COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 1: Implementing a Simple Firewall

For this task, the IP addresses of the systems are as follows

Host VM → 10.0.2.5 HostA → 10.0.2.4

Task 1.A: Implement a Simple Kernel Module

A simple kernel module 'hello' is inserted into the kernel and the corresponding outputs of that module are stored in the kernel buffer. These outputs are observed when the module is being inserted and removed.

```
PES1UG20CS280(10.0.2.5) -$make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/Desktop/CNS/Codes/kerne
l_module modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/Desktop/CNS/Codes/kernel_module/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/Desktop/CNS/Codes/kernel_module/hello.mod.o
  LD [M]  /home/seed/Desktop/CNS/Codes/kernel_module/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
PES1UG20CS280(10.0.2.5) -$sudo insmod hello.ko
PES1UG20CS280(10.0.2.5) -$lsmod | grep hello
hello                  16384  0
PES1UG20CS280(10.0.2.5) -$sudo rmmod hello.ko
PES1UG20CS280(10.0.2.5) -$
```

Hello.ko is inserted into the kernel and then removed. The corresponding messages printed into the buffer when inserted and removed are shown below-

```
PES1UG20CS280(10.0.2.5) -$sudo dmesg -k -w
[    0.000000] Linux version 4.8.0-36-generic (buildd@lgw01-13) (gcc version 5.4
.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.4) ) #36~16.04.1-Ubuntu SMP Sun Feb 5 0
9:39:41 UTC 2017 (Ubuntu 4.8.0-36.36~16.04.1-generic 4.8.11)
[    0.000000] KERNEL supported cpus:
[    0.000000]   Intel GenuineIntel
[    0.000000]   AMD AuthenticAMD
[    0.000000]   NSC Geode by NSC
[    0.000000]   Cyrix CyrixInstead
[    0.000000]   Centaur CentaurHauls
[    0.000000]   Transmeta GenuineTMx86
[    0.000000]   Transmeta TransmetaCPU
[    0.000000]   UMC UMC UMC UMC
[    0.000000] [Firmware Bug]: TSC doesn't count with P0 frequency!
[    0.000000] x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating point regi
sters'
```

```
17:40:15) release log
                00:00:00.000228 main     Log opened 2022-10-21T08:04:18.802982000
Z
[    9.968176] 00:00:00.000435 main     OS Product: Linux
[    9.968215] 00:00:00.000481 main     OS Release: 4.8.0-36-generic
[    9.968253] 00:00:00.000519 main     OS Version: #36~16.04.1-Ubuntu SMP Sun F
eb 5 09:39:41 UTC 2017
[    9.968314] 00:00:00.000557 main     Executable: /opt/VBoxGuestAdditions-5.1.
14/sbin/VBoxService
                00:00:00.000558 main     Process ID: 1400
                00:00:00.000560 main     Package type: LINUX_32BITS_GENERIC
[    9.971364] 00:00:00.003512 main     5.1.14 r112924 started. Verbose level =
0
[    9.989518] 00:00:00.021690 automount vbsvcAutoMountWorker: Shared folder 'se
ed' was mounted to '/media/sf_seed'
[ 3424.296337] Hello World!
[ 3494.618779] Bye-bye World!.
```

The other contents shown in the buffer are messages that are already present inside of it. Along with those messages, Hello World and Bye-bye World are displayed (the two messages printed by the kernel module).

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280

5th Semester E section

Task 1.B: Implement a Simple Firewall Using Netfilter

1. Google's DNS server is used to obtain the IP address of www.example.com. The result is as shown below-

```
PES1UG20CS280(10.0.2.5) -$dig @8.8.8.8 www.example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @8.8.8.8 www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 21297
;; 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.        21076   IN      A       93.184.216.34

;; Query time: 19 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Oct 21 05:10:12 EDT 2022
;; MSG SIZE  rcvd: 60

PES1UG20CS280(10.0.2.5) -$
```

seedFilter module is inserted into the kernel. This module drops all UDP packets directed to 8.8.8.8

```
PES1UG20CS280(10.0.2.5) -$make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/Desktop/CNS/Codes/packe
t_filter modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/Desktop/CNS/Codes/packet_filter/seedFilter.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/Desktop/CNS/Codes/packet_filter/seedFilter.mod.o
  LD [M]  /home/seed/Desktop/CNS/Codes/packet_filter/seedFilter.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
PES1UG20CS280(10.0.2.5) -$sudo insmod seedFilter.ko
PES1UG20CS280(10.0.2.5) -$lsmod | grep seedFilter
seedFilter             16384  0
PES1UG20CS280(10.0.2.5) -$dig @8.8.8.8 www.example.com
^Z
[1]+  Stopped                 dig @8.8.8.8 www.example.com
PES1UG20CS280(10.0.2.5) -$
```

When dig is executed, directed to 8.8.8.8, we see that we do not obtain any results. This indicates that the UDP packet has been dropped by the module.

When we check the kernel buffer, we see that the packets from 10.0.2.5 directed to 8.8.8.8 are dropped by the seedFilter module at the LOCAL_OUT hook.

```
[ 4288.219241] *** LOCAL_OUT
[ 4288.219243]     10.0.2.5  --> 10.0.2.3 (UDP)
[ 4382.307442] *** LOCAL_OUT
[ 4382.307446]     10.0.2.5  --> 8.8.8.8 (UDP)
[ 4382.307449] *** Dropping 8.8.8.8 (UDP), port 53
[ 4387.307770] *** LOCAL_OUT
[ 4387.307772]     10.0.2.5  --> 8.8.8.8 (UDP)
[ 4387.307775] *** Dropping 8.8.8.8 (UDP), port 53
[ 4523.111901] *** LOCAL_OUT
[ 4523.111905]     10.0.2.5  --> 10.0.2.3 (UDP)
[ 4627.091179] The filters are being removed.
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280

5th Semester E section

2. In this task seedPrint.ko is inserted into the kernel. This module hooks the module to all the 5 hooks present and prints details of each hook appropriately when a dig command is used. The output of the dig command is as shown below-
   The output of lsmod can also be seen at the beginning.

```
seedPrint              16384  0
PES1UG20CS280(10.0.2.5) -$dig @8.8.8.8 www.example.com

; <<>> DiG 9.10.3-P4-Ubuntu <<>> @8.8.8.8 www.example.com
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 40956
;; 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.        21367   IN      A       93.184.216.34

;; Query time: 14 msec
;; SERVER: 8.8.8.8#53(8.8.8.8)
;; WHEN: Fri Oct 21 05:26:45 EDT 2022
;; MSG SIZE  rcvd: 60

PES1UG20CS280(10.0.2.5) -$
```

UDP query is generated by the host system. This query first passes to the LOCAL_OUT hook and then to the POST_ROUTING hook before leaving the system. Once the query response reaches the system , it passes through the PRE_ROUTING hook and then into the LOCAL_IN hook and the response IP is displayed on screen to the user.

```
PES1UG20CS280(10.0.2.5) -$sudo dmesg -k -w
[ 4889.696288] Registering filters.
[ 4955.529141] *** LOCAL_OUT
[ 4955.529144]     10.0.2.5  --> 8.8.8.8 (UDP)
[ 4955.529144] *** POST_ROUTING
[ 4955.529145]     10.0.2.5  --> 8.8.8.8 (UDP)
[ 4955.543377] *** PRE_ROUTING
[ 4955.543381]     8.8.8.8  --> 10.0.2.5 (UDP)
[ 4955.543388] *** LOCAL_IN
[ 4955.543389]     8.8.8.8  --> 10.0.2.5 (UDP)
[ 5057.748409] *** LOCAL_OUT
[ 5057.748413]     10.0.2.5  --> 10.0.2.3 (UDP)
[ 5057.748414] *** POST_ROUTING
[ 5057.748416]     10.0.2.5  --> 10.0.2.3 (UDP)
[ 5057.762582] *** PRE_ROUTING
[ 5057.762585]     10.0.2.3  --> 10.0.2.5 (UDP)
[ 5057.762590] *** LOCAL_IN
[ 5057.762590]     10.0.2.3  --> 10.0.2.5 (UDP)
[ 5127.917765] The filters are being removed.
```

Once the module is removed, the corresponding message in the exit() function is put into the buffer and the module is removed.

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

3. seedBlock prevents any TCP or ICMP packets from reaching the host (10.0.2.5 in this case). The module is created and inserted into the kernel

```
PES1UG20CS280(10.0.2.5) -$make
make -C /lib/modules/4.8.0-36-generic/build M=/home/seed/Desktop/CNS/Codes/packe
t_filter modules
make[1]: Entering directory '/usr/src/linux-headers-4.8.0-36-generic'
  CC [M]  /home/seed/Desktop/CNS/Codes/packet_filter/seedBlock.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /home/seed/Desktop/CNS/Codes/packet_filter/seedBlock.mod.o
  LD [M]  /home/seed/Desktop/CNS/Codes/packet_filter/seedBlock.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.8.0-36-generic'
PES1UG20CS280(10.0.2.5) -$sudo insmod seedBlock.ko
PES1UG20CS280(10.0.2.5) -$lsmod | grep seedBlock
seedBlock              16384  0
PES1UG20CS280(10.0.2.5) -$sudo rmmod seedBlock
PES1UG20CS280(10.0.2.5) -$
```

When host A (10.0.2.4) tries pinging 10.0.2.5, we see that the ping is unsuccessful. Similarly, the same is true when host A tries to telnet too.

```
PES1UG20CS280(10.0.2.4) -$ping 10.0.2.5
PING 10.0.2.5 (10.0.2.5) 56(84) bytes of data.
^Z
[2]+  Stopped                 ping 10.0.2.5
PES1UG20CS280(10.0.2.4) -$telnet 10.0.2.5
Trying 10.0.2.5...
^Z
^C
PES1UG20CS280(10.0.2.4) -$
```

In the kernel buffer, we see that the corresponding packets for ICMP and TCP are dropped by the kernel module that is attached to the input portion of the hooks. HostA keeps sending ping requests but they all keep getting dropped at 10.0.2.5. The same holds true for TCP packets that are sent during telnet.

```
PES1UG20CS280(10.0.2.5) -$sudo dmesg -k -w
[  498.722817] Registering filters.
[  519.021304] *** LOCAL_OUT
[  519.021306]     10.0.2.5  --> 224.0.0.251 (UDP)
[  519.849127] *** Dropping 10.0.2.5 (ICMP)
[  520.854490] *** Dropping 10.0.2.5 (ICMP)
[  521.880767] *** Dropping 10.0.2.5 (ICMP)
[  522.901040] *** Dropping 10.0.2.5 (ICMP)
[  523.925179] *** Dropping 10.0.2.5 (ICMP)
[  524.948529] *** Dropping 10.0.2.5 (ICMP)
[  525.975732] *** Dropping 10.0.2.5 (ICMP)
[  527.009648] *** Dropping 10.0.2.5 (ICMP)
[  528.019656] *** Dropping 10.0.2.5 (ICMP)
[  529.043161] *** Dropping 10.0.2.5 (ICMP)
[  540.643803] *** Dropping 10.0.2.5 (TCP), port 23
[  541.659348] *** Dropping 10.0.2.5 (TCP), port 23
[  543.660072] *** Dropping 10.0.2.5 (TCP), port 23
[  547.762374] *** Dropping 10.0.2.5 (TCP), port 23
[  570.688275] *** LOCAL_OUT
[  570.688277]     10.0.2.5  --> 10.0.2.3 (UDP)
[  571.087518] The filters are being removed.
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 2: Experimenting with Stateless Firewall Rules

Task 2.A: Protecting the Router

The first command shown below gives the details of the filter IP table and the rules that are attached to its respective chains.

```
ROUTER_CS280:/# 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
ROUTER_CS280:/# iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
ROUTER_CS280:/# iptables -A OUTPUT -p icmp --icmp-type echo-reply -j ACCEPT
ROUTER_CS280:/# iptables -P OUTPUT DROP
ROUTER_CS280:/# iptables -P INPUT DROP
ROUTER_CS280:/#  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

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

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

The first rule entered is used to indicate that any echo requests packets that reach the firewall (as INPUT) must be allowed to pass. The second rule states that any echo reply packets that reach the firewall (as response/ OUTPUT) must be allowed to pass through.

The third and fourth rules are used to change the standard policy of the INPUT and OUTPUT chains of the filter table to DROP i.e any other packets apart from those that can be accepted by the rules are dropped at those respective chains.

When the seed-router is pinged, ICMP echo requests are allowed to enter the router and ICMP echo replies are allowed to leave the router to HostA as these packets are allowed by the firewall.

```
host_A_10.9.0.5:/# 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=0.126 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=2 ttl=64 time=0.069 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=3 ttl=64 time=0.069 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=4 ttl=64 time=0.184 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=5 ttl=64 time=0.109 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=6 ttl=64 time=0.067 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=7 ttl=64 time=0.072 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=8 ttl=64 time=0.145 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=9 ttl=64 time=0.076 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=10 ttl=64 time=0.068 ms
```

When HostA tries to telnet to the router however, it is not successful as shown below

```
host_A_10.9.0.5:/# telnet seed-router
Trying 10.9.0.11...
```

This is because all TCP packets that reach the INPUT section of the firewall are dropped (based on the rule). Any other packet apart from ICMP will be dropped as the rules say so.

The corresponding cleanup code is shown below. -F is used to flush all the rules that are put into the IP table. The INPUT and OUTPUT chains are shifted back to their original policy of ACCEPT/allow.

```
ROUTER_CS280:/# iptables -F
ROUTER_CS280:/# iptables -P OUTPUT ACCEPT
ROUTER_CS280:/# iptables -P INPUT ACCEPT
ROUTER_CS280:/#
```

Questions :

(1) Can you ping the router?

Ans: Yes, ping is successful. The reason for the same is already explained above.

(2) Can you telnet into the router (a telnet server is running on all the containers; an account called seed was created on them with a password dees).

Ans: No, telnet to the seed-router is not possible. The explanation for the same is given right above.

Task 2.B: Protecting the Internal Network

The first rule drops ICMP echo request packets that are originating from the eth0 network that are to be forwarded to some other network via the seed-router (the place where the firewall is placed).

The second rule allows any ICMP echo request packets that are originating from the eth1 network (the internal LAN i.e 192.168.60.0/24) to be forwarded to any other network via the seed-router.

The third rule allows any ICMP echo replies arriving from the eth0 network to pass through.

The fourth rule is used to change the standard policy of the forwarding chain to DROP i.e packets that do not follow the FORWARD rules are dropped (and not forwarded) by default.

```
ROUTER_CS280:/# iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-request -j DROP
ROUTER_CS280:/# iptables -A FORWARD -i eth1 -p icmp --icmp-type echo-request -j ACCEPT
ROUTER_CS280:/# iptables -A FORWARD -i eth0 -p icmp --icmp-type echo-reply -j ACCEPT
ROUTER_CS280:/# iptables -P FORWARD DROP
ROUTER_CS280:/# 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 DROP       icmp --  eth0   *       0.0.0.0/0            0.0.0.0/0           icmptype 8
    0     0 ACCEPT     icmp --  eth1   *       0.0.0.0/0            0.0.0.0/0           icmptype 8
    0     0 ACCEPT     icmp --  eth0   *       0.0.0.0/0            0.0.0.0/0           icmptype 0

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
ROUTER_CS280:/#
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Testing tasks:

1. Outside hosts cannot ping internal hosts.
   The first rule drops ICMP echo request packets (that are to be forwarded), therefore the output is as shown below-

```
host_A_10.9.0.5:/# ping 192.168.60.5
PING 192.168.60.5 (192.168.60.5) 56(84) bytes of data.
```

2. Outside hosts can ping the router.
   The standard policy for INPUT and OUTPUT chains is ACCEPT. There is no forwarding in this case as the user is pinging the router and not a system on another network. Because of the standard policies, ICMP echo request packets are accepted at the INPUT chain and ICMP echo reply packets are sent out of the OUTPUT chain.

```
host_A_10.9.0.5:/# 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=0.059 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=2 ttl=64 time=0.072 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=3 ttl=64 time=0.067 ms
64 bytes from seed-router.net-10.9.0.0 (10.9.0.11): icmp_seq=4 ttl=64 time=0.075 ms
```

3. Internal hosts can ping Outside Hosts.
   The second rule allows hosts inside to ping hosts outside. Therefore, the result of the same is as shown below-

```
host_1_60.5:/# 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=0.195 ms
64 bytes from 10.9.0.5: icmp_seq=2 ttl=63 time=0.082 ms
64 bytes from 10.9.0.5: icmp_seq=3 ttl=63 time=0.128 ms
64 bytes from 10.9.0.5: icmp_seq=4 ttl=63 time=0.144 ms
```

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

```
host_1_60.5:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

The fourth rule states that packets apart from ICMP that are to be forwarded must be dropped. Therefore, telnet is not successful.

The code for cleanup are as shown below-

```
ROUTER_CS280:/# iptables -F
ROUTER_CS280:/#  iptables -P OUTPUT ACCEPT
ROUTER_CS280:/# iptables -P INPUT ACCEPT
ROUTER_CS280:/#
```

The manual does not include iptables -P FORWARD ACCEPT command. This is also needed to ensure that the default FORWARD policy is set to ACCEPT. That command is also executed during cleanup

Task 2.C: Protecting Internal Servers

The first rule states that only TCP packets sent from eth0 network destined for 192.168.60.5 at port 23 be allowed through the firewall (for forwarding).

The second rule states that only TCP packets sent from 192.168.60.5 (eth1 network) originating at source port 23 be allowed through the firewall (for forwarding)

The default policy of FORWARD is set to DROP.

```
ROUTER_CS280:/# iptables -A FORWARD -i eth0 -d 192.168.60.5 -ptcp --dport 23 -j ACCEPT
ROUTER_CS280:/# iptables -A FORWARD -i eth1 -s 192.168.60.5 -p tcp --sport 23 -j ACCEPT
ROUTER_CS280:/# iptables -P FORWARD DROP
ROUTER_CS280:/# v
bash: v: command not found
ROUTER_CS280:/# 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            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
ROUTER_CS280:/#
```

Testing tasks:-

1. All the internal hosts run a telnet server (listening to port 23). Outside hosts can only access the telnet server on 192.168.60.5, not the other internal hosts.

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

```
host_A_10.9.0.5:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dee1a53aa497 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.

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.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@dee1a53aa497:~$
```

The first rule allows telnet to 192.168.60.5 from 10.9.0.5 (part of eth0 network).

2. Outside hosts cannot access other internal servers.

The DROP rule ensures that any other TCP packets directed to the other servers in the eth1 network are dropped at the firewall. Therefore, both the telnets to .6 and .7 are unsuccessful.

```
host_A_10.9.0.5:/# telnet 192.168.60.6
Trying 192.168.60.6...
```

```
host_A_10.9.0.5:/# telnet 192.168.60.7
Trying 192.168.60.7...
```

4. Internal hosts can access all the internal servers.
   There is no forwarding at the router when two systems on a local network intend to communicate with each other. Therefore, both the telnets (to 60.5 and 60.7) are successful.

Host 1

```
host_2_60.6:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dee1a53aa497 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.
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Host 3

```
host_2_60.6:/# telnet 192.168.60.7
Trying 192.168.60.7...
Connected to 192.168.60.7.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
e9e1e4f6dfda 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.

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.
```

4. Internal hosts cannot access external servers.

   The second rule only allows 192.168.60.5 to access external servers via telnet; the other internal servers cannot do so as the firewall blocks those packets.

```
host_2_60.6:/# telnet 10.9.0.5
Trying 10.9.0.5...
```

Thus, telnet to 10.9.0.5 is not successful via Host2

Cleanup

The commands for cleanup stay the same. They are as follows-

```
ROUTER_CS280:/# iptables -F
ROUTER_CS280:/#  iptables -P OUTPUT ACCEPT
ROUTER_CS280:/# iptables -P INPUT ACCEPT
```

Along with this iptables -P FORWARD ACCEPT must also be used to set the default FORWARD policy to ACCEPT.

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 3: Connection Tracking and Stateful Firewall

Task 3.A: Experiment with the Connection Tracking

In this task conntrack is used to create a stateful firewall and store details of the connection b/w the two systems on the router(firewall) for a certain period of time, depending on the nature of the protocol used for packet exchange.

1. ICMP experiment

   Host1 is pinged from HostA. The result of the ping is as shown below-

```
host_A_10.9.0.5:/# 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=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.098 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.139 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.123 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.108 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.099 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.126 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.081 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.104 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.122 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.083 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.130 ms
^Z
[10]+  Stopped                 ping 192.168.60.5
host_A_10.9.0.5:/#
```

The time for which an ICMP connection is kept alive in conntrack is 29s. Every time an ICMP packet is sent from hostA(10.9.0,5), the ICMP timer on conntrack is reset to 29s. The corresponding output shows the same-

```
ROUTER_CS280:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 29 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
```

Once the ping is terminated, no more ICMP packets are sent to the router/firewall; the timer does not refresh anymore. The timer value keeps decreasing every second and that can be seen below (it reduces from 27 to 26 to 24)

```
icmp     1 27 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 26 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 24 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/#
```

Once the timer reaches 0, details of that conntrack connection are removed. We can see the 0 entries detail on conntrack after the timer has completed its duration below-

```
icmp     1 11 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 5 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 3 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
icmp     1 0 src=10.9.0.5 dst=192.168.60.5 type=8 code=0 id=50 src=192.168.60.5 dst=10.9.0.5 type=0 code=0 id=50 mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
conntrack v1.4.5 (conntrack-tools): 0 flow entries have been shown.
ROUTER_CS280:/#
```

2. UDP experiment

In this case, HostA acts as the client and Host1 the server. UDP packets are exchanged b/w the two systems and details of this connection are examined and stored on conntrack.

The client connects to the server. Every time the client enters some data, the data is sent as a UDP packet to the server.

```
host_A_10.9.0.5:/# nc -u 192.168.60.5 9090
PAVAN 280CS
HELLO
^Z
[13]+  Stopped                 nc -u 192.168.60.5 9090
host_A_10.9.0.5:/#
```

The outputs of the client and server are shown here.

```
host_1_60.5:/# nc -lu 9090
PAVAN 280CS
HELLO
^Z
[4]+  Stopped                 nc -lu 9090
host_1_60.5:/#
```

The time for which a UDP connection is stored in the conntrack detail is ~27s. Every time a user enters a character on the client side, a UDP packet is sent across and the conntrack timer is reset back to 27. This timer continues to linearly decay if there are no UDP packets sent across/ the connection is closed.

Once the timer has reached 0, the entry is removed.

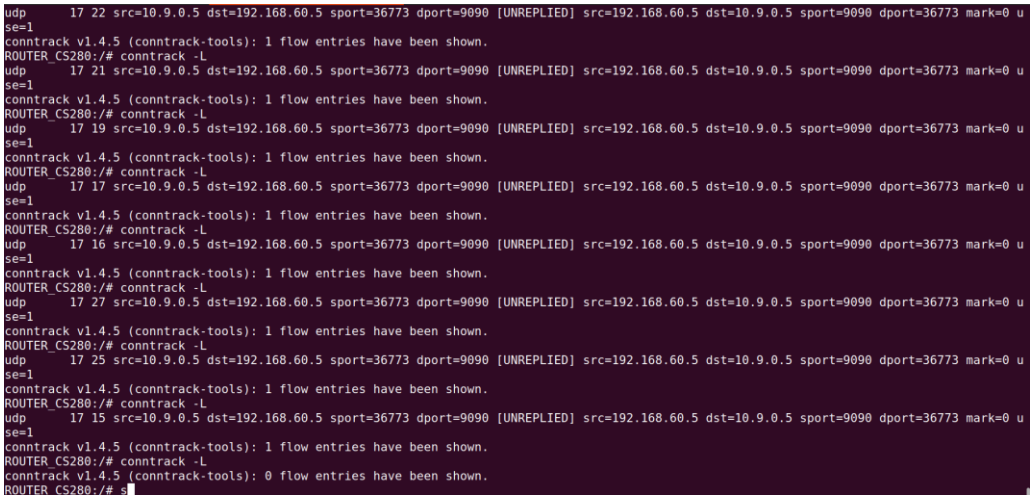
The detail stored on conntrack states UNREPLIED, indicating that the server does not reply to the message/packet sent by the client (as it happens with TCP netcat).

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280

5th Semester E section

The result of this is shown below-



3. TCP experiment

   A TCP connection is established once a 3-way handshake is completed b/w the client and server. The connection established is a reliable connection, therefore the time for which this connection detail is stored on conntrack is huge (>> UDP and ICMP).

```
host_A_10.9.0.5:/# nc 192.168.60.5 9090
HELLO THERE CNS STUDENTS
EXIT
quit
^Z
[7]+  Stopped                 nc 192.168.60.5 9090
host_A_10.9.0.5:/#
```

The output on client and server are as shown

```
host_1_60.5:/# nc -l 9090
HELLO THERE CNS STUDENTS
EXIT
quit
^C
host_1_60.5:/#
```

The timer is set to 431984s as can be seen below. The timer refreshes when a new packet is sent and decays linearly when no packets are being exchanged.

Name: Pavan R Kashyap SRN: PES1UG20CS280

5th Semester E section


```
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431984 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431984 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431983 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431983 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431983 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431982 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431982 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431981 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431981 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34470 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34470 [ASSU
```

The detail stored on conntrack states ASSURED, indicating that the server replies back to the client and that the message/packet is assured to be from the sender itself.

Once the connection is closed, the timer starts linearly decaying until it reaches 0. However, the time taken for the same is much more than that of ICMP or/and UDP.

```
ROUTER_CS280:/# conntrack -L
tcp      6 431847 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34472 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34472 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431846 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34472 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34472 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/# conntrack -L
tcp      6 431845 ESTABLISHED src=10.9.0.5 dst=192.168.60.5 sport=34472 dport=9090 src=192.168.60.5 dst=10.9.0.5 sport=9090 dport=34472 [ASSU
RED] mark=0 use=1
conntrack v1.4.5 (conntrack-tools): 1 flow entries have been shown.
ROUTER_CS280:/#
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 3.B: Setting Up a Stateful Firewall

The rules are as shown below.

Conntrack -m module is used to specify the state of the connection and the type of packets that must be allowed (based on the state).

```
ROUTER_CS280:/# iptables -A FORWARD -p tcp -i eth0 -d 192.168.60.5 --dport 23 --syn -m conntrack --ctstate NEW -j ACCEPT
ROUTER_CS280:/# iptables -A FORWARD -i eth1 -p tcp --syn -m conntrack --ctstate NEW -j ACCEPT
ROUTER_CS280:/#  iptables -A FORWARD -p tcp -m conntrack --ctstate RELATED,ESTABLISHED -j ACCEPT
ROUTER_CS280:/# iptables -A FORWARD -p tcp -j DROP
ROUTER_CS280:/# iptables -P FORWARD ACCEPT
ROUTER_CS280:/# 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
    0     0 ACCEPT     tcp  --  eth0   *       0.0.0.0/0            192.168.60.5         tcp dpt:23 flags:0x17/0x02 ctstate NEW
    0     0 ACCEPT     tcp  --  eth1   *       0.0.0.0/0            0.0.0.0/0            tcp flags:0x17/0x02 ctstate NEW
    0     0 ACCEPT     tcp  --  *      *       0.0.0.0/0            0.0.0.0/0            ctstate RELATED,ESTABLISHED
    0     0 DROP       tcp  --  *      *       0.0.0.0/0            0.0.0.0/0

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
ROUTER_CS280:/#
```

1. Telnet an inside host (198.162.60.5 in eth1 network):
   The first rule indicates that any TCP packets coming from the eth0 interface destined for 192.168.60.5 at port 23 (any NEW connections too) are to be allowed/accepted. Therefore, when HostA in eth0 interface telnets to 60.5, it allows it.

```
host_A_10.9.0.5:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dee1a53aa497 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.
```

2. Telnet other inside hosts (198.162.60.6 and .7 in eth1 network):

The third rule allows any TCP packets passing through the firewall to be accepted, based on the contention that they are packets of an established connection (not NEW) or a related connection.

The fourth rule drops all TCP packets that pass through it. The third rule is executed first, then the last.

```
host_A_10.9.0.5:/# telnet 192.168.60.6
Trying 192.168.60.6...
^Z^C
host_A_10.9.0.5:/# telnet 192.168.60.7
Trying 192.168.60.7...
^Z^C
host_A_10.9.0.5:/#
```

When the user tries to telnet, we see that both are not successful. This is because HostA is trying to establish 2 new connections with those respective systems and following rules 3 and 4, they are dropped.

3. Telnetting into another system on the same local network

```
host_2_60.6:/# telnet 192.168.60.7
Trying 192.168.60.7...
Connected to 192.168.60.7.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
e9e1e4f6dfda 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.
```

Host2 can telnet into both Host1 and Host3. The packets do not pass through the seed-router (the firewall); they directly communicate to each other across the network. Therefore, both the telnets are successful.

```
host_2_60.6:/# telnet 192.168.60.5
Trying 192.168.60.5...
Connected to 192.168.60.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
dee1a53aa497 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.
```

4.Host2 tries to telnet into HostA (which is outside the eth1 network)

```
host_2_60.6:/# telnet 10.9.0.5
Trying 10.9.0.5...
Connected to 10.9.0.5.
Escape character is '^]'.
Ubuntu 20.04.1 LTS
ea900d148e88 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.

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.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

seed@ea900d148e88:~$
```

The second rule allows any NEW TCP connection packets to be sent from the eth1 network. Host2 is sending NEW connection packets to 10.9.0.5 in the eth0 network and therefore, this is successful too.

The flush commands are the same as before and therefore, aren't repeated here.

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 4: Limiting Network Traffic

In this task, the number of packets reaching a particular system is limited using the limit keyword. When mentioning the rule in the ip table, the limit is specified along with the IP address on which this limit is to be applied on.

The first rule added to the iptables below indicates that the number of packets being forwarded from 10.9.0.5 (as source) must be limited to 10 per minute. The burst portion is used to indicate the initial number of packets that are allowed to pass before the limit comes into picture. In this case, it is 5 packets.

```
ROUTER_CS280:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-burst 5 -j ACCEPT
ROUTER_CS280:/# iptables -A FORWARD -s 10.9.0.5 -j DROP
ROUTER_CS280:/#  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
    0     0 ACCEPT     all  --  *      *       10.9.0.5             0.0.0.0/0            limit: avg 10/min burst 5
    0     0 DROP       all  --  *      *       10.9.0.5             0.0.0.0/0

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
ROUTER_CS280:/# iptables -F
```

The second rule indicates that any packets being sent from 10.9.0.5 as source must be dropped. However, when combined with the previous rule, it indicates that the limit on the number of packets that can pass the firewall from 10.9.0.5 is 10/min and therefore, any packet that does not follow this rule must be dropped.

The result of ping is as shown below.

```
host_A_10.9.0.5:/# 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=0.102 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.087 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.104 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.109 ms
64 bytes from 192.168.60.5: icmp_seq=18 ttl=63 time=0.077 ms
64 bytes from 192.168.60.5: icmp_seq=24 ttl=63 time=0.083 ms
64 bytes from 192.168.60.5: icmp_seq=30 ttl=63 time=0.205 ms
64 bytes from 192.168.60.5: icmp_seq=36 ttl=63 time=0.174 ms
```

Ping generates an ICMP packet every second. However, the firewall has a cap at 10 packets/min which means that it will only allow (60 seconds/10 packets in 1 min) 1 packet for every six seconds on an average. The first 5 packets are burst packets and there onward, every 6th packet from seq=7 (13,24,30 and so on…) is allowed and all the others in between the two values are dropped.

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280

5th Semester E section

When the last rule is removed and executed, no packets are dropped; all of them are forwarded but with the cap limit in place.

```
ROUTER_CS280:/# iptables -A FORWARD -s 10.9.0.5 -m limit --limit 10/minute --limit-burst 5 -j ACCEPT
ROUTER_CS280:/#  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
    0     0 ACCEPT     all  --  *      *       10.9.0.5             0.0.0.0/0           limit: avg 10/min burst 5

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
ROUTER_CS280:/# iptables -F
ROUTER_CS280:/#
```

The first five packets are burst packets. Subsequently, all the packets are allowed but their time values are higher than their regular ping values, which means that the firewall is capping the traffic flow, but every packet generated is being allowed to pass through.

```
host_A_10.9.0.5:/# 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=0.203 ms
64 bytes from 192.168.60.5: icmp_seq=2 ttl=63 time=0.106 ms
64 bytes from 192.168.60.5: icmp_seq=3 ttl=63 time=0.117 ms
64 bytes from 192.168.60.5: icmp_seq=4 ttl=63 time=0.142 ms
64 bytes from 192.168.60.5: icmp_seq=5 ttl=63 time=0.084 ms
64 bytes from 192.168.60.5: icmp_seq=6 ttl=63 time=0.119 ms
64 bytes from 192.168.60.5: icmp_seq=7 ttl=63 time=0.113 ms
64 bytes from 192.168.60.5: icmp_seq=8 ttl=63 time=0.131 ms
64 bytes from 192.168.60.5: icmp_seq=9 ttl=63 time=0.117 ms
64 bytes from 192.168.60.5: icmp_seq=10 ttl=63 time=0.129 ms
64 bytes from 192.168.60.5: icmp_seq=11 ttl=63 time=0.089 ms
64 bytes from 192.168.60.5: icmp_seq=12 ttl=63 time=0.105 ms
64 bytes from 192.168.60.5: icmp_seq=13 ttl=63 time=0.133 ms
64 bytes from 192.168.60.5: icmp_seq=14 ttl=63 time=0.082 ms
^Z
[16]+  Stopped                 ping 192.168.60.5
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Task 5: Load Balancing

In this task, the firewall on the router (10.9.0.11) carries out load balancing i.e it distributes the incoming packets to the internal network (192.168.60.0) based on the statistic mode that is set. There are two modes of statistic that are handled in this task.

nth mode (round-robin)

The rules for this packet routing have to written in the NAT table as these packets have to be distributed to an internal network (from the router). The hook on which these rules must be applied is the PREROUTING hook as any incoming packets to the system first go through this particular hook.

UDP packets are being exchanged and the port on which the server will listen is 8080.

The given three rules below are written to redirect every 3rd packet to 192.168.60.5, every 2nd packet to 192.168.60.6 and every 1st packet to 60.7 system.

```
ROUTER_CS280:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every 3 --packet 0 -j DNAT --to-destination 192.1
68.60.5:8080
ROUTER_CS280:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode nth --every 2 --packet 0 -j DNAT --to-destination 192.1
68.60.6:8080
ROUTER_CS280:/#  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
ROUTER_CS280:/# 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
ROUTER_CS280:/#
```

Once these rules have been inserted into the NAT table, when the contents of the NAT iptables is printed, it is as shown below-

```
ROUTER_CS280:/# iptables -t nat -L -n -v
Chain PREROUTING (policy ACCEPT 2 packets, 136 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0           udp dpt:8080 statistic mode nth every 3 to:192.168.6
0.5:8080
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0           udp dpt:8080 statistic mode nth every 2 to:192.168.6
0.6:8080
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0           udp dpt:8080 statistic mode nth every 1 to:192.168.6
0.7:8080

Chain INPUT (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
    0     0 DOCKER_OUTPUT  all  --  *      *       0.0.0.0/0            127.0.0.11

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DOCKER_POSTROUTING  all  --  *      *       0.0.0.0/0            127.0.0.11

Chain DOCKER_OUTPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            127.0.0.11          tcp dpt:53 to:127.0.0.11:44903
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            127.0.0.11          udp dpt:53 to:127.0.0.11:50650

Chain DOCKER_POSTROUTING (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 SNAT       tcp  --  *      *       127.0.0.11           0.0.0.0/0           tcp spt:44903 to::53
    0     0 SNAT       udp  --  *      *       127.0.0.11           0.0.0.0/0           udp spt:50650 to::53
ROUTER_CS280:/#
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

Host A acts like the client and all the other hosts in the 192.168.60 local network act as servers.

Three messages (Pavan, Kashyap, CS280) are entered at the client's netcat and their subsequent outputs are shown below -

```
hostA:/# nc -u 10.9.0.11 8080
Pavan
Kashyap
CS280
```

The first packet gets routed to Host 1 and hence host1 holds the first packet information. Subsequent packets (2 and 3) are not routed by the firewall/seed-router to this host and therefore, those packets are not seen on the server netcat.

```
host1:/# nc -luk 8080
Pavan
```

The second packet is routed to Host2 and displayed on screen as shown below-

```
host2:/# nc -luk 8080
Kashyap
```

Subsequently the last packet is routed to the last host and the output is shown below

```
host3:/# nc -luk 8080
CS280
```

The PREROUTING hook is filled with rules and they have to be removed. Therefore, the iptables -t nat -F PREROUTING command is used to flush all the rules attached to that particular hook.

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

the random mode

This involves routing packets to the local network based on associated probabilities i.e there is a probability associated with every system and that probability determines the proportion of traffic that the particular system/ server will receive.

Firewalls can be used to balance and divert excessive traffic reaching a single server and this task is intended to explain that.

The probability associated with 60.5's rule is set to 1/3 (0.333). The probability of 60.6's rule is set to ½ (0.5) and that of 60.7's rule is set to 1.

The rules are executed sequentially. In simple terms, it means that the probability that the first server (60.5) gets the routed packet is 0.3333. This means that the rule is not executed 0.66 or 66% of the time. The probability that the 60.6's server gets the routed packet is (0.5*0.666 = 0.333).

Only 33% of the time, the last rule is applied and therefore, the probability that the 60.7's server receives the packet is 0.333 too.

```
ROUTER_CS280:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random --probability 0.3333   -j DNAT --to-destination
192.168.60.5:8080
ROUTER_CS280:/#  iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random --probability 0.5   -j DNAT --to-destination 19
2.168.60.6:8080
ROUTER_CS280:/# iptables -t nat -A PREROUTING -p udp --dport 8080 -m statistic --mode random --probability 1   -j DNAT --to-destination 192.1
68.60.7:8080
ROUTER_CS280:/# iptables -t nat -L -n -v
Chain PREROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statistic mode random probability 0.333
30000006 to:192.168.60.5:8080
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statistic mode random probability 0.500
00000000 to:192.168.60.6:8080
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:8080 statistic mode random probability 1.000
00000000 to:192.168.60.7:8080

Chain INPUT (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
    0     0 DOCKER_OUTPUT  all  --  *      *       0.0.0.0/0            127.0.0.11

Chain POSTROUTING (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DOCKER_POSTROUTING  all  --  *      *       0.0.0.0/0            127.0.0.11

Chain DOCKER_OUTPUT (1 references)
 pkts bytes target     prot opt in     out     source               destination
    0     0 DNAT       tcp  --  *      *       0.0.0.0/0            127.0.0.11           tcp dpt:53 to:127.0.0.11:37789
    0     0 DNAT       udp  --  *      *       0.0.0.0/0            127.0.0.11           udp dpt:53 to:127.0.0.11:49585

Chain DOCKER_POSTROUTING (1 references)
 pkts bytes target     prot opt in     out     source               destination
```

The corresponding rules attached to the PREROUTING hook are shown above.

Three messages are written on the client netcat as before. There is no change here, however changes are seen at the server netcats'.

```
hostA:/# nc -u 10.9.0.11 8080
Pavan
Kashyap
CS280
```

COMPUTER NETWORK SECURITY-07

Name: Pavan R Kashyap SRN: PES1UG20CS280
5th Semester E section

We see that on host1 only one of the three packets has arrived(the first packet). It is in sync with the probability it is assigned (1/3)

```
host1:/# nc -luk 8080
Pavan
```

Host 2 and 3 in turn have 1 in 3 chance that the given packet is routed. However, we see that both the packets get redirected to Host 2 (none of the packets reach Host3). The rules imply that each host has an equiprobable chance of being chosen and therefore, in an ideal situation, each host would've obtained one packet. However, the output is as shown below-

```
host2:/# nc -luk 8080
Kashyap
CS280
```

Rule 1 was not selected and Rule 2 was selected both times (Rule 3 was never reached).

```
host3:/# nc -luk 8080
```