213/0.017 ms
```

Cannot telnet

```
root@guoyuchen-hostA-10:/# telnet 10.9.0.11
Trying 10.9.0.11...
telnet: Unable to connect to remote host: Connection timed out
```

## Task 2.B: Protecting the Internal Network

```
root@guoyuchen-seed-router:/# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.9.0.11 netmask 255.255.255.0 broadcast 10.9.0.255
    ether 02:42:0a:09:00:0b txqueuelen 0 (Ethernet)
    RX packets 157 bytes 13742 (13.7 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 116 bytes 8492 (8.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.60.11 netmask 255.255.255.0 broadcast 192.168.60.255
    ether 02:42:c0:a8:3c:0b txqueuelen 0 (Ethernet)
    RX packets 131 bytes 12229 (12.2 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 74 bytes 5316 (5.3 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Thus out-interface: eth0, in-interface: eth1

```
// Outside hosts cannot ping internal hosts. All icmp echo request packages from
eth0 will be dropped.
```

```
iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP
```

```
// Outside hosts can ping the router. All icmp echo request packages from eth0
to the router will be accepted.
```

```
iptables -A FORWARD -i eth1 -d 10.9.0.11 -p icmp --icmp-type echo-request -j ACCEPT
```

```
// Internal hosts can ping outside hosts. All icmp echo request packages from eth1
and icmp echo reply packages from eth0 will be accepted.
```

```
iptables -I FORWARD -i eth1 -o eth0 -p icmp --icmp-type echo-request -j ACCEPT
```



```
iptables -I FORWARD -i eth0 -o eth1 -p icmp --icmp-type echo-reply -j ACCEPT

// All other packets between the internal and external networks should be dropped.
iptables -P FORWARD DROP
```

```
root@guoyuchen-seed-router:/# iptables -t filter -L -n --line-numbers
Chain INPUT (policy ACCEPT)
num target      prot opt source      destination
Chain FORWARD (policy DROP)
num target      prot opt source      destination      icmptype
1  ACCEPT        icmp -- 0.0.0.0/0    0.0.0.0/0        icmptype 0
2  ACCEPT        icmp -- 0.0.0.0/0    0.0.0.0/0        icmptype 8
3  DROP          icmp -- 0.0.0.0/0    0.0.0.0/0        icmptype 8
4  ACCEPT        icmp -- 0.0.0.0/0    10.9.0.11        icmptype 8
Chain OUTPUT (policy ACCEPT)
num target      prot opt source      destination
```

- Outside hosts cannot ping internal hosts.

From 10.9.0.5

```
root@guoyuchen-hostA-10:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
^C
--- 192.168.60.5 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3071ms
```

- Outside hosts can ping the router.

From 10.9.0.5

```
root@guoyuchen-hostA-10:/# ping 10.9.0.11
PING 10.9.0.11 (10.9.0.11) 56(84) bytes of data.
64 bytes from 10.9.0.11: icmp_seq=1 ttl=64 time=0.058 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.127 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.109 ms
^C
--- 10.9.0.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2049ms
```

- Internal hosts can ping outside hosts.

From host1-192.168.60.5

```
root@guoyuchen-host1-192:/# ping 10.9.0.5
PING 10.9.0.5 (10.9.0.5) 56(84) bytes of data.
64 bytes from 10.9.0.5: icmp_seq=1 ttl=63 time=0.068 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.129 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.114 ms
^C
--- 10.9.0.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2068ms
rtt min/avg/max/mdev = 0.068/0.103/0.129/0.025 ms
```

- All other packets between the internal and external networks should be blocked.

telnet from 10.9.0.5 to 192.168.60.5 is blocked.

```
root@guoyuchen-hostA-10:/# telnet 192.168.60.5
Trying 192.168.60.5...
telnet: Unable to connect to remote host: Connection timed out
```

telnet from 192.168.60.5 to 10.9.0.5 is blocked.

```
root@guoyuchen-host1-192:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
```

telnet from 10.9.0.11 to 10.9.0.5 is accepted.

```
root@guoyuchen-seed-router:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
guoyuchen-hostA-10.9.0.5 login: dees
Password: Connection closed by foreign host.
```

ping from 10.9.0.5 to google.com is accepted.

```
root@guoyuchen-hostA-10:/# ping google.com
PING google.com (74.125.24.100) 56(84) bytes of data.
64 bytes from sf-in-f100.1e100.net (74.125.24.100): icmp_seq=1 ttl
=103 time=6.63 ms
64 bytes from sf-in-f100.1e100.net (74.125.24.100): icmp_seq=2 ttl
=101 time=7.05 ms
^C
--- google.com ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 6.628/6.838/7.048/0.210 ms
```

ping from 192.168.60.5 to 192.168.60.7 is accepted.

```
root@guoyuchen-host1-192:/# ping 192.168.60.7
PING 192.168.60.7 (192.168.60.7) 56(84) bytes of data.
64 bytes from 192.168.60.7: icmp_seq=1 ttl=64 time=0.148 ms
64 bytes from 192.168.60.7: icmp_seq=2 ttl=64 time=0.105 ms
^C
--- 192.168.60.7 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1012ms
rtt min/avg/max/mdev = 0.105/0.126/0.148/0.021 ms
```

## Task 2.C: Protecting Internal Servers

```
// Set default FORWARD policy to DROP. Since other traffic between internal and
```

external networks is prohibited.

```
iptables -P FORWARD DROP
```

// Outside hosts can only access the telnet server on 192.168.60.5.

```
iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 --dport 23 -j ACCEPT
```

```
iptables -A FORWARD -i eth1 -p tcp -s 192.168.60.5 --sport 23 -j ACCEPT
```

- Outside hosts can only access the telnet server on 192.168.60.5, not the other internal hosts.

From 10.9.0.5

```
root@guoyuchen-hostA-10:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```

```
root@guoyuchen-hostA-10:/# telnet 192.168.60.6
Trying 192.168.60.6...
telnet: Unable to connect to remote host: Connection timed out
```

```
root@guoyuchen-hostA-10:/# telnet 192.168.60.7
Trying 192.168.60.7...
telnet: Unable to connect to remote host: Connection timed out
```

- Internal hosts can access all the internal servers.

From 192.168.60.5

```
root@guoyuchen-host1-192:/# telnet 192.168.60.7
Trying 192.168.60.7...
Connected to 192.168.60.7.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```

```
root@guoyuchen-host1-192:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
```

- Internal hosts cannot access external servers.

From 192.168.60.5

```
root@guoyuchen-host1-192:/# telnet 10.9.0.5
Trying 10.9.0.5...
telnet: Unable to connect to remote host: Connection timed out
```

## Task 3: Connection Tracking and Stateful Firewall

### Task 3.A: Experiment with the Connection Tracking

#### ICMP experiment

- Protocol of the connection -> ICMP
- **timeout: how long the connection state is kept. -> 30 seconds**
- src: Source IP address -> 10.9.0.5
- dst: Destination IP address -> 192.168.60.5
- icmp-type: echo request (8)
- what we expect of return packets: the echo reply message from 192.168.60.5

```
root@guoyuchen-seed-router:/# conntrack -L
icmp      1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=71 src=192.168.
60.5 dst=10.9.0.5 type=0 code=0 id=71 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

#### UDP experiment

- Protocol of the connection -> UDP
- **timeout: how long the connection state is kept. -> 30 seconds**
- src: source IP address -> 10.9.0.5
- dst: destination IP address -> 192.168.60.5
- sport: source port -> 42465
- dport: destination port -> 9090
- [UNREPLIED]: there's no return traffic for this connection.
- what we expect of return packets: message from 192.168.60.5 at port 9090

```
root@guoyuchen-seed-router:/# conntrack -L
udp       17 29 src=10.9.0.5 dst=192.168.60.5 sport=42465 dport=9090 [UNREPLIED]
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=42465 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

#### TCP experiment

- Protocol of the connection -> TCP
- **timeout: how long the connection state is kept. -> 43200 seconds**
- state of the connection: ESTABLISHED
- src: source IP address -> 10.9.0.5
- dst: destination IP address -> 192.168.60.5
- sport: source port -> 44476



- dport: destination port -> 9090
- what we expect of return packets: message from 192.168.60.5 at port 9090

```
root@guoyuchen-seed-router:/# conntrack -L
tcp        6 431999 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44476 dport=9090
src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44476 [ASSURED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

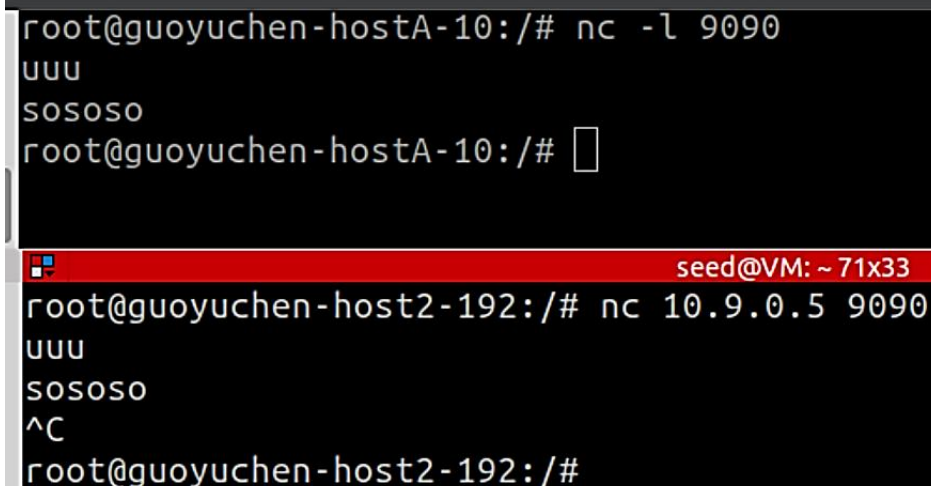
## Task 3.B: Setting Up a Stateful Firewall

```
// previous added rules
iptables -P FORWARD DROP
iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 --dport 23 -j ACCEPT
iptables -A FORWARD -i eth1 -p tcp -s 192.168.60.5 --sport 23 -j ACCEPT

// Newly added rules
// Allow existing connections
iptables -A FORWARD -p tcp -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT
// Allow internal hosts to connect to external hosts for the first time
iptables -A FORWARD -i eth1 -p tcp --syn -m conntrack --ctstate NEW -j ACCEPT
```

Internal hosts can visit external servers.

192.168.60.6 can connect to 10.9.0.5.



```
root@guoyuchen-hostA-10:/# nc -l 9090
uuu
sososo
root@guoyuchen-hostA-10:/#

seed@VM: ~ 71x33
root@guoyuchen-host2-192:/# nc 192.168.60.6 9090
uuu
sososo
^C
root@guoyuchen-host2-192:/#
```

To implement without Connection Tracking: Need to explicitly allow traffic for both outgoing and incoming packets based on the source and destination.

### Stateful Filtering (With Connection Tracking)

- **Advantages:**
  - More specific and fine-grained rules and policies can be enforced.
  - Can maintain state information about connections, which can be used to distinguish between legitimate,

established connections and unauthorized or malicious traffic, etc.

- **Disadvantages:**

- Consumes more system resources to maintain state information for each connection.
- It brings complexity to the firewall configuration, which makes managing and troubleshooting rules related to established connections and connection states more demanding.

#### Stateless Filtering (Without Connection Tracking)

- **Advantages:**

- Consumes fewer system resources because it doesn't maintain connection state information. Reduces the overhead.
- Simpler to configure and maintain.

- **Disadvantages:**

- Configuring rules for more complex setups can be limited since it does not inspect states of traffic.
- Maintaining the rules becomes harder as the number of allowed connections grows.

## Task 4: Limiting Network Traffic

1. with the second rule: the speed is normal at first, but getting slower after 5 packets transmitted. We see from the terminal that 76% of the packets are lost.

```
root@guoyuchen-hostA-10:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.204 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.152 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.156 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.133 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.287 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.172 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.203 ms
64 bytes from 192.168.60.5: icmp_seq=19 ttl=63 time=0.140 ms
64 bytes from 192.168.60.5: icmp_seq=25 ttl=63 time=0.340 ms
64 bytes from 192.168.60.5: icmp_seq=31 ttl=63 time=0.181 ms
64 bytes from 192.168.60.5: icmp_seq=37 ttl=63 time=0.073 ms
64 bytes from 192.168.60.5: icmp_seq=42 ttl=63 time=0.245 ms
64 bytes from 192.168.60.5: icmp_seq=48 ttl=63 time=0.212 ms
64 bytes from 192.168.60.5: icmp_seq=54 ttl=63 time=0.168 ms
64 bytes from 192.168.60.5: icmp_seq=60 ttl=63 time=0.109 ms
^C
--- 192.168.60.5 ping statistics ---
63 packets transmitted, 15 received, 76.1905% packet loss, time 64053ms
rtt min/avg/max/mdev = 0.073/0.185/0.340/0.065 ms
```

2. without the second rule: does not limit traffic.

```

root@guoyuchen-hostA-10:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
64 bytes from 192.168.60.5: icmp_seq=1 ttl=63 time=0.092 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.137 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.146 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.113 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.103 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.100 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.227 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.143 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.227 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.172 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.206 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.299 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.381 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.167 ms
64 bytes from 192.168.60.5: icmp_seq=15 ttl=63 time=0.079 ms
64 bytes from 192.168.60.5: icmp_seq=16 ttl=63 time=0.167 ms
64 bytes from 192.168.60.5: icmp_seq=17 ttl=63 time=0.225 ms
64 bytes from 192.168.60.5: icmp_seq=18 ttl=63 time=0.115 ms
64 bytes from 192.168.60.5: icmp_seq=19 ttl=63 time=0.096 ms
64 bytes from 192.168.60.5: icmp_seq=20 ttl=63 time=0.103 ms
64 bytes from 192.168.60.5: icmp_seq=21 ttl=63 time=0.095 ms
64 bytes from 192.168.60.5: icmp_seq=22 ttl=63 time=0.091 ms
64 bytes from 192.168.60.5: icmp_seq=23 ttl=63 time=0.095 ms
64 bytes from 192.168.60.5: icmp_seq=24 ttl=63 time=0.125 ms
64 bytes from 192.168.60.5: icmp_seq=25 ttl=63 time=0.087 ms
64 bytes from 192.168.60.5: icmp_seq=26 ttl=63 time=0.130 ms
64 bytes from 192.168.60.5: icmp_seq=27 ttl=63 time=0.165 ms
64 bytes from 192.168.60.5: icmp_seq=28 ttl=63 time=0.120 ms
64 bytes from 192.168.60.5: icmp_seq=29 ttl=63 time=0.133 ms
64 bytes from 192.168.60.5: icmp_seq=30 ttl=63 time=0.125 ms
64 bytes from 192.168.60.5: icmp_seq=31 ttl=63 time=0.169 ms
64 bytes from 192.168.60.5: icmp_seq=32 ttl=63 time=0.081 ms
^C
--- 192.168.60.5 ping statistics ---
32 packets transmitted, 32 received, 0% packet loss, time 32240ms
rtt min/avg/max/mdev = 0.079/0.147/0.381/0.066 ms

```

Thus the second rule is needed. Because the kernel will process the packages that are out of the limit according to the default policy, which is ACCEPT if we don't apply the second rule.

## Task 5: Load Balancing

### Using the nth mode (round-robin).

Redirect traffic at every 1/2/3 packages.

```

iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every
3 --packet 0 -j DNAT --to-destination 192.168.60.5:8080
iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every
2 --packet 0 -j DNAT --to-destination 192.168.60.6:8080
iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every
1 --packet 0 -j DNAT --to-destination 192.168.60.7:8080

```

```

root@guoyuchen-hostA-10:/# echo 0 | nc -u 10.9.0.11 8080
^C
root@guoyuchen-hostA-10:/# echo 1 | nc -u 10.9.0.11 8080
^C
root@guoyuchen-hostA-10:/# echo 2 | nc -u 10.9.0.11 8080
^C
root@guoyuchen-hostA-10:/# echo 3 | nc -u 10.9.0.11 8080
^C
root@guoyuchen-hostA-10:/# echo 4 | nc -u 10.9.0.11 8080
^C
root@guoyuchen-hostA-10:/# echo 5 | nc -u 10.9.0.11 8080
^C

```

10.9.0.5

```

root@guoyuchen-host1-192:/# nc -luk 8080
0
3

```

192.168.60.5

```

root@guoyuchen-host2-192:/# nc -luk 8080
1
4

```

192.168.60.6

```

root@guoyuchen-host3-192:/# nc -luk 8080
2
5

```

192.168.60.7

Thus each host gets equal amount of messages.

## Using the random mode.

Probability set to 0.33 to balance between 3 servers.

```

iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random
--probability 0.33 -j DNAT --to-destination 192.168.60.5:8080
iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random
--probability 0.33 -j DNAT --to-destination 192.168.60.6:8080
iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random
--probability 0.33 -j DNAT --to-destination 192.168.60.7:8080

```



```
root@guoyuchen-seed-router:/# iptables -t nat -L -n
Chain PREROUTING (policy ACCEPT)
target     prot opt source                destination              udp dpt:8080
DNAT       udp  --  0.0.0.0/0              0.0.0.0/0                udp dpt:8080
statistic mode random probability 0.330000000007 to:192.168.60.5:8080
DNAT       udp  --  0.0.0.0/0              0.0.0.0/0                udp dpt:8080
statistic mode random probability 0.330000000007 to:192.168.60.6:8080
DNAT       udp  --  0.0.0.0/0              0.0.0.0/0                udp dpt:8080
statistic mode random probability 0.330000000007 to:192.168.60.7:8080
```

```
root@guoyuchen-host1-192:/# nc -luk 8080
```

```
1
4
6
7
10
11
14
```

192.168.60.5

```
root@guoyuchen-host2-192:/# nc -luk 8080
```

```
2
12
15
```

192.168.60.6

```
root@guoyuchen-host3-192:/# nc -luk 8080
```

```
1
3
8
```

192.168.60.7

```
root@guoyuchen-seed-router:/# nc -luk 8080
```

```
0
5
9
13
```

10.9.0.11

Traffic is redistributed among 4 hosts.