Name – Harshit Agrawal SUID – 912538842

Task 1: Implementing a Simple Firewall

Task 1.A: Implement a Simple Kernel Module

```
[03/29/23]seed@VM:~/.../Labsetup$ dcup
Creating network "net-10.9.0.0" with the default driver
Creating network "net-192.168.60.0" with the default driver
WARNING: Found orphan containers (server-2-10.9.0.6, server-1-10.9.0.5, server-3
-10.9.0.7, host-192.168.50.6, server-4-10.9.0.8) for this project. If you remove
d or renamed this service in your compose file, you can run this command with th
e --remove-orphans flag to clean it up.
Creating host3-192.168.60.7 ... done
Creating hostA-10.9.0.5     ... done
Creating seed-router        ... done
Creating host2-192.168.60.6 ... done
Creating host1-192.168.60.5 ... done
Attaching to host2-192.168.60.6, hostA-10.9.0.5, host3-192.168.60.7, seed-router
, host1-192.168.60.5
host2-192.168.60.6 |  * Starting internet superserver inetd              [ OK ]
host3-192.168.60.7 |  * Starting internet superserver inetd              [ OK ]
hostA-10.9.0.5 |  * Starting internet superserver inetd                  [ OK ]
seed-router |  * Starting internet superserver inetd                     [ OK ]
host1-192.168.60.5 |  * Starting internet superserver inetd              [ OK ]
```

In kernel module folder, compile the files , kernel module using the make command.

```
[03/31/23]seed@VM:~/.../kernel_module$ ls
ıello.c   hello.mod    hello.mod.o  Makefile       Module.symvers
ıello.ko  hello.mod.c  hello.o      modules.order
[03/31/23]seed@VM:~/.../kernel_module$ sudo insmod hello.ko
[03/31/23]seed@VM:~/.../kernel_module$
```

```
[03/31/23]seed@VM:~/.../kernel_module$ sudo insmod hello.ko
[03/31/23]seed@VM:~/.../kernel_module$ lsmod | grep -i hello
hello                  16384  0
[03/31/23]seed@VM:~/.../kernel_module$ sudo rmmod hello
[03/31/23]seed@VM:~/.../kernel_module$ lsmod | grep -i hello
[03/31/23]seed@VM:~/.../kernel_module$
```

use of dmesg command

```
[03/31/23]seed@VM:~/.../kernel_module$ dmesg -k -w
[ 1670.670093] hello: module verification failed: signature and/or required key
missing - tainting kernel
[ 1670.686162] Hello World!
```

```
[ 2801.438521] br-3a97a7f9d0c4: port 4(veth340cc70) entered blocking state
[ 2801.438522] br-3a97a7f9d0c4: port 4(veth340cc70) entered forwarding state
[ 3002.362715] Bye-bye World!.
```

removing the hello.ko module and checking the message using dmesg:

```
[ 3896.641949] Hello World!
[ 3988.097526] Bye-bye World!.
[ 3998.522548] Hello World!
```

Task 1.B: Implement a Simple Firewall Using Netfilter

Makefile | seedFilter.c | seedPrint.c

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/udp.h>
#include <linux/icmp.h>
#include <linux/if_ether.h>
#include <linux/inet.h>


static struct nf_hook_ops hook1, hook2;


unsigned int blockUDP(void *priv, struct sk_buff *skb,
                      const struct nf_hook_state *state)
{
   struct iphdr *iph;
   struct udphdr *udph;

   u16  port   = 53; //DNS
   char ip[16] = "8.8.8.8";
   u32  ip_addr;

   if (!skb) return NF_ACCEPT;

   iph = ip_hdr(skb);
   // Convert the IPv4 address from dotted decimal to 32-bit binary
   in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

   if (iph->protocol == IPPROTO_UDP) {
       udph = udp_hdr(skb);
       if (iph->daddr == ip_addr && ntohs(udph->dest) == port){
           printk(KERN_WARNING "*** Dropping %pI4 (UDP), port %d\n",
  &(iph->daddr), port);
```

compiling the seedFileter.c file

```
[03/31/23]seed@VM:~/.../packet_filter$ ls
Makefile  seedFilter.c
[03/31/23]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Firewallab/Labs
etup/Files/packet_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedFilter.
o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedFilter.
mod.o
  LD [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedFilter.
ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[03/31/23]seed@VM:~/.../packet_filter$ ls
Makefile       Module.symvers  seedFilter.ko   seedFilter.mod.c  seedFilter.o
modules.order  seedFilter.c    seedFilter.mod  seedFilter.mod.o
[03/31/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
[03/31/23]seed@VM:~/.../packet_filter$ lsmod | grep -i seed
seedFilter             16384  0
[03/31/23]seed@VM:~/.../packet_filter$
```

```
[ 3896.641949] Hello World!
[ 3988.097526] Bye-bye World!.
[ 3998.522548] Hello World!
[ 8659.289214] Registering filters.
[ 8729.068094] *** LOCAL_OUT
[ 8729.068097]     10.0.2.4  --> 192.168.1.1 (UDP)
[ 8729.239185] *** LOCAL_OUT
[ 8729.239187]     10.0.2.4  --> 185.125.190.49 (TCP)
[ 8729.330247] *** LOCAL_OUT
[ 8729.330250]     10.0.2.4  --> 185.125.190.49 (TCP)
[ 8729.330448] *** LOCAL_OUT
[ 8729.330450]     10.0.2.4  --> 185.125.190.49 (TCP)
[ 8729.420815] *** LOCAL_OUT
[ 8729.420817]     10.0.2.4  --> 185.125.190.49 (TCP)
[ 8729.421224] *** LOCAL_OUT
[ 8729.421225]     10.0.2.4  --> 185.125.190.49 (TCP)
[ 8759.223289] *** LOCAL_OUT
[ 8759.223291]     10.0.2.4  --> 10.0.2.3 (UDP)
[ 8918.499671] *** LOCAL_OUT
[ 8918.499673]     10.0.2.4  --> 192.168.1.1 (UDP)
```

using the following command to generate UDP packets to 8.8.8.8, which is Google's DNS server.

```
odules.order  seedFilter.c    seedFilter.mod  seedFilter.mod.o
03/31/23]seed@VM:~/.../packet_filter$ sudo insmod seedFilter.ko
03/31/23]seed@VM:~/.../packet_filter$ lsmod | grep -i seed
eedFilter              16384  0
03/31/23]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

03/31/23]seed@VM:~/.../packet_filter$
```

```
[10829.164835]     10.0.2.4  --> 35.224.170.84 (TCP)
[10829.164922] *** LOCAL_OUT
[10829.164923]     10.0.2.4  --> 35.224.170.84 (TCP)
[10859.181379] *** LOCAL_OUT
[10859.181381]     10.0.2.4  --> 10.0.2.3 (UDP)
[10963.824141] *** LOCAL_OUT
[10963.824142]     127.0.0.1  --> 127.0.0.1 (UDP)
[10963.825031] *** LOCAL_OUT
[10963.825032]     10.0.2.4  --> 8.8.8.8 (UDP)
[10963.825037] *** Dropping 8.8.8.8 (UDP), port 53
[10968.833502] *** LOCAL_OUT
[10968.833504]     10.0.2.4  --> 8.8.8.8 (UDP)
[10968.833520] *** Dropping 8.8.8.8 (UDP), port 53
[10973.835621] *** LOCAL_OUT
[10973.835624]     10.0.2.4  --> 8.8.8.8 (UDP)
[10973.835643] *** Dropping 8.8.8.8 (UDP), port 53
[11018.437777] *** LOCAL_OUT
[11018.437778]     10.0.2.4  --> 192.168.1.1 (UDP)
```

2.

Changes in Makefile

```
#obj-m += seedFilter.o
obj-m += seedPrint.o
all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean

ins:
	sudo dmesg -C
	sudo insmod seedFilter.ko

rm:
	sudo rmmod seedFilter
```

Hooking the printInfo function to all of the netfilter hooks.

Makefile | seedFilter.c | seedPrint.c

```
74
75 int registerFilter(void) {
76    printk(KERN_INFO "SEEDPRINT: Registering filters.\n");
77
78         //NF_INET_PRE_ROUTING
79
80    hook1.hook = printInfo;
81    hook1.hooknum = NF_INET_PRE_ROUTING;
82    hook1.pf = PF_INET;
83    hook1.priority = NF_IP_PRI_FIRST;
84    nf_register_net_hook(&init_net, &hook1);
85
86
87         //NF_INET_LOCAL_IN
88    hook2.hook = printInfo;
89    hook2.hooknum = NF_INET_LOCAL_IN;
90    hook2.pf = PF_INET;
91    hook2.priority = NF_IP_PRI_FIRST;
92    nf_register_net_hook(&init_net, &hook2);
93
94
95         //NF_INET_FORWARD
96    hook3.hook = printInfo;
97    hook3.hooknum = NF_INET_FORWARD;
98    hook3.pf = PF_INET;
99    hook3.priority = NF_IP_PRI_FIRST;
100   nf_register_net_hook(&init_net, &hook3);
101
102
103        //NF_INET_LOCAL_OUT
104   hook4.hook = printInfo;
105   hook4.hooknum = NF_INET_LOCAL_OUT;
106   hook4.pf = PF_INET;
107   hook4.priority = NF_IP_PRI_FIRST;
108   nf_register_net_hook(&init_net, &hook4);
```

Makefile | seedFilter.c | seedPrint.c

```
2
3         //NF_INET_LOCAL_OUT
4    hook4.hook = printInfo;
5    hook4.hooknum = NF_INET_LOCAL_OUT;
6    hook4.pf = PF_INET;
7    hook4.priority = NF_IP_PRI_FIRST;
8    nf_register_net_hook(&init_net, &hook4);
9
0
1         //NF_INET_POST_ROUTING
2
3    hook5.hook = printInfo;
4    hook5.hooknum = NF_INET_POST_ROUTING;
5    hook5.pf = PF_INET;
6    hook5.priority = NF_IP_PRI_FIRST;
7    nf_register_net_hook(&init_net, &hook5);
8
9
0
1    return 0;
2 }
3
4 void removeFilter(void) {
5    printk(KERN_INFO "SEEDFILTER: The filters are being removed.\n");
6    nf_unregister_net_hook(&init_net, &hook1);
7    nf_unregister_net_hook(&init_net, &hook2);
8    nf_unregister_net_hook(&init_net, &hook3);
9    nf_unregister_net_hook(&init_net, &hook4);
0    nf_unregister_net_hook(&init_net, &hook5);
1 }
2
3 module_init(registerFilter);
4 module_exit(removeFilter);
5
6 MODULE_LICENSE("GPL");
7
```

For this task we create a new file named seedPrint and add its executable kernel to the Maekefile:

```
[03/31/23]seed@VM:~/.../packet_filter$ la
.gitignore     Module.symvers  seedPrint.ko       seedPrint.mod.c     .seedPrint.mod.o.cmd
Makefile       seedFilter.c    .seedPrint.ko.cmd  .seedPrint.mod.cmd  seedPrint.o
modules.order  seedPrint.c     seedPrint.mod      seedPrint.mod.o     .seedPrint.o.cmd
[03/31/23]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Firewallab/Labsetup/Files/p
acket_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedPrint.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedPrint.mod.o
  LD [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedPrint.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[03/31/23]seed@VM:~/.../packet_filter$ sudo insmod seedPrint.ko
[03/31/23]seed@VM:~/.../packet_filter$ sudo rmmod seedPrint
[03/31/23]seed@VM:~/.../packet_filter$
```

Now we insert the kernel module and check for the message:

Below is the dmesg output

```
[12652.012573]     127.0.0.1  --> 127.0.0.1 (UDP)
[12652.012575] *** LOCAL_IN
[12652.012575]     127.0.0.1  --> 127.0.0.1 (UDP)
[12652.013032] *** LOCAL_OUT
[12652.013033]     10.0.2.4  --> 8.8.8.8 (UDP)
[12652.013037] *** POST_ROUTING
[12652.013037]     10.0.2.4  --> 8.8.8.8 (UDP)
[12652.033559] *** PRE_ROUTING
[12652.036818]     8.8.8.8  --> 10.0.2.4 (UDP)
[12652.038455] *** LOCAL_IN
[12652.041242]     8.8.8.8  --> 10.0.2.4 (UDP)
[12659.289081] *** LOCAL_OUT
[12659.289082]     10.0.2.4  --> 10.0.2.3 (UDP)
[12659.289099] *** POST_ROUTING
[12659.289099]     10.0.2.4  --> 10.0.2.3 (UDP)
[12659.292220] *** PRE_ROUTING
[12659.292594]     10.0.2.3  --> 10.0.2.4 (UDP)
[12659.292744] *** LOCAL_IN
[12659.292882]     10.0.2.3  --> 10.0.2.4 (UDP)
[12803.499010] SEEDFILTER: The filters are being removed.
```

We see that LOCAL_OUT, LOCAL_IN, POST_ROUTING and PRE_ROUTING functions were invoked as the UDP packets were generated except the FORWARD function.

```
[12603.809003] SEEDPRINT: Registering filters.
[12649.124603] *** LOCAL_OUT
[12649.124605]     10.0.2.4  --> 192.168.1.1 (UDP)
[12649.124617] *** POST_ROUTING
[12649.124617]     10.0.2.4  --> 192.168.1.1 (UDP)
[12649.210764] *** PRE_ROUTING
[12649.210990]     192.168.1.1  --> 10.0.2.4 (UDP)
[12649.211110] *** LOCAL_IN
[12649.211242]     192.168.1.1  --> 10.0.2.4 (UDP)
[12649.212375] *** LOCAL_OUT
[12649.212377]     10.0.2.4  --> 35.232.111.17 (TCP)
[12649.212387] *** POST_ROUTING
[12649.212388]     10.0.2.4  --> 35.232.111.17 (TCP)
[12649.246054] *** PRE_ROUTING
[12649.246255]     35.232.111.17  --> 10.0.2.4 (TCP)
[12649.246442] *** LOCAL_IN
[12649.246575]     35.232.111.17  --> 10.0.2.4 (TCP)
[12649.246942] *** LOCAL_OUT
[12649.247047]     10.0.2.4  --> 35.232.111.17 (TCP)
[12649.247186] *** POST_ROUTING
[12649.247372]     10.0.2.4  --> 35.232.111.17 (TCP)
[12649.247601] *** LOCAL_OUT
[12649.247602]     10.0.2.4  --> 35.232.111.17 (TCP)
[12649.247605] *** POST_ROUTING
```

And we make use of the dig command to check the generated UDP packets

```
[03/31/23]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
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 57068
;; flags: qr rd ra ad; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 512
;; QUESTION SECTION:
;www.example.com.               IN      A

;; ANSWER SECTION:
www.example.com.        17886   IN      A       93.184.216.34

;; Query time: 31 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Mar 31 16:49:53 EDT 2023
;; MSG SIZE  rcvd: 60

[03/31/23]seed@VM:~/.../packet_filter$
```

```
[03/31/23]seed@VM:~/.../packet_filter$ docksh hostA-10.9.0.5
root@b211cda3468f:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
64 bytes from 10.9.0.1: icmp_seq=1 ttl=64 time=0.143 ms
64 bytes from 10.9.0.1: icmp_seq=2 ttl=64 time=0.061 ms
64 bytes from 10.9.0.1: icmp_seq=3 ttl=64 time=0.070 ms
64 bytes from 10.9.0.1: icmp_seq=4 ttl=64 time=0.068 ms
^C
--- 10.9.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3080ms
rtt min/avg/max/mdev = 0.061/0.085/0.143/0.033 ms
root@b211cda3468f:/# telnet 10.9.0.1
Trying 10.9.0.1...
Connected to 10.9.0.1.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
VM login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

3.

In this task we will create another new file named seedBlock as follows

We two separate functions: 1. preventing other computers to ping the VM 2. preventing other computers to telnet into the VM

Makefile | seedBlock.c

```c
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/netfilter.h>
#include <linux/netfilter_ipv4.h>
#include <linux/ip.h>
#include <linux/tcp.h>
#include <linux/udp.h>
#include <linux/icmp.h>
#include <linux/if_ether.h>
#include <linux/inet.h>


static struct nf_hook_ops hook1, hook2, hook3, hook4;

// block udp for 8.8.8.8:53
unsigned int blockUDP(void *priv, struct sk_buff *skb,
                      const struct nf_hook_state *state)
{
   struct iphdr *iph;
   struct udphdr *udph;

   u16  port   = 53; //DNS
   char ip[16] = "8.8.8.8";
   u32  ip_addr;

   if (!skb) return NF_ACCEPT;

   iph = ip_hdr(skb);
   // Convert the IPv4 address from dotted decimal to 32-bit binary
   in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

   if (iph->protocol == IPPROTO_UDP) {
       udph = udp_hdr(skb);
       if (iph->daddr == ip_addr && ntohs(udph->dest) == port){
           printk(KERN_WARNING "*** Dropping %pI4 (UDP), port %d\n",
  &(iph->daddr), port);
```

C | Tab Width: 8 | Ln 89, Col 56 | INS

```
4        if (iph->daddr == ip_addr && ntohs(udph->dest) == port){
5             printk(KERN_WARNING "*** Dropping %pI4 (UDP), port %d\n",
 &(iph->daddr), port);
6             return NF_DROP;
7         }
8    }
9    return NF_ACCEPT;
0}
1
2// block ping to vm: 10.9.0.1
3unsigned int blockICMP(void *priv, struct sk_buff *skb,
4                       const struct nf_hook_state *state)
5{
6    struct iphdr *iph;
7    struct icmphdr *icmph;
8
9    //u16  port   = 53; //DNS
0    char ip[16] = "10.9.0.1";
1    u32  ip_addr;
2
3    if (!skb) return NF_ACCEPT;
4
5    iph = ip_hdr(skb);
6    // Convert the IPv4 address from dotted decimal to 32-bit binary
7    in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);
8
9    if (iph->protocol == IPPROTO_ICMP) {
0        icmph = icmp_hdr(skb);
1        if (iph->daddr == ip_addr && icmph->type == ICMP_ECHO){
2             printk(KERN_WARNING "*** Dropping %pI4 (ICMP)\n", &(iph-
 >daddr));
3             return NF_DROP;
4         }
5    }
6    return NF_ACCEPT;
7}
```

```
    }
    return NF_ACCEPT;
}

// blocking telnet to vm: 10.9.0.1:23
unsigned int blockTelnet(void *priv, struct sk_buff *skb,
                         const struct nf_hook_state *state)
{
    struct iphdr *iph;
    struct tcphdr *tcph;

    u16  port   = 23; //telnet
    char ip[16] = "10.9.0.1";
    u32  ip_addr;

    if (!skb) return NF_ACCEPT;

    iph = ip_hdr(skb);
    // Convert the IPv4 address from dotted decimal to 32-bit binary
    in4_pton(ip, -1, (u8 *)&ip_addr, '\0', NULL);

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


unsigned int printInfo(void *priv, struct sk_buff *skb,
                const struct nf_hook_state *state)
{
```

```
7        tcph = tcp_hdr(skb);
8        if (iph->daddr == ip_addr && ntohs(tcph->dest) == port){
9            printk(KERN_WARNING "*** Dropping %pI4 (TCP), port %d\n",
 &(iph->daddr), port);
0            return NF_DROP;
1        }
2    }
3    return NF_ACCEPT;
4}
5
6
7unsigned int printInfo(void *priv, struct sk_buff *skb,
8                const struct nf_hook_state *state)
9{
0    struct iphdr *iph;
1    char *hook;
2    char *protocol;
3
4    switch (state->hook){
5      case NF_INET_LOCAL_IN:     hook = "LOCAL_IN";     break;
6      case NF_INET_LOCAL_OUT:    hook = "LOCAL_OUT";    break;
7      case NF_INET_PRE_ROUTING:  hook = "PRE_ROUTING";  break;
8      case NF_INET_POST_ROUTING: hook = "POST_ROUTING"; break;
9      case NF_INET_FORWARD:      hook = "FORWARD";      break;
0      default:                   hook = "IMPOSSIBLE";   break;
1    }
2    printk(KERN_INFO "*** %s\n", hook); // Print out the hook info
3
4    iph = ip_hdr(skb);
5    switch (iph->protocol){
6      case IPPROTO_UDP:  protocol = "UDP";   break;
7      case IPPROTO_TCP:  protocol = "TCP";   break;
8      case IPPROTO_ICMP: protocol = "ICMP";  break;
9      default:           protocol = "OTHER"; break;
0
1    }
```

```
123    printk(KERN_INFO "    %pI4  --> %pI4 (%s)\n",
124                     &(iph->saddr), &(iph->daddr), protocol);
125
126    return NF_ACCEPT;
127 }
128
129
130 int registerFilter(void) {
131    printk(KERN_INFO "seedBlock: Registering filters.\n");
132
133    hook1.hook = printInfo;
134    hook1.hooknum = NF_INET_LOCAL_OUT;
135    hook1.pf = PF_INET;
136    hook1.priority = NF_IP_PRI_FIRST;
137    nf_register_net_hook(&init_net, &hook1);
138
139    hook2.hook = blockUDP;
140    hook2.hooknum = NF_INET_POST_ROUTING;
141    hook2.pf = PF_INET;
142    hook2.priority = NF_IP_PRI_FIRST;
143    nf_register_net_hook(&init_net, &hook2);
144
145    hook3.hook = blockICMP;
146    hook3.hooknum = NF_INET_PRE_ROUTING;
147    hook3.pf = PF_INET;
148    hook3.priority = NF_IP_PRI_FIRST;
149    nf_register_net_hook(&init_net, &hook3);
150
151    hook4.hook = blockTelnet;
152    hook4.hooknum = NF_INET_PRE_ROUTING;
153    hook4.pf = PF_INET;
154    hook4.priority = NF_IP_PRI_FIRST;
155    nf_register_net_hook(&init_net, &hook4);
156
157    return 0;
158 }
```

Changes in Makefile:

```
#obj-m += seedFilter.o
#obj-m += seedPrint.o
obj-m += seedBlock.o
all:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules

clean:
	make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean

ins:
	sudo dmesg -C
	sudo insmod seedFilter.ko

rm:
	sudo rmmod seedFilter
```

Now, making our module and inserting in the kernel

```
[03/31/23]seed@VM:~/.../packet_filter$ sudo rmmod seedBlock.ko
[03/31/23]seed@VM:~/.../packet_filter$ make
make -C /lib/modules/5.4.0-54-generic/build M=/home/seed/Desktop/Firewallab/Labsetup/Fil
acket_filter modules
make[1]: Entering directory '/usr/src/linux-headers-5.4.0-54-generic'
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedBlock.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedBlock.mod.o
  LD [M]  /home/seed/Desktop/Firewallab/Labsetup/Files/packet_filter/seedBlock.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.4.0-54-generic'
[03/31/23]seed@VM:~/.../packet_filter$ sudo insmod seedBlock.ko
[03/31/23]seed@VM:~/.../packet_filter$ sudo rmmod seedBlock.ko
[03/31/23]seed@VM:~/.../packet_filter$ ls
Makefile       Module.symvers  seedBlock.ko   seedBlock.mod.c  seedBlock.o   seedPrint.c
modules.order  seedBlock.c     seedBlock.mod  seedBlock.mod.o  seedFilter.c
[03/31/23]seed@VM:~/.../packet_filter$
```

pinging into the VM and see that the UDP and ICMP packets are getting dropped:

```
^C
root@b211cda3468f:/# ping 10.9.0.1
PING 10.9.0.1 (10.9.0.1) 56(84) bytes of data.
^C
--- 10.9.0.1 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6133ms

root@b211cda3468f:/# telnet 10.9.0.1
Trying 10.9.0.1...
^C
```

```
[17632.030922] seedBlock: The filters are being removed.
[17647.104459] seedBlock: Registering filters.
[17656.607487] *** Dropping 10.9.0.1 (ICMP)
[17657.634108] *** Dropping 10.9.0.1 (ICMP)
[17658.645670] *** Dropping 10.9.0.1 (ICMP)
[17659.669044] *** Dropping 10.9.0.1 (ICMP)
[17660.693014] *** Dropping 10.9.0.1 (ICMP)
[17661.719644] *** Dropping 10.9.0.1 (ICMP)
[17662.741548] *** Dropping 10.9.0.1 (ICMP)
[17665.283102] *** Dropping 10.9.0.1 (TCP), port 23
[17666.295011] *** Dropping 10.9.0.1 (TCP), port 23
[17668.311842] *** Dropping 10.9.0.1 (TCP), port 23
[17672.407104] *** Dropping 10.9.0.1 (TCP), port 23
[17707.046540] *** LOCAL_OUT
[17707.046542]     127.0.0.1  --> 224.0.0.251 (UDP)
[17707.046690] *** LOCAL_OUT
[17707.046691]     10.0.2.4  --> 224.0.0.251 (UDP)
[17707.046743] *** LOCAL_OUT
[17707.046744]     10.12.0.1  --> 224.0.0.251 (UDP)
[17707.046759] *** LOCAL_OUT
[17707.046760]     10.104.0.1  --> 224.0.0.251 (UDP)
[17707.046772] *** LOCAL_OUT
```

Task 2: Experimenting with Stateless Firewall Rules Task

2.A: Protecting the Router

We can see that the two interfaces have two IPs. Eth0 has ip 10.9.0.11 while Eth1 has ip 192.168.60.11.

Pinging the interfaces eth0 and eth1

```
root@b211cda3468f:/# 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=0.127 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.061 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.063 ms
^C
--- 10.9.0.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2048ms
rtt min/avg/max/mdev = 0.061/0.083/0.127/0.030 ms
root@b211cda3468f:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
64 bytes from 192.168.60.11: icmp_seq=1 ttl=64 time=0.072 ms
64 bytes from 192.168.60.11: icmp_seq=2 ttl=64 time=0.065 ms
64 bytes from 192.168.60.11: icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from 192.168.60.11: icmp_seq=4 ttl=64 time=0.072 ms
64 bytes from 192.168.60.11: icmp_seq=5 ttl=64 time=0.067 ms
^C64 bytes from 192.168.60.11: icmp_seq=6 ttl=64 time=0.079 ms
64 bytes from 192.168.60.11: icmp_seq=7 ttl=64 time=0.207 ms
^C
--- 192.168.60.11 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6152ms
rtt min/avg/max/mdev = 0.065/0.089/0.207/0.048 ms
root@b211cda3468f:/# ping seed-router
```

We can also use telnet into 10.9.0.11 succesfully.

```
7 packets transmitted, 7 received, 0% packet loss, time 6152ms
rtt min/avg/max/mdev = 0.065/0.089/0.207/0.048 ms
root@b211cda3468f:/# ping seed-router
PING seed-router (10.9.0.11) 56(84) bytes of data.
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=1 ttl=64 time=0.046 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=2 ttl=64 time=0.063 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=3 ttl=64 time=0.062 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=4 ttl=64 time=0.096 ms
^C
--- seed-router ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3042ms
rtt min/avg/max/mdev = 0.046/0.066/0.096/0.018 ms
root@b211cda3468f:/# telnet 10.9.0.11
Trying 10.9.0.11...
Connected to 10.9.0.11.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
f3d8b897942d login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

check the current status of iptables:

```
root@f3d8b897942d:/# iptables -t filter -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
root@f3d8b897942d:/# iptables -t filter -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@f3d8b897942d:/# iptables -P OUTPUT DROP
root@f3d8b897942d:/# iptables -P INPUT DROP
```

set up the rules for preventing outside machines from accessing to router machine

```
root@f3d8b897942d:/# iptables -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
root@f3d8b897942d:/# iptables -P OUTPUT DROP
root@f3d8b897942d:/# iptables -P INPUT DROP
root@f3d8b897942d:/# iptables -t filter -L -n
Chain INPUT (policy DROP)
target     prot opt source               destination
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 0

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy DROP)
target     prot opt source               destination
root@f3d8b897942d:/#
```

testing the rules by pinging and telnet which are denied

```
Connection closed by foreign host.
root@b211cda3468f:/# ping 192.168.60.11
PING 192.168.60.11 (192.168.60.11) 56(84) bytes of data.
^C
--- 192.168.60.11 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3066ms

root@b211cda3468f:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
root@b211cda3468f:/#
```

restoring the filter table to its original state

```
root@f3d8b897942d:/# iptables -F
root@f3d8b897942d:/# iptables -P OUTPUT ACCEPT
root@f3d8b897942d:/# iptables -P INPUT ACCEPT
root@f3d8b897942d:/# iptables -t filter -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/#
```

Task 2.B: Protecting the Internal Network

```
[03/31/23]seed@VM:~/.../Labsetup$ iptables -p icmp -h
iptables v1.8.4

Usage: iptables -[ACD] chain rule-specification [options]
       iptables -I chain [rulenum] rule-specification [options]
       iptables -R chain rulenum rule-specification [options]
       iptables -D chain rulenum [options]
       iptables -[LS] [chain [rulenum]] [options]
       iptables -[FZ] [chain] [options]
       iptables -[NX] chain
       iptables -E old-chain-name new-chain-name
       iptables -P chain target [options]
       iptables -h (print this help information)

Commands:
Either long or short options are allowed.
  --append  -A chain            Append to chain
  --check   -C chain            Check for the existence of a rule
  --delete  -D chain            Delete matching rule from chain
  --delete  -D chain rulenum
                                Delete rule rulenum (1 = first) from chain
  --insert  -I chain [rulenum]
                                Insert in chain as rulenum (default 1=first)
```

checking the status of iptables before setting the rule for dropping ICMP echo requests and the table status

```
root@f3d8b897942d:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP
root@f3d8b897942d:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/#
```

We can see that the outside hosts will not be able to ping the inside hosts because there will be a drop of the packets-

outside host can ping the router but outside host cannot ping the internal host as can seen below.

```
root@b211cda3468f:/# telnet 10.9.0.11
Trying 10.9.0.11...
^C
root@b211cda3468f:/# 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=0.162 ms
64 bytes from 10.9.0.11: icmp_seq=2 ttl=64 time=0.076 ms
64 bytes from 10.9.0.11: icmp_seq=3 ttl=64 time=0.075 ms
^C
--- 10.9.0.11 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2046ms
rtt min/avg/max/mdev = 0.075/0.104/0.162/0.040 ms
root@b211cda3468f:/# 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 ---
10 packets transmitted, 0 received, 100% packet loss, time 9195ms

root@b211cda3468f:/#
```

internal host can ping the outside hosts:

```
[03/31/23]seed@VM:~/.../packet_filter$ docksh host1-192.168.60.5
root@06541b30cbd0:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@06541b30cbd0:/# 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=0.122 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.081 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.092 ms
^C
--- 10.9.0.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2036ms
rtt min/avg/max/mdev = 0.081/0.098/0.122/0.017 ms
root@06541b30cbd0:/#
```

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

For this we need to generate a rule which accepts the packets sent outside from the internal host and vice versa as follows-

```
root@f3d8b897942d:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP
root@f3d8b897942d:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/# iptables -A FORWARD -i eth1 -p icmp --icmp-type echo-request -j ACCEPT
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-reply -j ACCEPT
root@f3d8b897942d:/# iptables -P FORWARD DROP
root@f3d8b897942d:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy DROP)
target     prot opt source               destination
DROP       icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 8
ACCEPT     icmp --  0.0.0.0/0            0.0.0.0/0            icmptype 0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@f3d8b897942d:/#
```

We can see the iptables as follows:

```
root@f3d8b897942d:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
   10   840 DROP       icmp --  eth0   *       0.0.0.0/0            0.0.0.0/0           icm
ptype 8
    0     0 ACCEPT     icmp --  eth1   *       0.0.0.0/0            0.0.0.0/0           icm
ptype 8
    0     0 ACCEPT     icmp --  eth0   *       0.0.0.0/0            0.0.0.0/0           icm
ptype 0

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@f3d8b897942d:/#
```

Now, testing by pinging the external hosts with internal hosts and vice versa:

```
root@b211cda3468f:/# 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 ---
10 packets transmitted, 0 received, 100% packet loss, time 9195ms

root@b211cda3468f:/# 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 ---
32 packets transmitted, 0 received, 100% packet loss, time 31799ms

root@b211cda3468f:/# telnet 192.168.60.5
Trying 192.168.60.5...
```

It works for the internal host to external host, for other connection requests we make use of telnet ans we see that the connection did not establish.

```
[03/31/23]seed@VM:~/.../packet_filter$ docksh host1-192.168.60.5
root@06541b30cbd0:/# ls
bin   dev  home  lib32  libx32  mnt  proc  run   srv  tmp  var
boot  etc  lib   lib64  media   opt  root  sbin  sys  usr
root@06541b30cbd0:/# 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=0.122 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.081 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.092 ms
^C
--- 10.9.0.5 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2036ms
rtt min/avg/max/mdev = 0.081/0.098/0.122/0.017 ms
root@06541b30cbd0:/# 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=0.150 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.100 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.087 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.113 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=63 time=0.103 ms
64 bytes from 10.9.0.5: icmp_seq=6 ttl=63 time=0.086 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=63 time=0.126 ms
64 bytes from 10.9.0.5: icmp_seq=8 ttl=63 time=0.086 ms
64 bytes from 10.9.0.5: icmp_seq=9 ttl=63 time=0.083 ms
64 bytes from 10.9.0.5: icmp_seq=10 ttl=63 time=0.077 ms
^C
--- 10.9.0.5 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9224ms
rtt min/avg/max/mdev = 0.077/0.101/0.150/0.021 ms
root@06541b30cbd0:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@06541b30cbd0:/#
```

We see that the telnet connection from internal to external and vice versa both doesn't work at all which means our firewall is working as expected.

Restoring the filter table to its original state:

```
root@f3d8b897942d:/# iptables -F
root@f3d8b897942d:/# iptables -P FORWARD ACCEPT
root@f3d8b897942d:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

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

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@f3d8b897942d:/#
```

Now, checking the telnet connection for all the machines-

```
root@b211cda3468f:/# telnet 192.168.60.5
Trying 192.168.60.5...
^C
root@b211cda3468f:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
seed
Ubuntu 20.04.1 LTS
seed
06541b30cbd0 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

```
root@b211cda3468f:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
d08211aced63 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content th
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
```

```
root@b211cda3468f:/# 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@06541b30cbd0:/# telnet 10.9.0.5
Trying 10.9.0.5...
^C
root@06541b30cbd0:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
b211cda3468f login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
```

Task 2.C: Protecting Internal Servers

```
root@f3d8b897942d:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

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

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p tcp --sport 5000 -j ACCEPT
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 -dport 5000 -j ACCEP
T
iptables v1.8.4 (legacy): multiple -d flags not allowed
Try `iptables -h' or 'iptables --help' for more information.
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -p tcp -d 192.168.60.5 --dport 5000 -j ACCE
PT
root@f3d8b897942d:/# iptables -A FORWARD -i eth0 -d 192.168.60.5 -p tcp --dport 23 -j ACCEPT
root@f3d8b897942d:/# iptables -A FORWARD -i eth1 -s 192.168.60.5 -p tcp --sport 23 -j ACCEPT
root@f3d8b897942d:/# iptables -P FORWARD DROP
root@f3d8b897942d:/# iptables -L -n -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination

Chain FORWARD (policy DROP 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            0.0.0.0/0            tcp
 spt:5000
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            192.168.60.5         tcp
 dpt:5000
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            192.168.60.5         tcp
 dpt:23
    0     0 ACCEPT     tcp  --  eth1   *       192.168.60.5         0.0.0.0/0            tcp
 spt:23

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
root@f3d8b897942d:/#
```

Eth0 accepts the requests from outside to inside on the dest IP 192.168.60.5 and dport 23 only while Eth1 does the same but from inside to outside.

third rule says to drop any other packets not intended to the IP or port specified in the above rules.

Now, telnet from external server to internal host such as 192.168.60.6 or 192.168.60.7 which does not work but only to 192.168.60.5 will work as we have specified in our rules.

```
root@06541b30cbd0:/# telnet 192.168.60.6
Trying 192.168.60.6...
Connected to 192.168.60.6.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
d08211aced63 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54
```

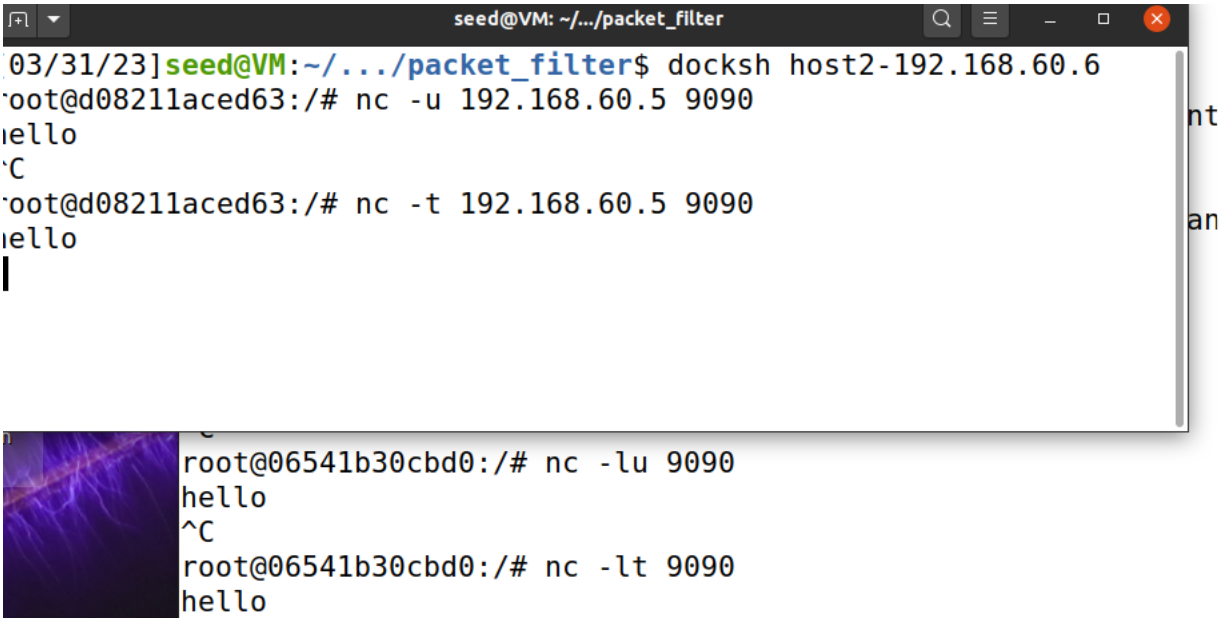
```
root@06541b30cbd0:/# telnet 192.168.60.7
Trying 192.168.60.7...
Connected to 192.168.60.7.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
41e1051d65e3 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage
Connection closed by foreign host.
root@b211cda3468f:/# telnet 192.168.60.7
Trying 192.168.60.7...
^C
root@b211cda3468f:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
root@b211cda3468f:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
06541b30cbd0 login: seed
Password:
Connection closed by foreign host.
root@06541b30cbd0:/# nc -lt 9090
^C
```

Trying external host to connect to any internal host on a port number which do not work as expected:

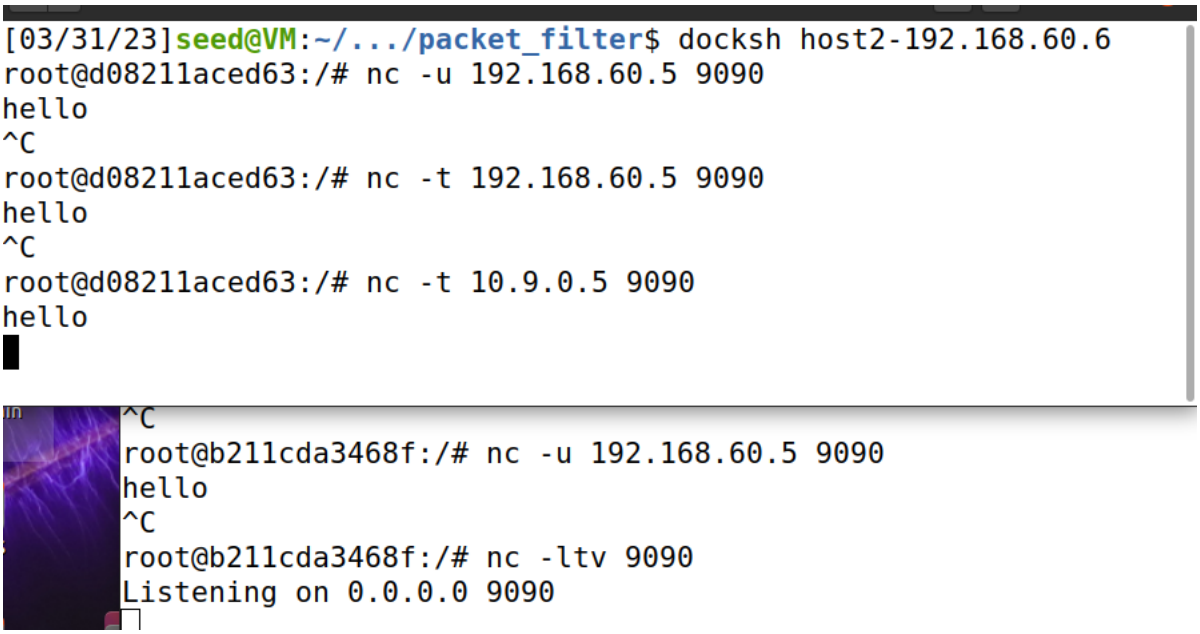
```
Connection closed by foreign host.
root@b211cda3468f:/# nc -lt 9090
^C
root@b211cda3468f:/# nc 192.168.60.5 9090
hello
^C
root@b211cda3468f:/# nc -u 192.168.60.5 9090
hello
^C
```

We didn’t receive any connection.

Now we try to connect from an internal host to access the internal server and we see that it connects successfully:



Now, the internal host try to connect to external server which is not possible:



Task 3: Connection Tracking and Stateful Firewall

Task 3.A: Experiment with the Connection Tracking ICMP experiment:

ICMP Experiment:

In the ICMP connection type we ping the 192.168.60.5 in the background to run continuously.

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh seed-router
root@57b7865b5245:/# conntrack -L
conntrack v1.4.5 (conntrack-tools): 0 flow entries have been shown.
root@57b7865b5245:/# ping 192.268.60.5 &
[1] 31
root@57b7865b5245:/# ping: 192.268.60.5: Name or service not known
^C
[1]+  Exit 2                  ping 192.268.60.5
root@57b7865b5245:/# ping 192.168.60.5 &
[1] 32
root@57b7865b5245:/# 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=64 time=0.125 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=64 time=0.094 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=64 time=0.077 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=64 time=0.132 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=64 time=0.091 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=64 time=0.066 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=64 time=0.179 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=64 time=0.145 ms
^C
root@57b7865b5245:/# 64 bytes from 192.168.60.5: icmp_seq=9 ttl=64 time=0.0
64 bytes from 192.168.60.5: icmp_seq=10 ttl=64 time=0.148 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=64 time=0.094 ms
```

In the server router, we can see the connection tracking information using the following command:

For the ICMP, the connection stays for about 5 to 6 seconds for each packet sent.

```
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@57b7865b5245:/# ping 192.168.60.5 &>/dev/null &
[1] 41
root@57b7865b5245:/# conntrack -L
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=41 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=41 mark=0 use=1
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
```

UDP experiment:

For this experiment, we need to create a netcat UDP server as follows:

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh hostA-10.9.0.5
root@1165e7b71b2c:/# nc -u 192.168.60.5 9090
hello
```

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh host1-192.168.60.5
root@e753f8b4b199:/# nc -lu 9090
hello
^C
root@e753f8b4b199:/#
```

By this, can say that the UDP connection stays for around 6-7 seconds for each packet connection tracking. This is done using the conntrack -L command:

```
root@57b7865b5245:/# conntrack -L
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@57b7865b5245:/# jobs
root@57b7865b5245:/# conntrack -L
udp      17 6 src=10.9.0.5 dst=192.168.60.5 sport=53292 dport=9090 [UNREPLIED] s
rc=192.168.60.5 dst=10.9.0.5 sport=9090 dport=53292 mark=0 use=1
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@57b7865b5245:/# conntrack -L
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@57b7865b5245:/# conntrack -L
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
root@57b7865b5245:/#
```

TCP experiment:

For this experiment we need a TCP server on netcat

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh hostA-10.9.0.5
root@1165e7b71b2c:/# nc -u 192.168.60.5 9090
hello
^C
root@1165e7b71b2c:/# nc 192.168.60.5 9090
hwllo
^C
root@1165e7b71b2c:/# ^C
root@1165e7b71b2c:/#
```

the connection state stays for almost around one to two minutes before being timed out.

```
root@57b7865b5245:/# conntrack -L
tcp      6 431988 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44112 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44112 [ASSURED] mark=0 use=1
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@57b7865b5245:/# conntrack -L
tcp      6 431990 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44112 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44112 [ASSURED] mark=0 use=1
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
.60.5 dst=192.168.60.11 type=0 code=0 id=32 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 2 flow entries have been shown.
root@57b7865b5245:/# conntrack -L
tcp      6 431951 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=44112 dport=90
90 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=44112 [ASSURED] mark=0 use=1
icmp     1 29 src=192.168.60.11 dst=192.168.60.5 type=8 code=0 id=32 src=192.168
```

We can see the reducing time indicating the time remaining for the connection before getting timed out.

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh host1-192.168.60.5
root@e753f8b4b199:/# nc -lu 9090
hello
^C
root@e753f8b4b199:/# nc -l 9090
hwllo
root@e753f8b4b199:/# ^C
root@e753f8b4b199:/#
```

Task 3.B: Setting Up a Stateful Firewall

Rewriting the rules for the following task using these commands:

We can check the state of our iptables as shown below:

```
Try `iptables -h' or 'iptables --help' for more information.
root@57b7865b5245:/# iptables -A FORWARD -p tcp -i eth0 --dport 8080 --syn \
> -m conntrack --ctstate NEW -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -p tcp -i eth0 --dport 8080 --syn -m co
nntrack --ctstate NEW -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -i eth1 -p tcp --syn -j -m conntrack --
ctstate NEW -j ACCEPT
Bad argument `conntrack'
Try `iptables -h' or 'iptables --help' for more information.
root@57b7865b5245:/# iptables -A FORWARD -i eth1 -p tcp --syn -m conntrack --cts
tate NEW -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -p tcp -m conntrack --ctstate RELATED,E
STABLISHED -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -p tcp -j DROP
root@57b7865b5245:/# iptables -P FORWARD ACCEPT
root@57b7865b5245:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080 flags
:0x17/0x02 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080 flags
:0x17/0x02 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp flags:0x17/0x0
2 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ES
TABLISHED
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@57b7865b5245:/#
```

In this initially, we will accept the new sync packet used to establish a connection to TCP. As we want to accept the TCP packets from the established and related connections as long as they follow the established and related rule and any other connection will be dropped.

```
root@57b7865b5245:/# iptable -A FORWARD -p tcp -i eth0 -d 192.168.60.5 --dport 2
3 --syn -m conntrack --ctstate NEW -j ACCEPT
bash: iptable: command not found
root@57b7865b5245:/# iptables -A FORWARD -p tcp -i eth0 -d 192.168.60.5 --dport
23 --syn -m conntrack --ctstate NEW -j ACCEPT
root@57b7865b5245:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080 flags
:0x17/0x02 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp dpt:8080 flags
:0x17/0x02 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           tcp flags:0x17/0x0
2 ctstate NEW
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0           ctstate RELATED,ES
TABLISHED
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0
ACCEPT     tcp  --  0.0.0.0/0            192.168.60.5        tcp dpt:23 flags:0
x17/0x02 ctstate NEW

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@57b7865b5245:/#
```

Inieially, we try telnet to 10.9.0.5 external host. We see it is successfully connected.

```
Connection closed by foreign host.
root@e753f8b4b199:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
1165e7b71b2c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Sat Apr  1 02:27:07 UTC 2023 from 192.168.60.5 on pts/2
seed@1165e7b71b2c:~$ nc -l 9090
^C
seed@1165e7b71b2c:~$ nc -u 10.9.0.5 9090
hello^C
seed@1165e7b71b2c:~$ nc -u 10.9.0.5 9090
hello^C
seed@1165e7b71b2c:~$
```

Here, we can see that in the internal network connection to host 60.6 or host 60.7 is not allowed as specified in the rules.

```
^C
root@1165e7b71b2c:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
root@1165e7b71b2c:/# telnet 192.168.60.6
Trying 192.168.60.6...
^C
root@1165e7b71b2c:/# nc 192.168.60.5 9090
hello
^C
root@1165e7b71b2c:/# nc -ul 9090
hello^C
root@1165e7b71b2c:/# nc -ul 9090
hello
^C
root@1165e7b71b2c:/#
```

From the outside host, we try connecting to the internal server through the netcat TCP server, which is not allowed.

Finally, we will set the default policy on FORWARD to drop everything. This way, if a packet is not accepted by the two rules above, they will be dropped.

```
root@57b7865b5245:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@57b7865b5245:/# iptables -P FORWARD ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -i eth0 -d 192.168.60.5 -p tcp --dport
23 --syn -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -i eth0 -p tcp --syn -j DROP
root@57b7865b5245:/# iptables -A FORWARD -p tcp -j ACCEPT
root@57b7865b5245:/# iptables -P FORWARD ACCEPT
root@57b7865b5245:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  0.0.0.0/0            192.168.60.5         tcp dpt:23 flags:0
x17/0x02
DROP       tcp  --  0.0.0.0/0            0.0.0.0/0            tcp flags:0x17/0x0
2
ACCEPT     tcp  --  0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@57b7865b5245:/#
```

```
root@1165e7b71b2c:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
e753f8b4b199 login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.
```

```
seed@1165e7b71b2c:~$ telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
1165e7b71b2c login: seed
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-54-generic x86_64)

 * Documentation:  https://help.ubuntu.com
```

On the basis of the rules set with the connection tracking mechanism in which with or without the mechanism used the disadvantage of the connection based mechanism is that it consumes more resources. As, the connection is needed to be kept open which is not the case for the without connection tracking mechanism.

Whereas, for the mechanism without connection tracking, the rules are not as strict which is not very safe if we compare to the connection tracking. This does not happen with the connection tracking based as the rules as they are strict as we observed above, such that, TCP connections reply only to those with established and related connections.

Task 4: Limiting Network Traffic

We run the following commands on router, and then ping 192.168.60.5 from 10.9.0.5

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh seed-router
root@57b7865b5245:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute
--limit-burst 5 -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
root@57b7865b5245:/#
```

We can see below that the connection is established but the firewall is not able to limit the number of packets that can pass through the firewall.

```
root@e753f8b4b199:/# 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=0.081 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.085 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.133 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.085 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=63 time=0.084 ms
64 bytes from 10.9.0.5: icmp_seq=6 ttl=63 time=0.098 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=63 time=0.087 ms
64 bytes from 10.9.0.5: icmp_seq=8 ttl=63 time=0.160 ms
64 bytes from 10.9.0.5: icmp_seq=9 ttl=63 time=0.203 ms
64 bytes from 10.9.0.5: icmp_seq=10 ttl=63 time=0.135 ms
64 bytes from 10.9.0.5: icmp_seq=11 ttl=63 time=0.087 ms
64 bytes from 10.9.0.5: icmp_seq=12 ttl=63 time=0.093 ms
64 bytes from 10.9.0.5: icmp_seq=13 ttl=63 time=0.124 ms
64 bytes from 10.9.0.5: icmp_seq=14 ttl=63 time=0.155 ms
64 bytes from 10.9.0.5: icmp_seq=15 ttl=63 time=0.100 ms
64 bytes from 10.9.0.5: icmp_seq=16 ttl=63 time=0.165 ms
64 bytes from 10.9.0.5: icmp_seq=17 ttl=63 time=0.098 ms
64 bytes from 10.9.0.5: icmp_seq=18 ttl=63 time=0.154 ms
^C
--- 10.9.0.5 ping statistics ---
18 packets transmitted, 18 received, 0% packet loss, time 17379ms
rtt min/avg/max/mdev = 0.081/0.118/0.203/0.035 ms
```

We add another rule for which the packets not dropped from the first rule will be dropped from the second rule which does not match the first rule:

```
root@57b7865b5245:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
root@57b7865b5245:/# iptables -F
```

After doing ping again and get the following results:

```
root@e753f8b4b199:/# 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=0.089 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.147 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.181 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.115 ms
64 bytes from 10.9.0.5: icmp_seq=5 ttl=63 time=0.086 ms
64 bytes from 10.9.0.5: icmp_seq=7 ttl=63 time=0.086 ms
64 bytes from 10.9.0.5: icmp_seq=13 ttl=63 time=0.212 ms
64 bytes from 10.9.0.5: icmp_seq=19 ttl=63 time=0.112 ms
64 bytes from 10.9.0.5: icmp_seq=25 ttl=63 time=0.366 ms
64 bytes from 10.9.0.5: icmp_seq=31 ttl=63 time=0.080 ms
^C
--- 10.9.0.5 ping statistics ---
32 packets transmitted, 10 received, 68.75% packet loss, time 31823ms
rtt min/avg/max/mdev = 0.080/0.147/0.366/0.084 ms
```

Hence, we add another rule for which the packets are not dropped from the first rule and will be dropped from the second rule which does not match the first rule as can be seen below:

There is a time condition in the connection which is of about 10-per-min. that means it takes about 6 seconds for each packet being sent. This shows that the second rule is required.

So, for next task, we restore the filter table to its original state:

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh seed-router
root@57b7865b5245:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit
-burst 5 -j ACCEPT
root@57b7865b5245:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
root@57b7865b5245:/# iptables -F
root@57b7865b5245:/# iptables -L -n
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 \
> -m statistic --mode nth --every 3 --packet 0 \
> exit
Bad argument `exit'
Try `iptables -h' or 'iptables --help' for more information.
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode nth --every 3 --packet 0 -j DNAT --to-destination 192.168.60.5:8080
root@57b7865b5245:/#
```

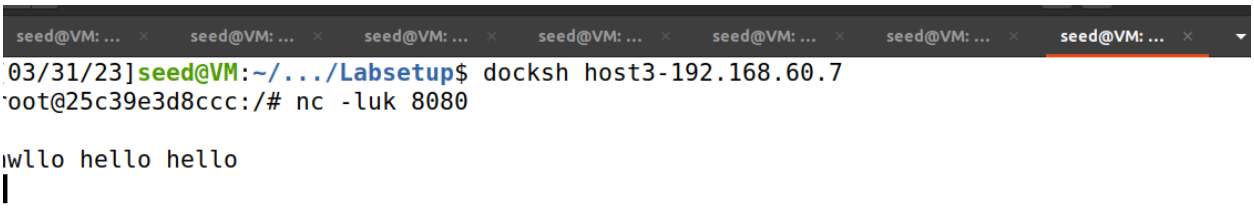
Using other ways to check the connection like by-

Sending echo messages to see the packets sent by starting server on each of the hosts: 192.168.60.5, 192.168.60.6, and 192.168.60.7.

```
seed@e753f8b4b199:~$ exit
logout
Connection closed by foreign host.
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
```

```
64 bytes from 10.9.0.5: icmp_seq=31 ttl=63 time=0.080 ms
^C
--- 10.9.0.5 ping statistics ---
32 packets transmitted, 10 received, 68.75% packet loss, time 31823ms
rtt min/avg/max/mdev = 0.080/0.147/0.366/0.084 ms
root@e753f8b4b199:/# nc -luk 8080
hello
hello
hwllo hello hello
```

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh host2-192.168.60.6
root@e82b9b0944e4:/# nc -luk 8080

hwllo hello hello
```



Task 5: Load Balancing:

In this we start the server on each of the hosts: 192.168.60.5, 192.168.60.6, and 192.168.60.7. In the router, we set up the following rules:

```
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode nth --every 3 --packet 0 -j DNAT --to-destination 192.168.60.5:8080
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode nth --every 2 --packet 0 -j DNAT --to-destination 192.168.60.6:8080
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode nth --every 1 --packet 0 -j DNAT --to-destination 192.168.60.7:8080
```

It sends the packets coming to 8080 to the inner hosts. From the external host we run the given commands to send a UDP packet to the router's 8080 port, we will see that one out of three packets gets to 192.168.60.5

```
Connection closed by foreign host.
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# nc -u 10.9.0.11 8080
qwe2
qwe3
^C
root@1165e7b71b2c:/# echo 'qwe4'| nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
```

```
32 packets transmitted, 10 received, 68.75% packet loss,
rtt min/avg/max/mdev = 0.080/0.147/0.366/0.084 ms
root@e753f8b4b199:/# nc -luk 8080
hello
hello
hwllo hello hello
```

Using the random mode:

Now, hence we modify the rules based on the random mode for achieving load balancing with random probabilities assigned to the packets send to different hosts:

```
Chain DOCKER_POSTROUTING (1 references)
target     prot opt source               destination
SNAT       tcp  --  127.0.0.11           0.0.0.0/0            tcp spt:41953 to::53
SNAT       udp  --  127.0.0.11           0.0.0.0/0            udp spt:44138 to::53
root@57b7865b5245:/# iptables -t nat -D PREROUTING 1
root@57b7865b5245:/# iptables -t nat -D PREROUTING 2
root@57b7865b5245:/# iptables -t nat -D PREROUTING 3
iptables: Index of deletion too big.
root@57b7865b5245:/# iptables -t nat -D PREROUTING 1
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode random --probability 0.33333 -j DNAT --to-destination 192.168.60.5:8080
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode random --probability 0.5 -j DNAT --to-destination 192.168.60.6:8080
root@57b7865b5245:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --m
ode random --probability 1 -j DNAT --to-destination 192.168.60.7:8080
root@57b7865b5245:/# iptables -t nat -L -n --line-numbers
Chain PREROUTING (policy ACCEPT)
num  target     prot opt source               destination
1    DNAT       udp  --  0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statist
ic mode nth every 1 to:192.168.60.7:8080
2    DNAT       udp  --  0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statist
ic mode random probability 0.33332999982 to:192.168.60.5:8080
3    DNAT       udp  --  0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statist
ic mode random probability 0.50000000000 to:192.168.60.6:8080
4    DNAT       udp  --  0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statist
ic mode random probability 1.00000000000 to:192.168.60.7:8080

Chain INPUT (policy ACCEPT)
num  target     prot opt source               destination
```

Hence, we now check by sending the messages repeatedly while we can see that all the internal hosts gets the equal number of packets as follows:

```
Connection closed by foreign host.
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo hello | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# echo "hwllo hello hello" | nc -u 10.9.0.11 8080
^C
root@1165e7b71b2c:/# nc -u 10.9.0.11 8080
```

Hence, we see that based on the probabilities the packets are send to different internal hosts randomly:

```
[03/31/23]seed@VM:~/.../Labsetup$ docksh host2-192.168.60.6
root@e82b9b0944e4:/# nc -luk 8080

hwllo hello hello
```

```
seed@VM: ... × seed@VM: ... × seed@VM: ... × seed@VM: ... × seed@VM: ... × seed@VM: ... × seed@VM: ... ×
[03/31/23]seed@VM:~/.../Labsetup$ docksh host3-192.168.60.7
root@25c39e3d8ccc:/# nc -luk 8080

hwllo hello hello
qwe2
qwe3
qwe4
hwllo hello hello
qwer hello hello
qwe5
qwe6
```