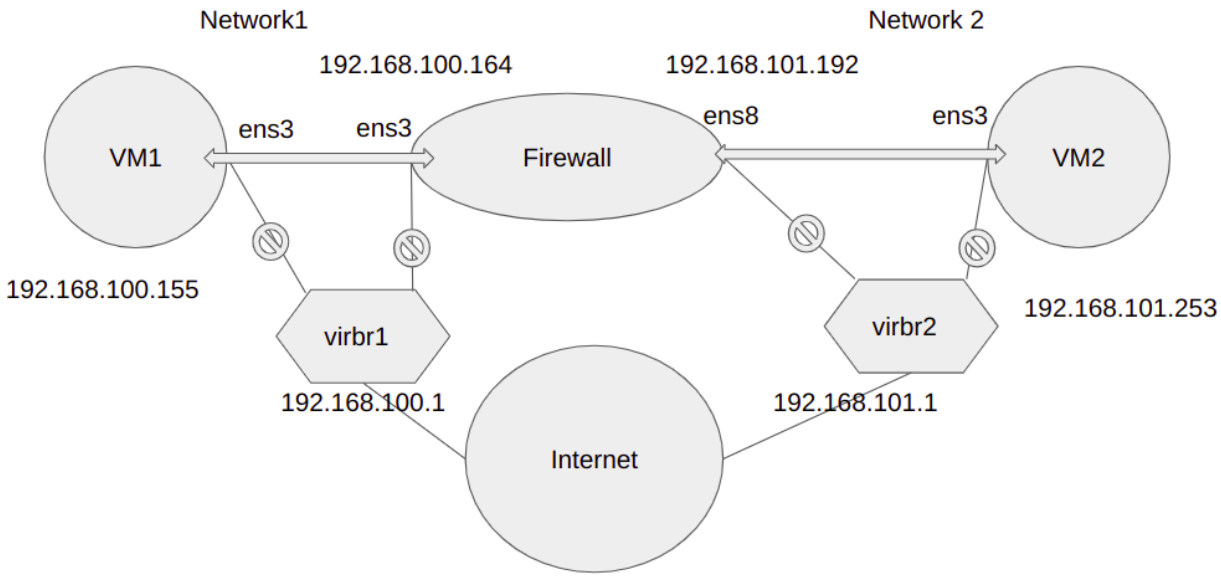
Assignment 8 : Firewall

Group Members

Divya Pathak - CS21MTECH12009
Arkadeb Ghosh - CS21MTECH14005
Anwesha Kar - CS21MTECH12006
K Shiv Kumar - CS21RESCH11003

TASK 1: Creating a simple Firewall using Socket Programming and dropping/allowing the packets using static rule matching

Network Topology :



Step 1:

1. Create 3 VM's with Ubuntu Desktop
    - host1
    - host2
    - firewall

2. Create 2 NIC's for Firewall (Interfaces : ens3 & ens8)

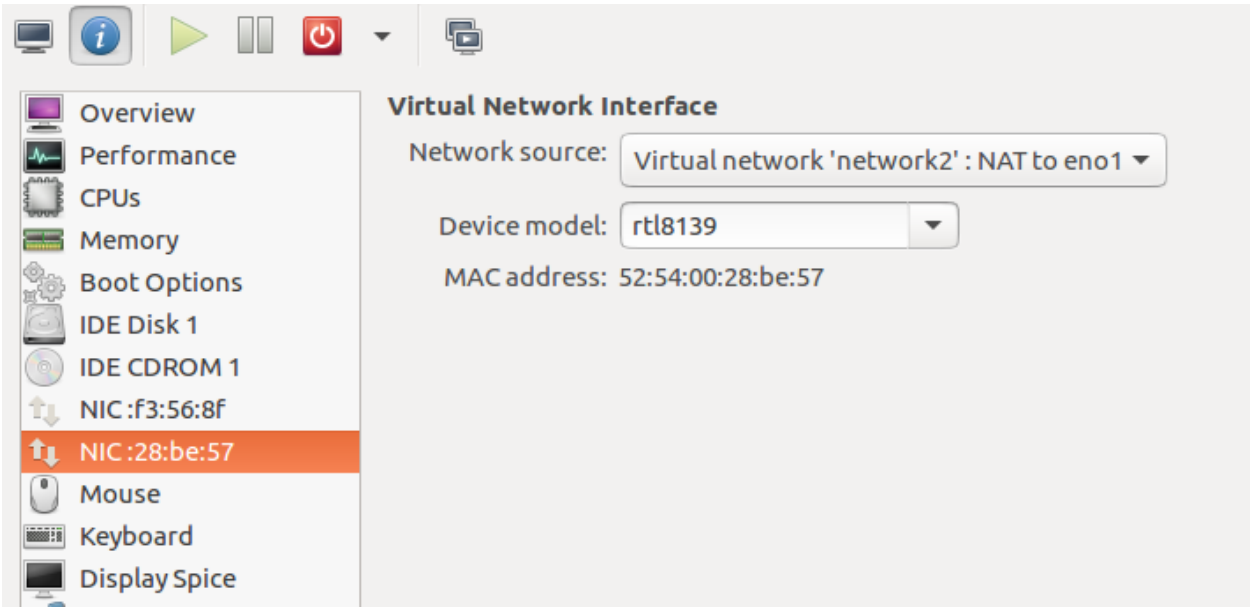
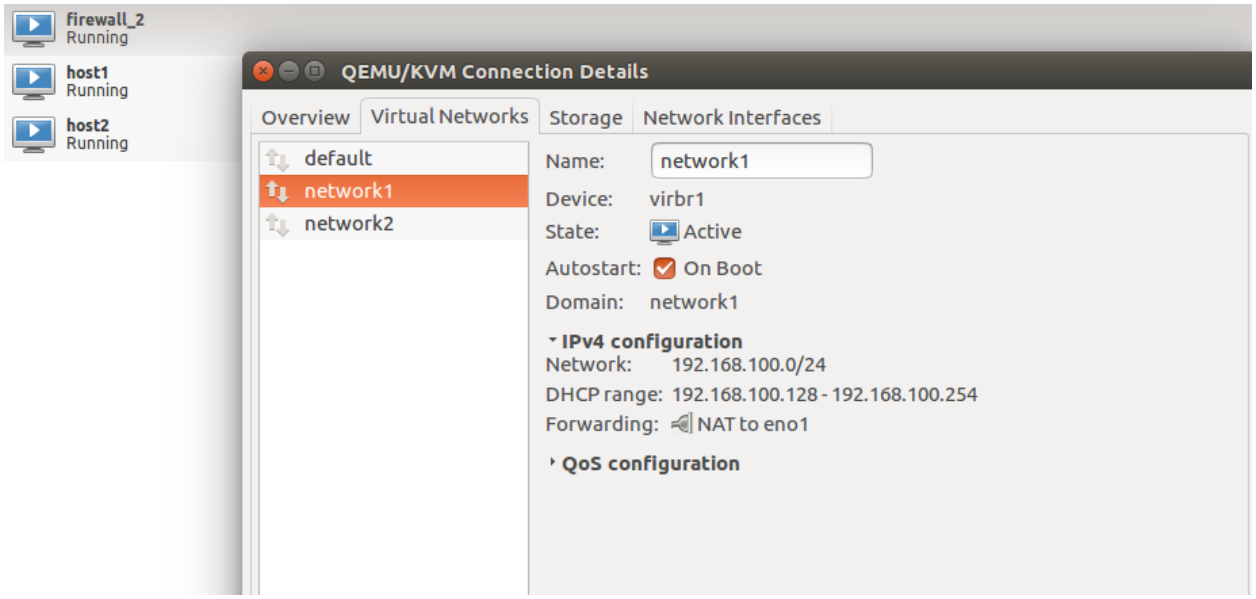
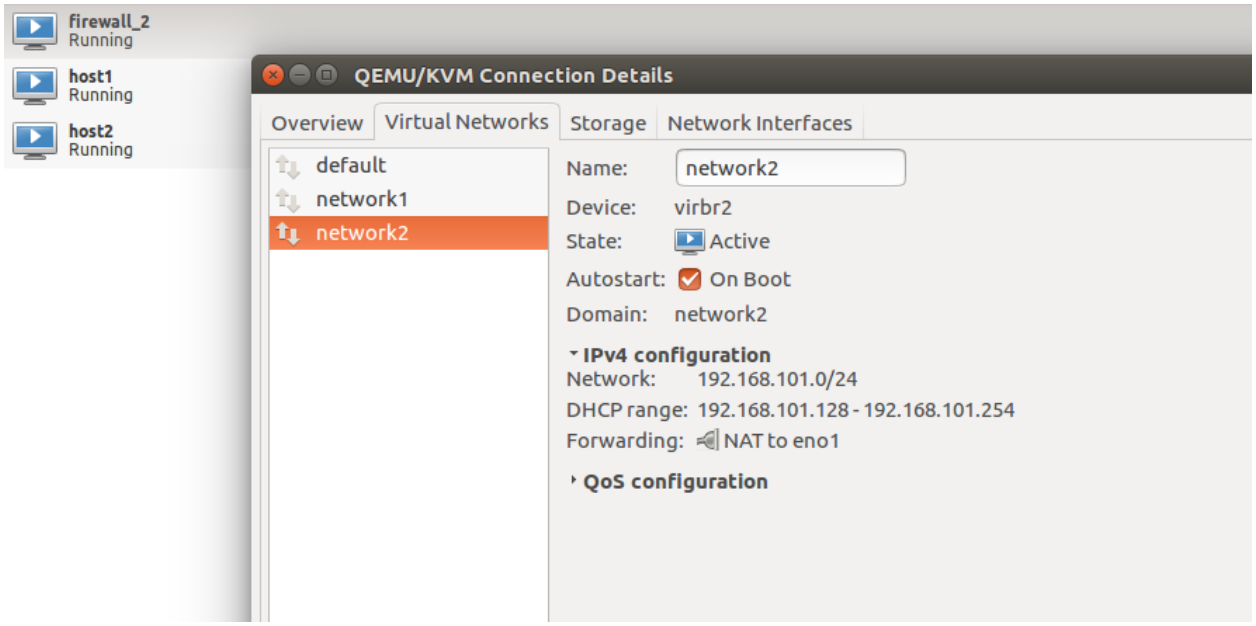


Fig: Firewall showing presence of two NIC's (one interface for Network 1 , one interface for Network 2)

3. Create two networks - network 1 , network 2
   - Network1 : Subnet -> 192.168.100.0/24
   - Gateway - virbr1 (192.168.100.1)



   - Network2 : Subnet -> 192.168.101.0/24
   - Gateway - virbr2 (192.168.101.1)



4. Configuring Host1(VM1) according to Topology:

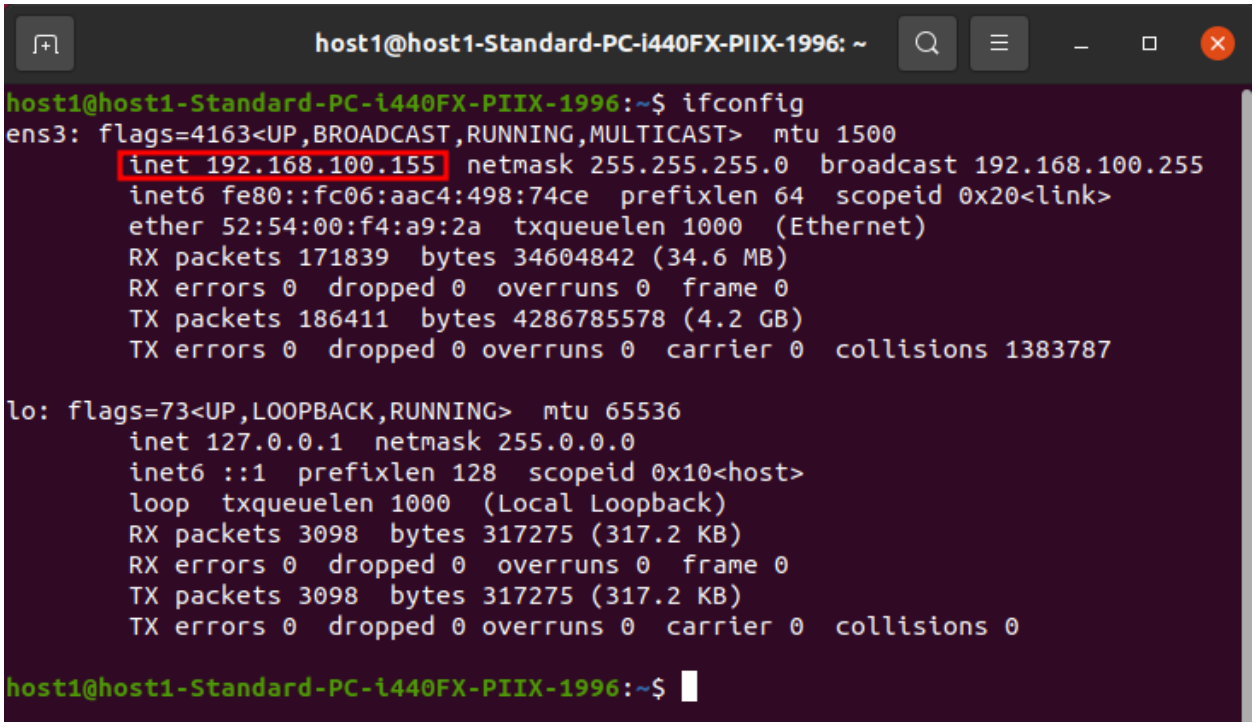


Fig: Showing IP address of host1 interface ens3 in network 1

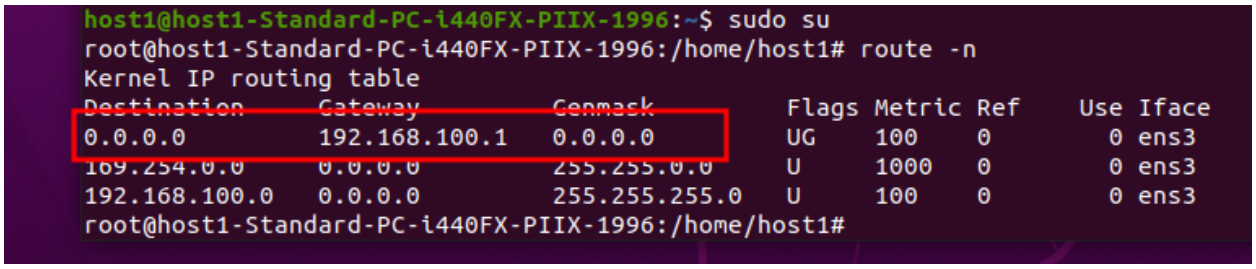


Fig: The default gateway of host 1 (virbr1). This needs to be deleted to prevent it from connecting to the internet.

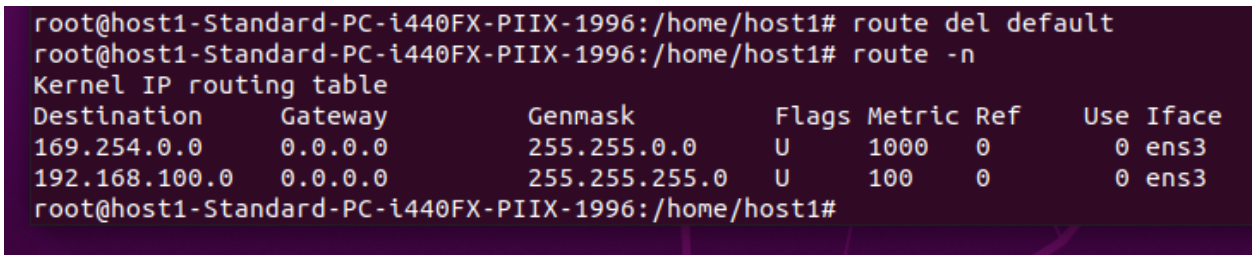


Fig: Deleting the default route.

```
root@host1-Standard-PC-i440FX-PIIX-1996:/home/host1# route add default gw 192.168.100.164
root@host1-Standard-PC-i440FX-PIIX-1996:/home/host1# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.100.164 0.0.0.0         UG    0      0        0 ens3
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 ens3
192.168.100.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
root@host1-Standard-PC-i440FX-PIIX-1996:/home/host1#
```

Fig: Directing packets from host1 to ens3 (192.168.100.164) interface of the Firewall by adding a new entry to the routing table.

----------host1 configuration done----------
----------firewall configuration ----------

```
firewall@firewall-Standard-PC-i440FX-PIIX-1996:~$ ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.100.164  netmask 255.255.255.0  broadcast 192.168.100.255
        inet6 fe80::8b31:9162:96af:af43  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:f3:56:8f  txqueuelen 1000  (Ethernet)
        RX packets 3008176  bytes 4484875188 (4.4 GB)
        RX errors 0  dropped 42385  overruns 0  frame 0
        TX packets 164055  bytes 28780258 (28.7 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 775053

ens8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.101.142  netmask 255.255.255.0  broadcast 192.168.101.255
        inet6 fe80::ad36:4293:d738:c65c  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:28:be:57  txqueuelen 1000  (Ethernet)
        RX packets 42508  bytes 49444427 (49.4 MB)
        RX errors 0  dropped 1634  overruns 0  frame 0
        TX packets 16881  bytes 2499780 (2.4 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 31578

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 3870  bytes 385254 (385.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3870  bytes 385254 (385.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Fig: Showing two interfaces of firewall - ens3 and ens8 of network1 and network2 respectively.

```
firewall@firewall-Standard-PC-i440FX-PIIX-1996:~$ sudo su
[sudo] password for firewall:
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         192.168.100.1   0.0.0.0         UG    100    0        0 ens3
0.0.0.0         192.168.101.1   0.0.0.0         UG    101    0        0 ens8
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 ens8
192.168.100.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
192.168.101.0   0.0.0.0         255.255.255.0   U     101    0        0 ens8
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall#
```

Fig: Showing default route entries for the firewall directed through virbr1 and virbr2.

```
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# route del default
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# route del default
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 ens8
192.168.100.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
192.168.101.0   0.0.0.0         255.255.255.0   U     101    0        0 ens8
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall#
```

Fig: Deleting the default routes through virbr1 and virbr2 .

```
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# echo "1">/proc/sys/net/ipv4/ip_forward
```

Fig: Enabling forwarding of traffic in Firewall

----------firewall-configuration done----------

----------host2 configuration----------

```
host2@host2-Standard-PC-i440FX-PIIX-1996:~$ ifconfig
ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.101.253  netmask 255.255.255.0  broadcast 192.168.101.255
        inet6 fe80::8fb:7feb:9979:cac0  prefixlen 64  scopeid 0x20<link>
        ether 52:54:00:df:63:44  txqueuelen 1000  (Ethernet)
        RX packets 60484  bytes 62539523 (62.5 MB)
        RX errors 0  dropped 1736  overruns 0  frame 0
        TX packets 28047  bytes 52501221 (52.5 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 65007

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 1753  bytes 175573 (175.5 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1753  bytes 175573 (175.5 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

Fig: Showing IP address of host2 interface ens3 in network 2

```
host2@host2-Standard-PC-i440FX-PIIX-1996:~$ sudo su
[sudo] password for host2:
root@host2-Standard-PC-i440FX-PIIX-1996:/home/host2# route del default
root@host2-Standard-PC-i440FX-PIIX-1996:/home/host2# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 ens3
192.168.100.0   192.168.101.142 255.255.255.0   UG    0      0        0 ens3
192.168.101.0   0.0.0.0         255.255.255.0   U     100    0        0 ens3
root@host2-Standard-PC-i440FX-PIIX-1996:/home/host2#
```

Fig: Routing table of host 2 after deleting the default route to gateway (virbr2) and adding new route of directing packets from host2 to host1 through firewall.

----------host2-configuration done----------

----------Checking The Connectivity Of Our Topology----------

```
root@host1-Standard-PC-i440FX-PIIX-1996:/home/host1# ping 192.168.101.253
PING 192.168.101.253 (192.168.101.253) 56(84) bytes of data.
64 bytes from 192.168.101.253: icmp_seq=1 ttl=63 time=1.99 ms
64 bytes from 192.168.101.253: icmp_seq=2 ttl=63 time=1.88 ms
64 bytes from 192.168.101.253: icmp_seq=3 ttl=63 time=1.81 ms
64 bytes from 192.168.101.253: icmp_seq=4 ttl=63 time=1.67 ms
64 bytes from 192.168.101.253: icmp_seq=5 ttl=63 time=1.69 ms
64 bytes from 192.168.101.253: icmp_seq=6 ttl=63 time=1.97 ms
64 bytes from 192.168.101.253: icmp_seq=7 ttl=63 time=1.38 ms
64 bytes from 192.168.101.253: icmp_seq=8 ttl=63 time=1.27 ms
64 bytes from 192.168.101.253: icmp_seq=9 ttl=63 time=1.66 ms
64 bytes from 192.168.101.253: icmp_seq=10 ttl=63 time=1.85 ms
```

Fig: Pinging from host1 to host 2

```
root@firewall-Standard-PC-i440FX-PIIX-1996:/home/firewall# tcpdump -i ens3
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on ens3, link-type EN10MB (Ethernet), capture size 262144 bytes
18:26:37.694210 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:39.710245 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:41.694238 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:43.710251 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:45.694197 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:47.368326 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo request, id
30, seq 1, length 64
18:26:47.369597 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo reply, id 30
, seq 1, length 64
18:26:47.710269 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:f3:56:8f.8002, length 43
18:26:48.369697 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo request, id
30, seq 2, length 64
18:26:48.370632 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo reply, id 30
, seq 2, length 64
18:26:49.370830 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo request, id
30, seq 3, length 64
18:26:49.371738 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo reply, id 30
```

Fig: Showing the ping packets going through ens3 interface of firewall (confirmed through tcpdump)

```
root@host2-Standard-PC-i440FX-PIIX-1996:/home/host2# ping 192.168.100.155
PING 192.168.100.155 (192.168.100.155) 56(84) bytes of data.
64 bytes from 192.168.100.155: icmp_seq=1 ttl=63 time=1.60 ms
64 bytes from 192.168.100.155: icmp_seq=2 ttl=63 time=1.68 ms
64 bytes from 192.168.100.155: icmp_seq=3 ttl=63 time=1.76 ms
64 bytes from 192.168.100.155: icmp_seq=4 ttl=63 time=1.78 ms
64 bytes from 192.168.100.155: icmp_seq=5 ttl=63 time=1.85 ms
64 bytes from 192.168.100.155: icmp_seq=6 ttl=63 time=1.53 ms
^C
```

Fig: Pinging from host 2 to host 1

```
18:31:24.958435 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo request, id
13, seq 3, length 64
18:31:24.959313 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo reply, id 13
, seq 3, length 64
18:31:25.960439 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo request, id
13, seq 4, length 64
18:31:25.961222 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo reply, id 13
, seq 4, length 64
18:31:26.659907 STP 802.1d, Config, Flags [none], bridge-id 8000.fe:54:00:28:be:57.8002, length 43
18:31:26.962458 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo request, id
13, seq 5, length 64
18:31:26.963304 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo reply, id 13
, seq 5, length 64
18:31:27.964362 IP host2-Standard-PC-i440FX-PIIX-1996.network2 > 192.168.100.155: ICMP echo request, id
13, seq 6, length 64
18:31:27.965186 IP 192.168.100.155 > host2-Standard-PC-i440FX-PIIX-1996.network2: ICMP echo reply, id 13
, seq 6, length 64
18:31:27.983595 ARP, Request who-has firewall-Standard-PC-i440FX-PIIX-1996.network2 tell host2-Standard-
```

Fig: Showing the ping packets going through ens8 interface of firewall (confirmed through tcpdump)

—----------Topology-Setup-Done—----------

Hardcoding a Rule-Set:

Task 1:

Static rules added for Task 1:

```
def icmp_static_rules(soc, src_ip, dst_ip, src_mac, dst_mac, packet):

    ################## STATIC RULES ###############
    if dst_ip == '192.168.100.155':
        print("Dropping packet with src ip" , src_ip, " access denied")

    elif dst_mac == '525400df6344' and src_ip == '192.168.100.156':

        print("Dropping packet with src mac", src_mac)
    else:
        soc.sendall(packet[0])
        print("Packet is allowed to pass to VM2 with src ip ", src_ip, "and dst ip number", dst_ip)


def tcp_udp_static_rules(socket, src_ip, dst_ip, src_port, dst_port, src_mac, dst_mac, packet):

    ################## STATIC RULES ###############
    if dst_ip == '192.168.100.155' or dst_ip == 80:
        print("Dropping packet with src ip" , src_ip, " access denied")

    else:
        socket.sendall(packet[0])
        print("Packet is allowed to pass to VM2 with src ip ", src_ip, "and dst port number ",
dst_port)
```

—----------Showing for Allow/Block of ICMP packet----------

ICMP:

Host 1:

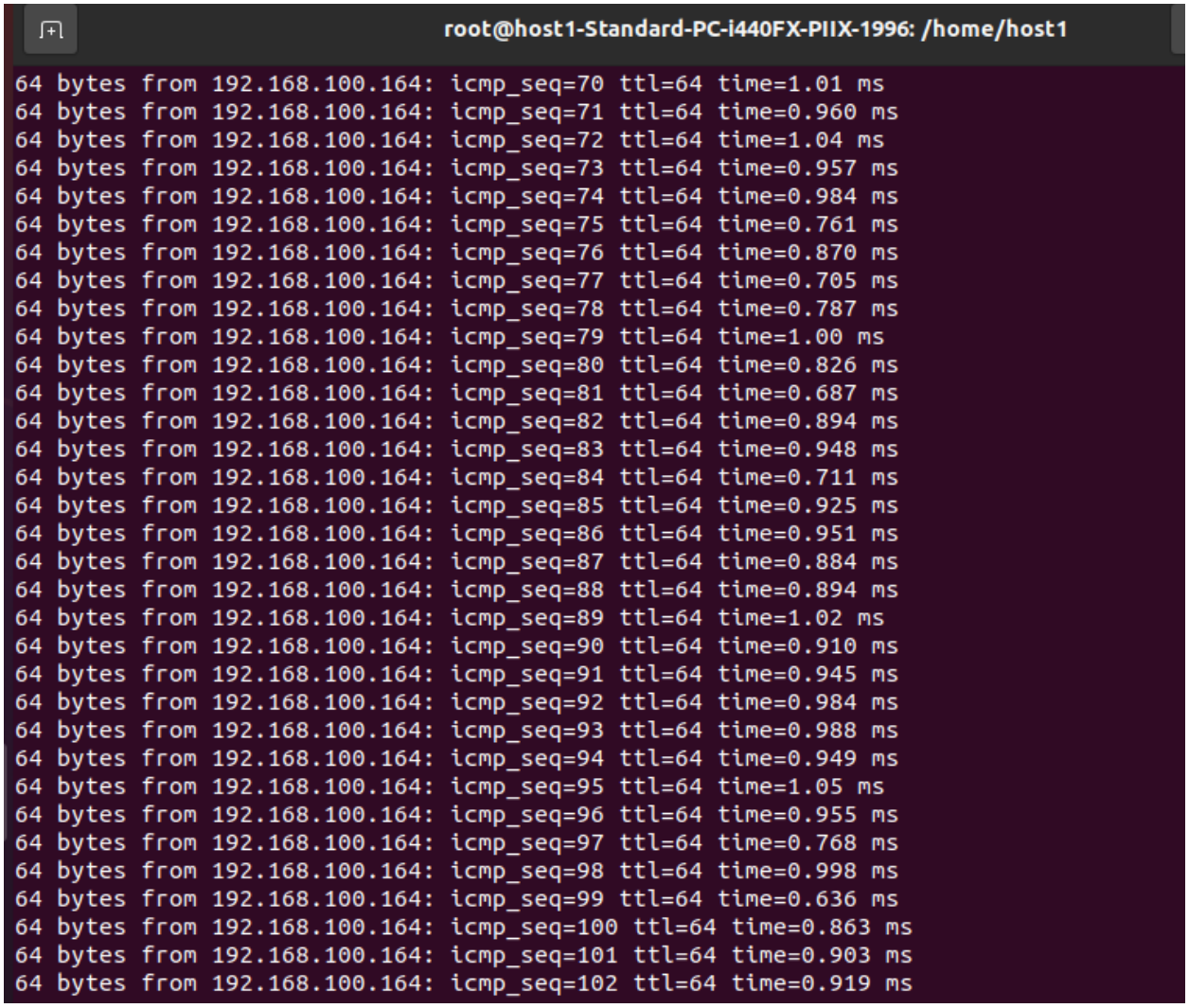


Fig: Sending ICMP request from Host1

Firewall is passing ECHO Request Packets from network 192.168.100.0 to 192.168.101.0 as it is based on the static rules.

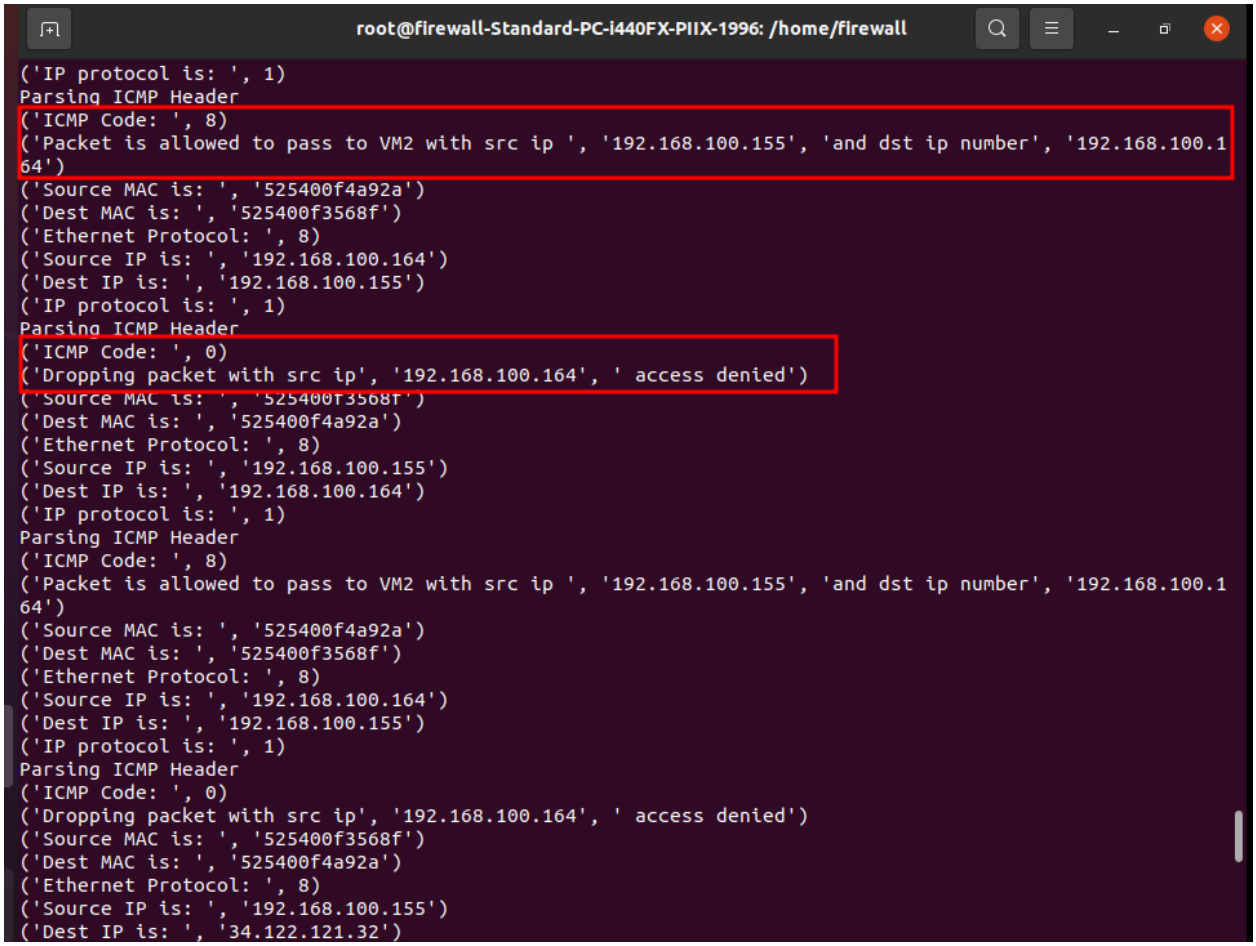


Fig: Showing ICMP request packets (Code:8) to pass. Blocking the ICMP response packets generated at ens3 interface of Firewall(Code:0)

Host 2:

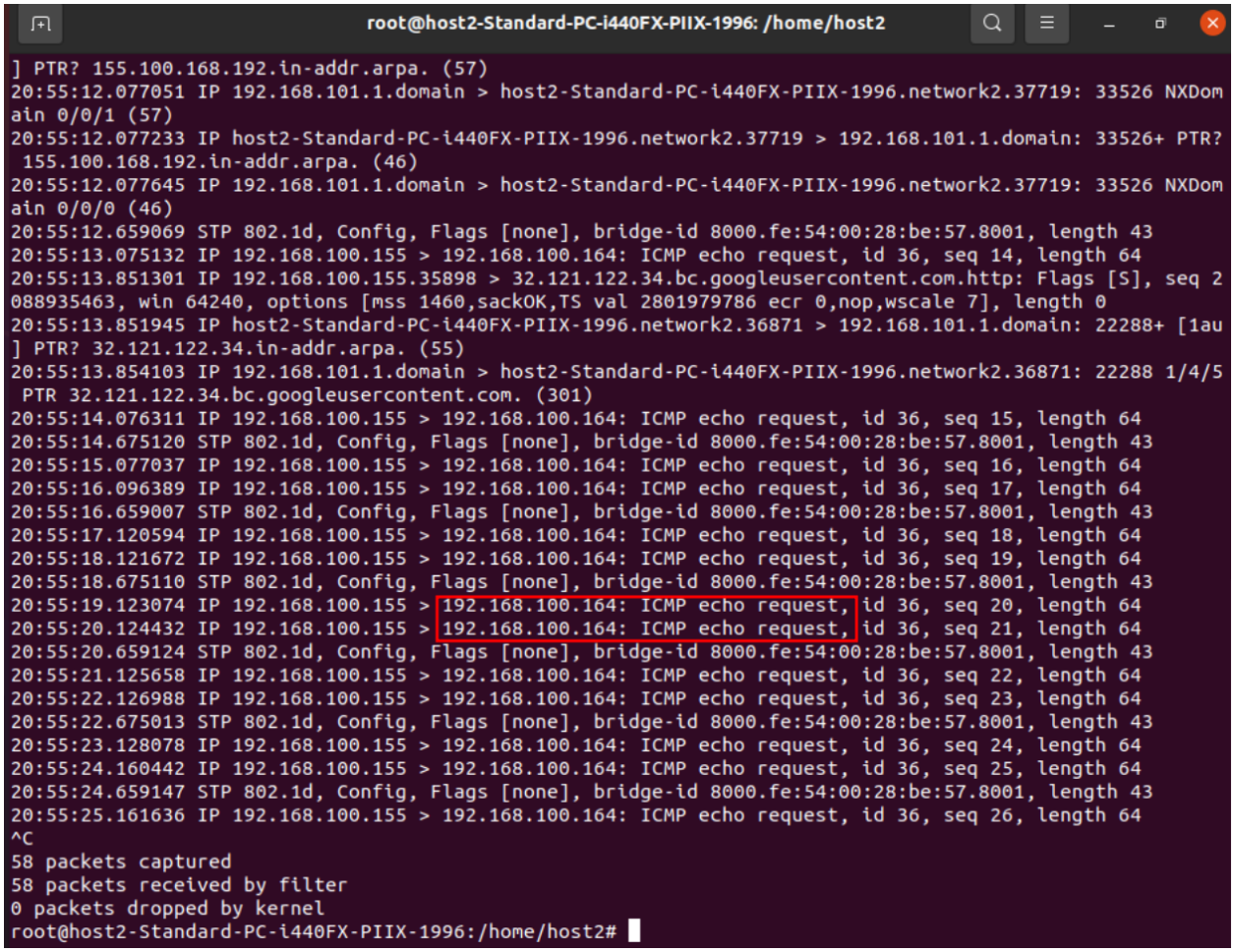
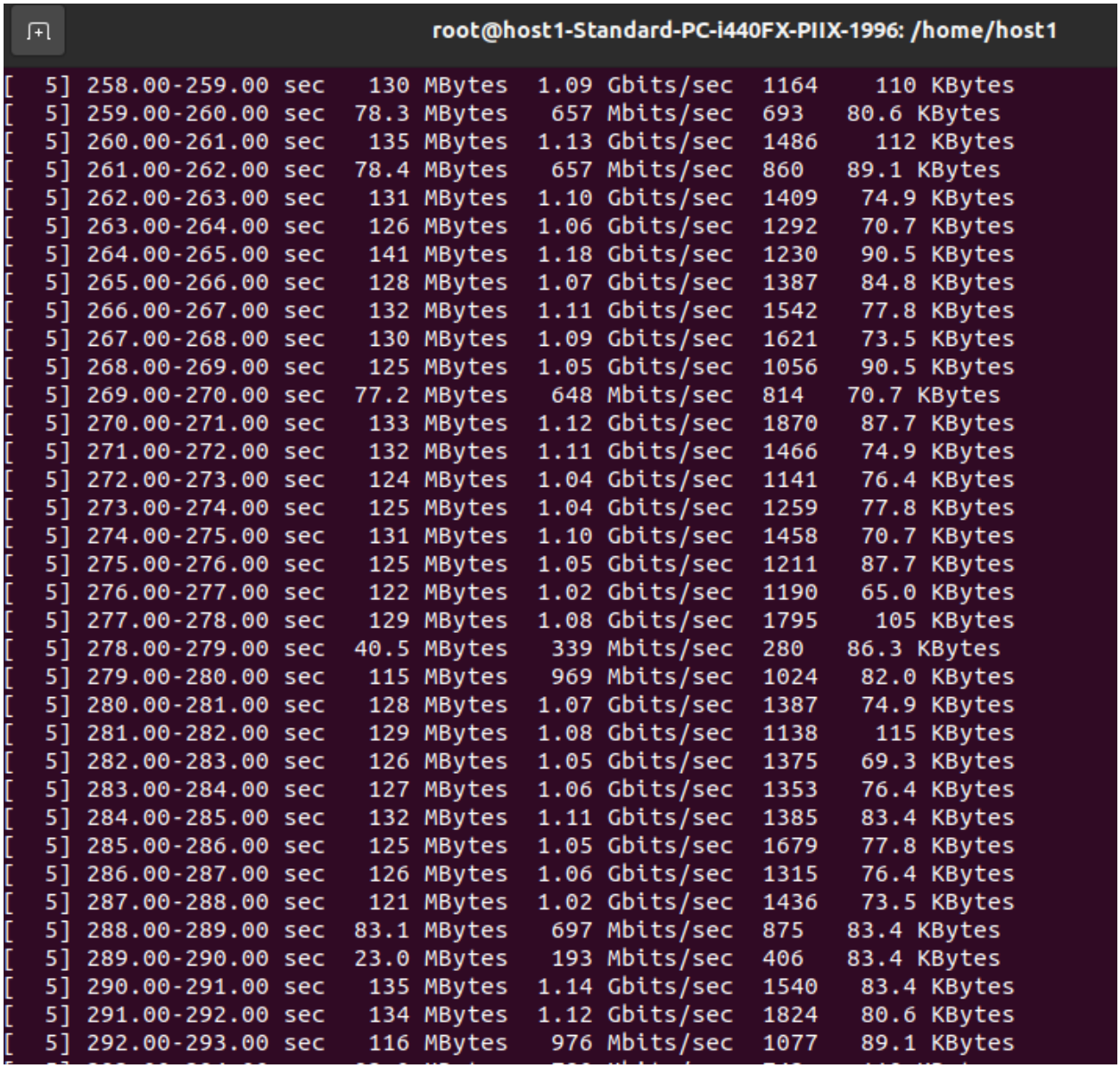


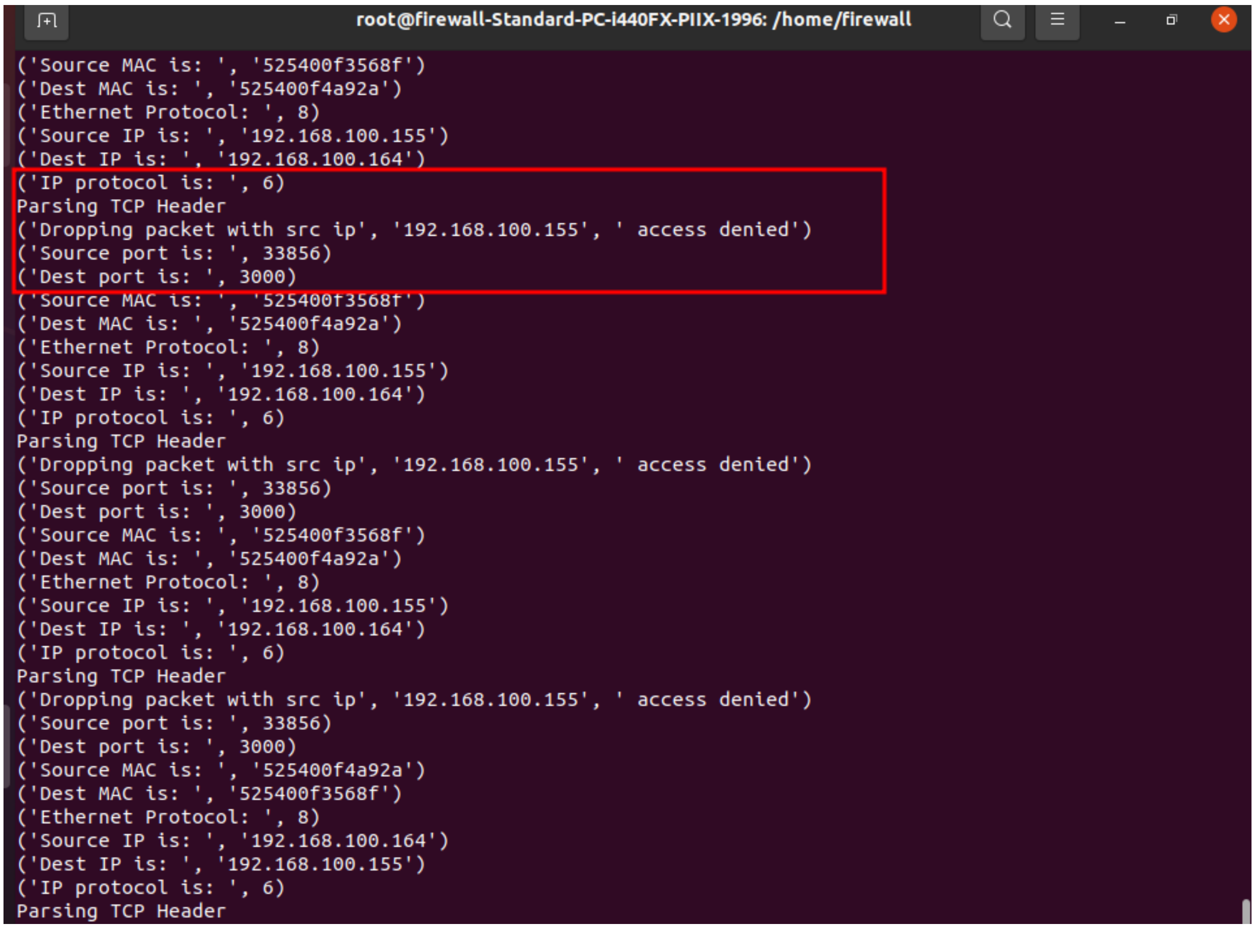
Fig: Showing ICMP Echo Requests has only been received at Host2 [Proper working of Static Rules of Firewall confirmed]

—----------Allow/Drop-TCP/UDP—----------

Host 1: Generated TCP traffic using iperf :

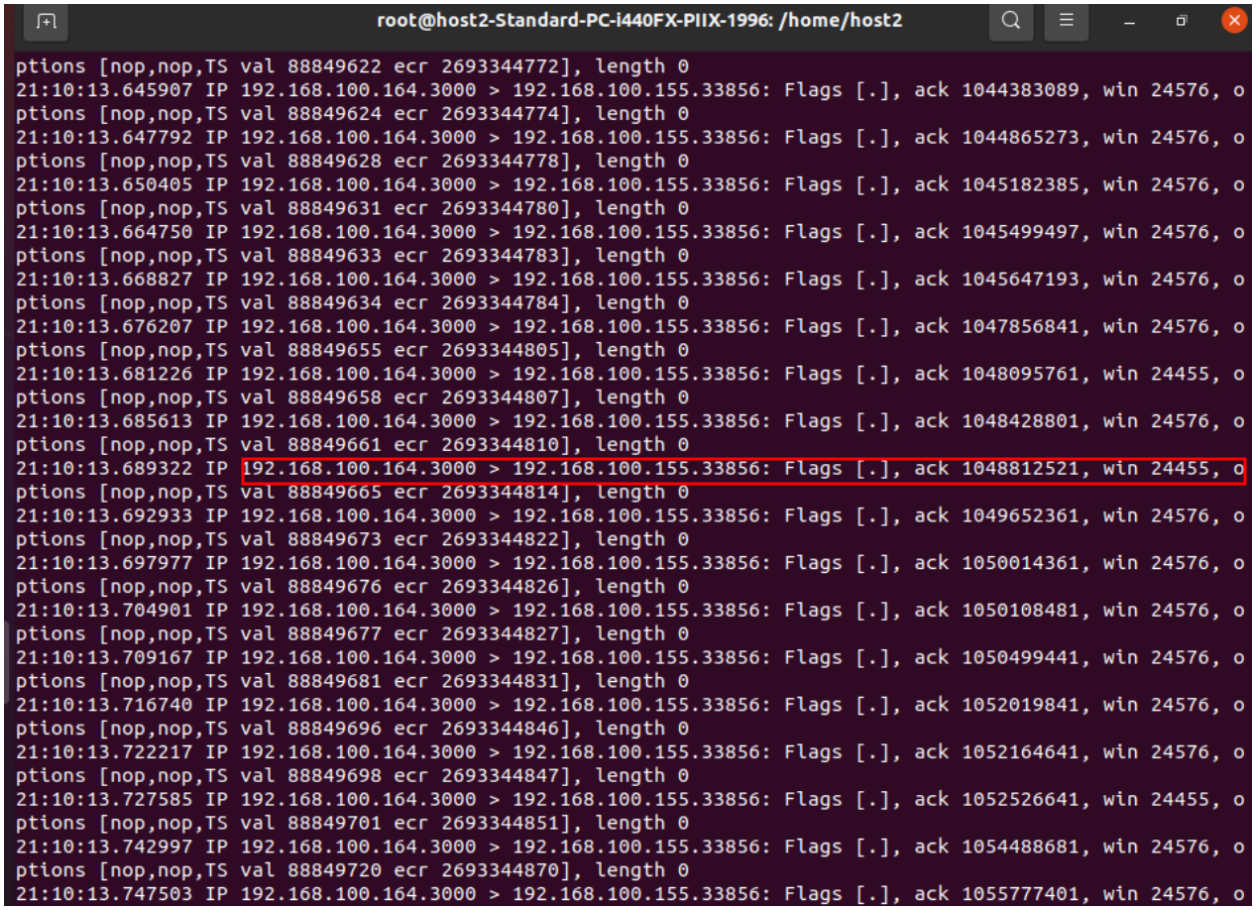

root@host1-Standard-PC-i440FX-PIIX-1996: /home/host1
[5] 258.00-259.00 sec 130 MBytes 1.09 Gbits/sec 1164 110 KBytes
[5] 259.00-260.00 sec 78.3 MBytes 657 Mbits/sec 693 80.6 KBytes
[5] 260.00-261.00 sec 135 MBytes 1.13 Gbits/sec 1486 112 KBytes
[5] 261.00-262.00 sec 78.4 MBytes 657 Mbits/sec 860 89.1 KBytes
[5] 262.00-263.00 sec 131 MBytes 1.10 Gbits/sec 1409 74.9 KBytes
[5] 263.00-264.00 sec 126 MBytes 1.06 Gbits/sec 1292 70.7 KBytes
[5] 264.00-265.00 sec 141 MBytes 1.18 Gbits/sec 1230 90.5 KBytes
[5] 265.00-266.00 sec 128 MBytes 1.07 Gbits/sec 1387 84.8 KBytes
[5] 266.00-267.00 sec 132 MBytes 1.11 Gbits/sec 1542 77.8 KBytes
[5] 267.00-268.00 sec 130 MBytes 1.09 Gbits/sec 1621 73.5 KBytes
[5] 268.00-269.00 sec 125 MBytes 1.05 Gbits/sec 1056 90.5 KBytes
[5] 269.00-270.00 sec 77.2 MBytes 648 Mbits/sec 814 70.7 KBytes
[5] 270.00-271.00 sec 133 MBytes 1.12 Gbits/sec 1870 87.7 KBytes
[5] 271.00-272.00 sec 132 MBytes 1.11 Gbits/sec 1466 74.9 KBytes
[5] 272.00-273.00 sec 124 MBytes 1.04 Gbits/sec 1141 76.4 KBytes
[5] 273.00-274.00 sec 125 MBytes 1.04 Gbits/sec 1259 77.8 KBytes
[5] 274.00-275.00 sec 131 MBytes 1.10 Gbits/sec 1458 70.7 KBytes
[5] 275.00-276.00 sec 125 MBytes 1.05 Gbits/sec 1211 87.7 KBytes
[5] 276.00-277.00 sec 122 MBytes 1.02 Gbits/sec 1190 65.0 KBytes
[5] 277.00-278.00 sec 129 MBytes 1.08 Gbits/sec 1795 105 KBytes
[5] 278.00-279.00 sec 40.5 MBytes 339 Mbits/sec 280 86.3 KBytes
[5] 279.00-280.00 sec 115 MBytes 969 Mbits/sec 1024 82.0 KBytes
[5] 280.00-281.00 sec 128 MBytes 1.07 Gbits/sec 1387 74.9 KBytes
[5] 281.00-282.00 sec 129 MBytes 1.08 Gbits/sec 1138 115 KBytes
[5] 282.00-283.00 sec 126 MBytes 1.05 Gbits/sec 1375 69.3 KBytes
[5] 283.00-284.00 sec 127 MBytes 1.06 Gbits/sec 1353 76.4 KBytes
[5] 284.00-285.00 sec 132 MBytes 1.11 Gbits/sec 1385 83.4 KBytes
[5] 285.00-286.00 sec 125 MBytes 1.05 Gbits/sec 1679 77.8 KBytes
[5] 286.00-287.00 sec 126 MBytes 1.06 Gbits/sec 1315 76.4 KBytes
[5] 287.00-288.00 sec 121 MBytes 1.02 Gbits/sec 1436 73.5 KBytes
[5] 288.00-289.00 sec 83.1 MBytes 697 Mbits/sec 875 83.4 KBytes
[5] 289.00-290.00 sec 23.0 MBytes 193 Mbits/sec 406 83.4 KBytes
[5] 290.00-291.00 sec 135 MBytes 1.14 Gbits/sec 1540 83.4 KBytes
[5] 291.00-292.00 sec 134 MBytes 1.12 Gbits/sec 1824 80.6 KBytes
[5] 292.00-293.00 sec 116 MBytes 976 Mbits/sec 1077 89.1 KBytes

Firewall is passing IPERF packets sent by Host1 to Host2. But Blocks IPERF packets from Host2 to Host1 based on static rules- Any TCP packets coming from Host2 (192.168.100.155) is blocked.



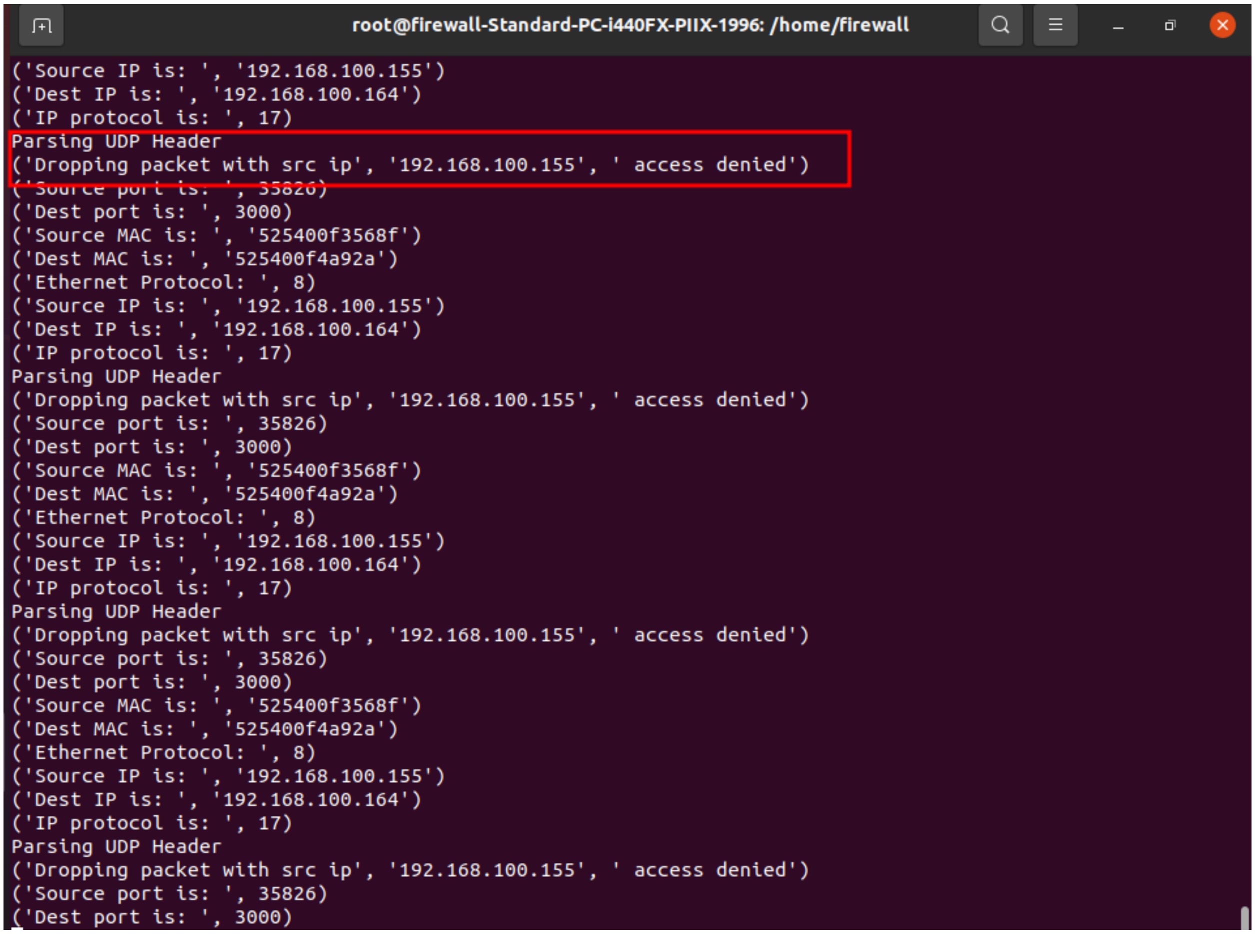
Host 2:

TCP packets sent by the host 1 are allowed to pass through the firewall to the host 2 traced through TCP dump at Host2 interface.



—----------Allow/Drop-UDP packets—----------

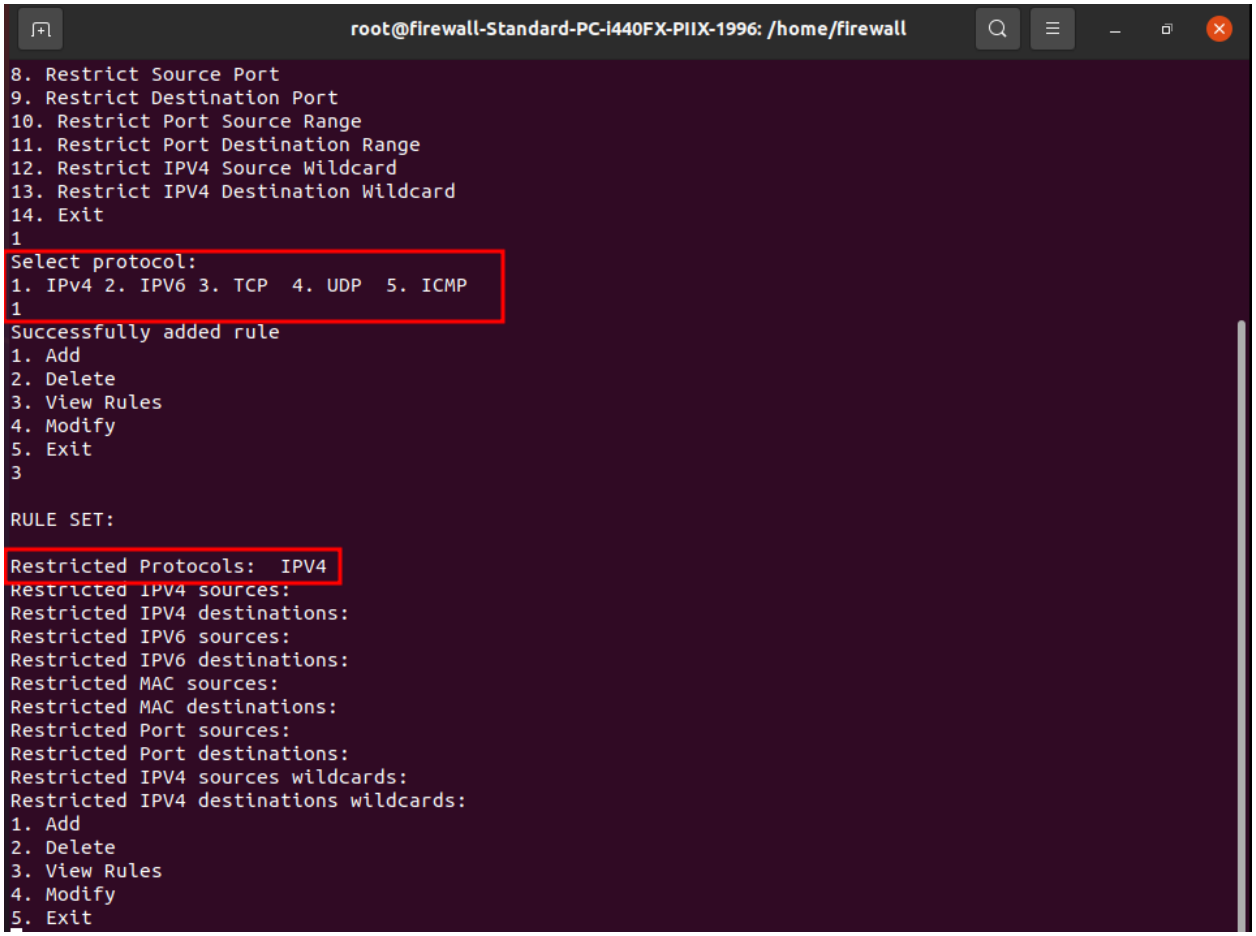
Similarly UDP packets coming from Host2 to Host1 are blocked based on static rules at the Firewall.



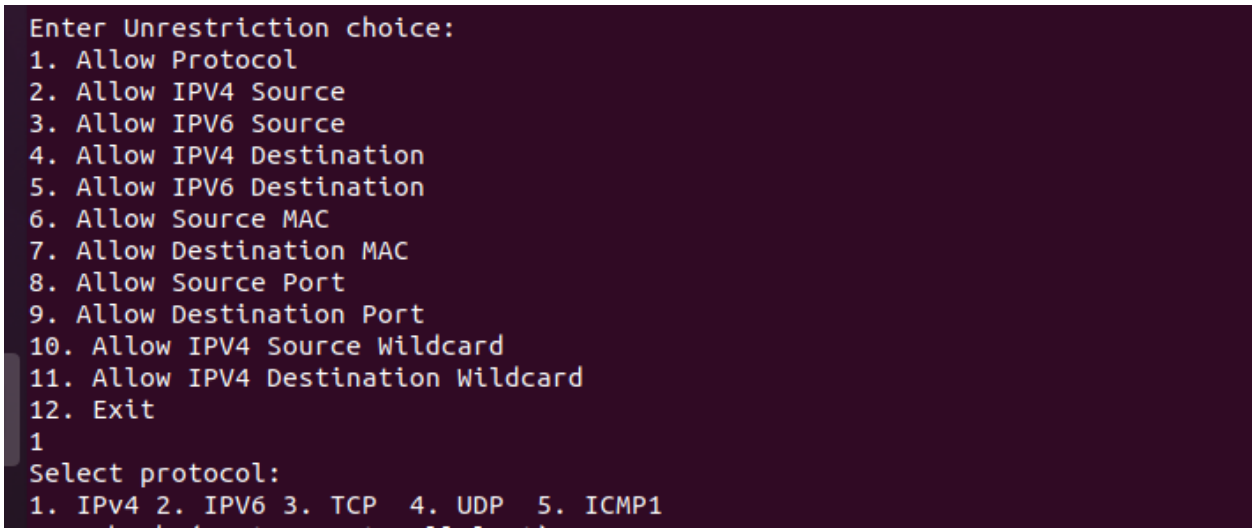
****
****

Task 2: Extending the rule set and its operation on Firewall

Task1 extended to interactive CLI for add, delete, modify and show rules. We have used a json file for inserting the rules from where the firewall python program reads.
Showcasing Firewall Interactive CLI:

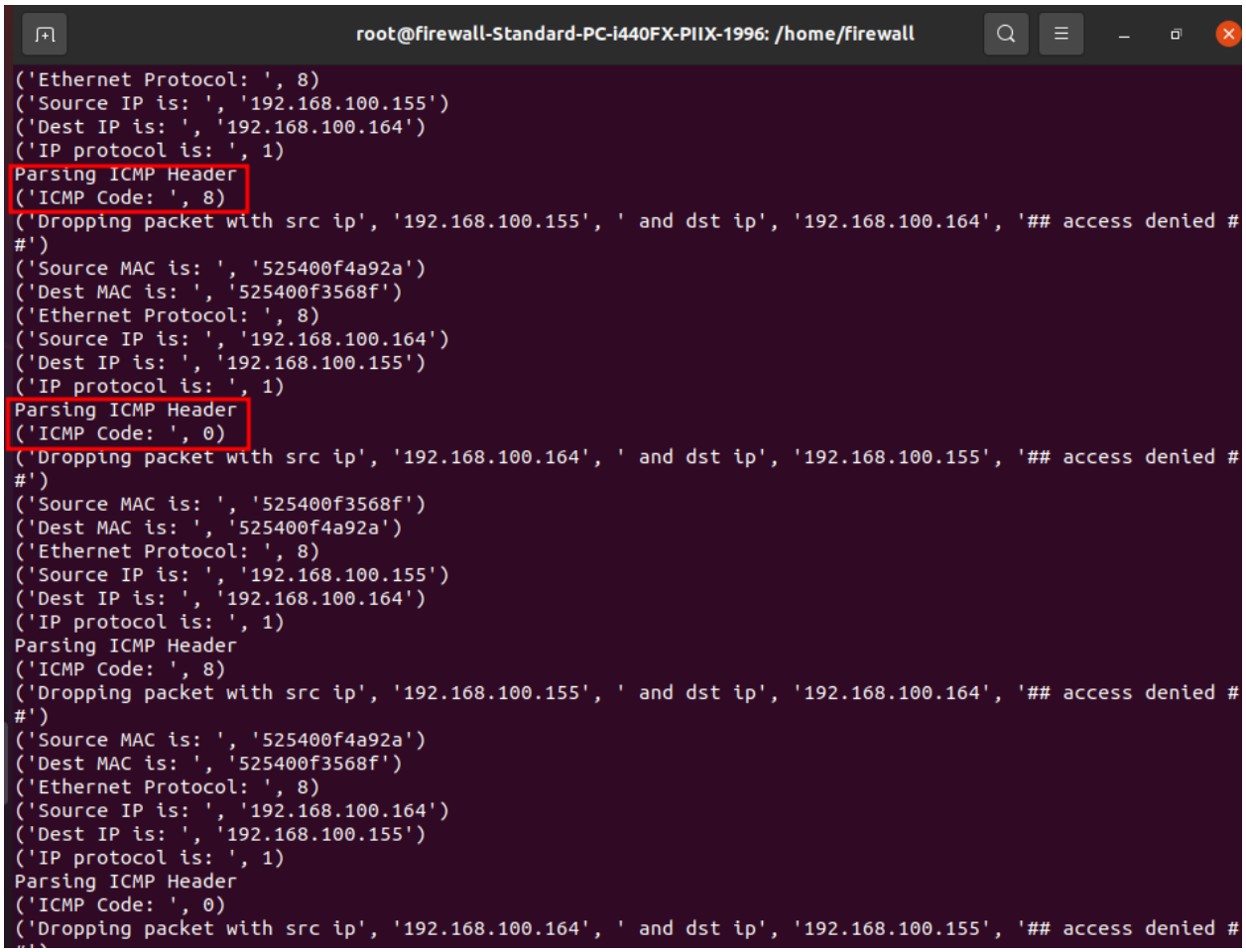


Wider View Of Deleting Rules/ Allowing Restrictions Feature:



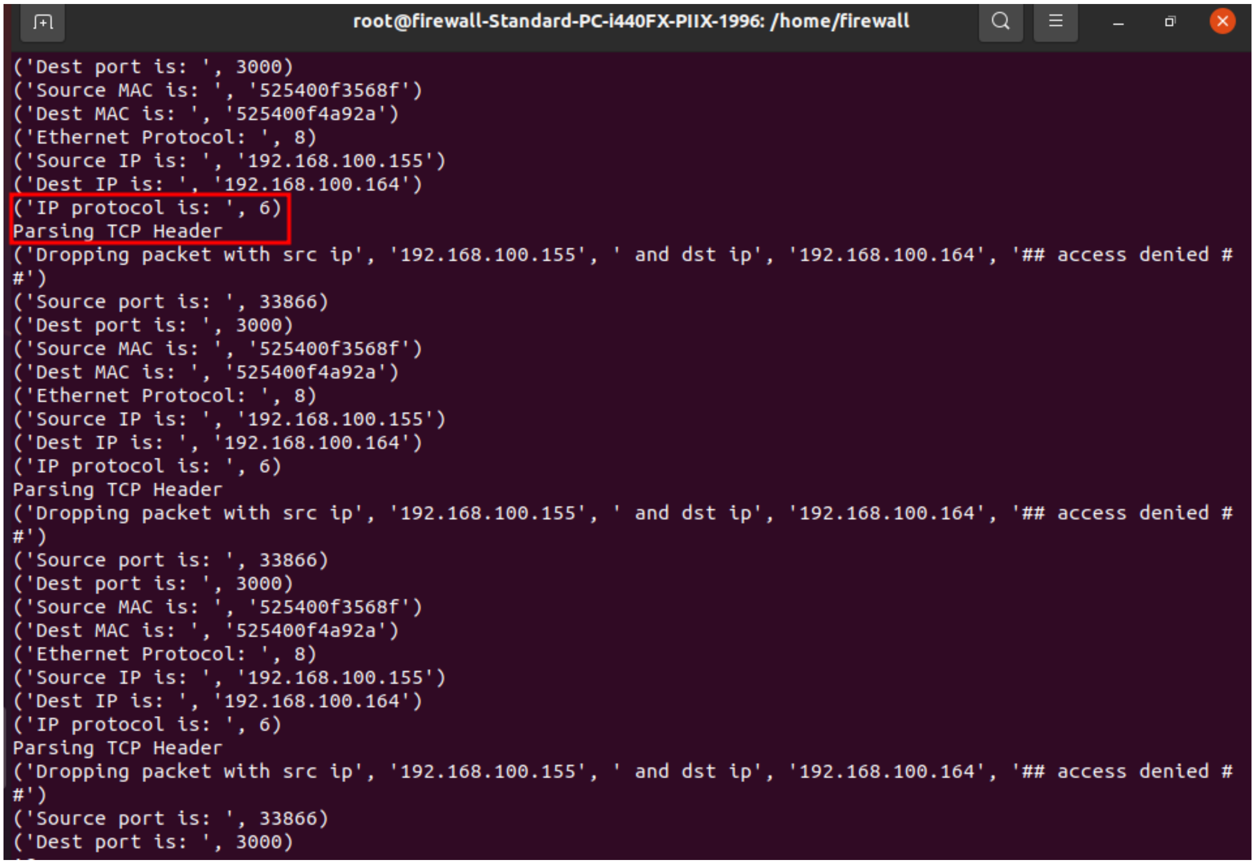
Our Firewall is able to parse ICMP, TCP and UDP packets and support rule sets for all layers.

----------Adding Rule Set for IPV4----------
Since we added IPv4 as a restricted protocol, all the packets in the following screenshots are blocked.
ICMP:



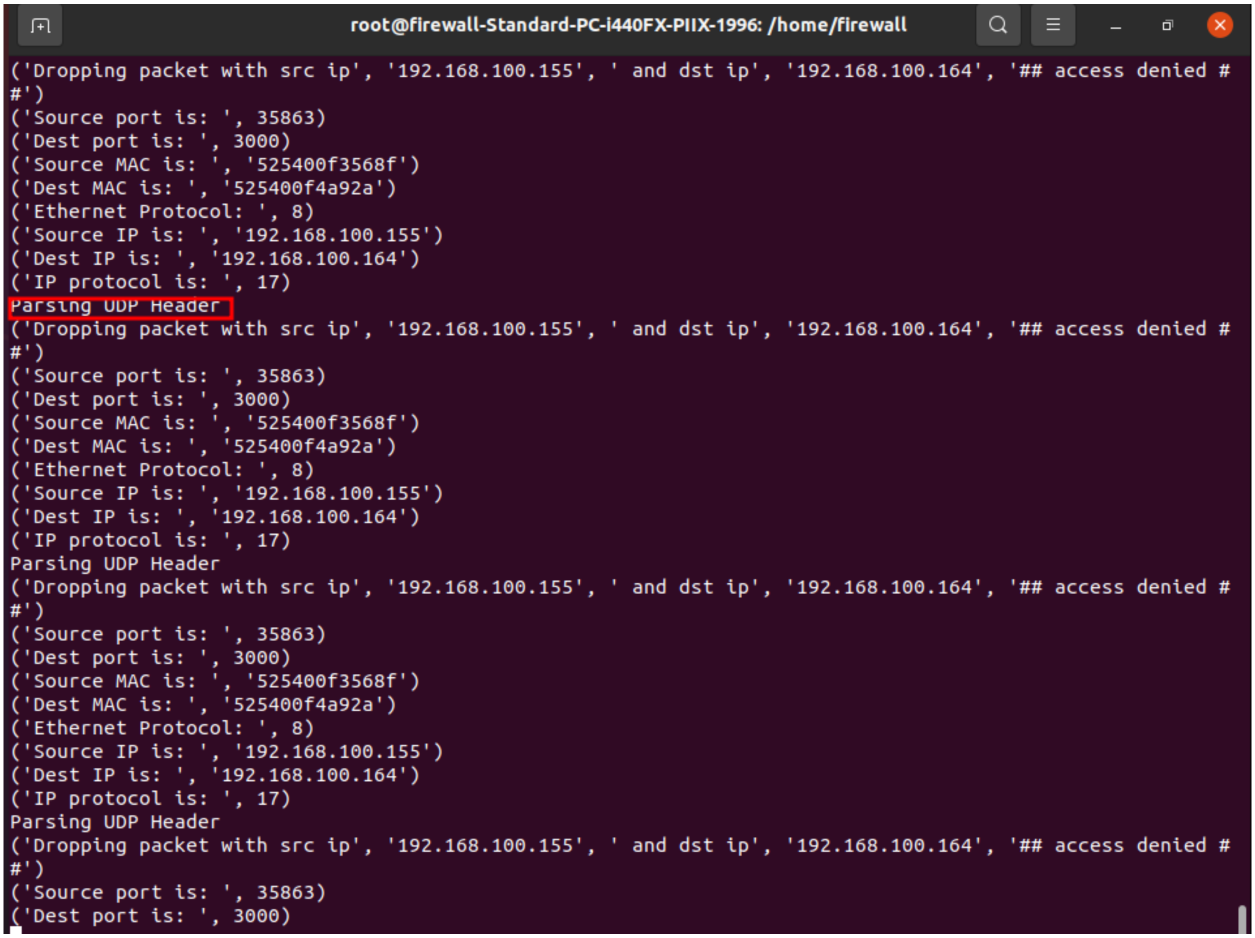
Both ICMP Request Packet(sent from Host1) and ICMP Response Packets generated at Firewall are blocked.

TCP:



TCP packets generated through IPERF are blocked at the Firewall and dropped due to restriction on IPV4 protocol.

UDP:



```
root@firewall-Standard-PC-i440FX-PIIX-1996: /home/firewall
('Dropping packet with src ip', '192.168.100.155', ' and dst ip', '192.168.100.164', '## access denied #
#')
('Source port is: ', 35863)
('Dest port is: ', 3000)
('Source MAC is: ', '525400f3568f')
('Dest MAC is: ', '525400f4a92a')
('Ethernet Protocol: ', 8)
('Source IP is: ', '192.168.100.155')
('Dest IP is: ', '192.168.100.164')
('IP protocol is: ', 17)
Parsing UDP Header
('Dropping packet with src ip', '192.168.100.155', ' and dst ip', '192.168.100.164', '## access denied #
#')
('Source port is: ', 35863)
('Dest port is: ', 3000)
('Source MAC is: ', '525400f3568f')
('Dest MAC is: ', '525400f4a92a')
('Ethernet Protocol: ', 8)
('Source IP is: ', '192.168.100.155')
('Dest IP is: ', '192.168.100.164')
('IP protocol is: ', 17)
Parsing UDP Header
('Dropping packet with src ip', '192.168.100.155', ' and dst ip', '192.168.100.164', '## access denied #
#')
('Source port is: ', 35863)
('Dest port is: ', 3000)
('Source MAC is: ', '525400f3568f')
('Dest MAC is: ', '525400f4a92a')
('Ethernet Protocol: ', 8)
('Source IP is: ', '192.168.100.155')
('Dest IP is: ', '192.168.100.164')
('IP protocol is: ', 17)
Parsing UDP Header
('Dropping packet with src ip', '192.168.100.155', ' and dst ip', '192.168.100.164', '## access denied #
#')
('Source port is: ', 35863)
('Dest port is: ', 3000)
```

UDP packets generated through IPERF are blocked at the Firewall and dropped due to restriction on IPV4 protocol.

—---------------------Adding Rule Set for MAC—---

Blocking via MAC Address:

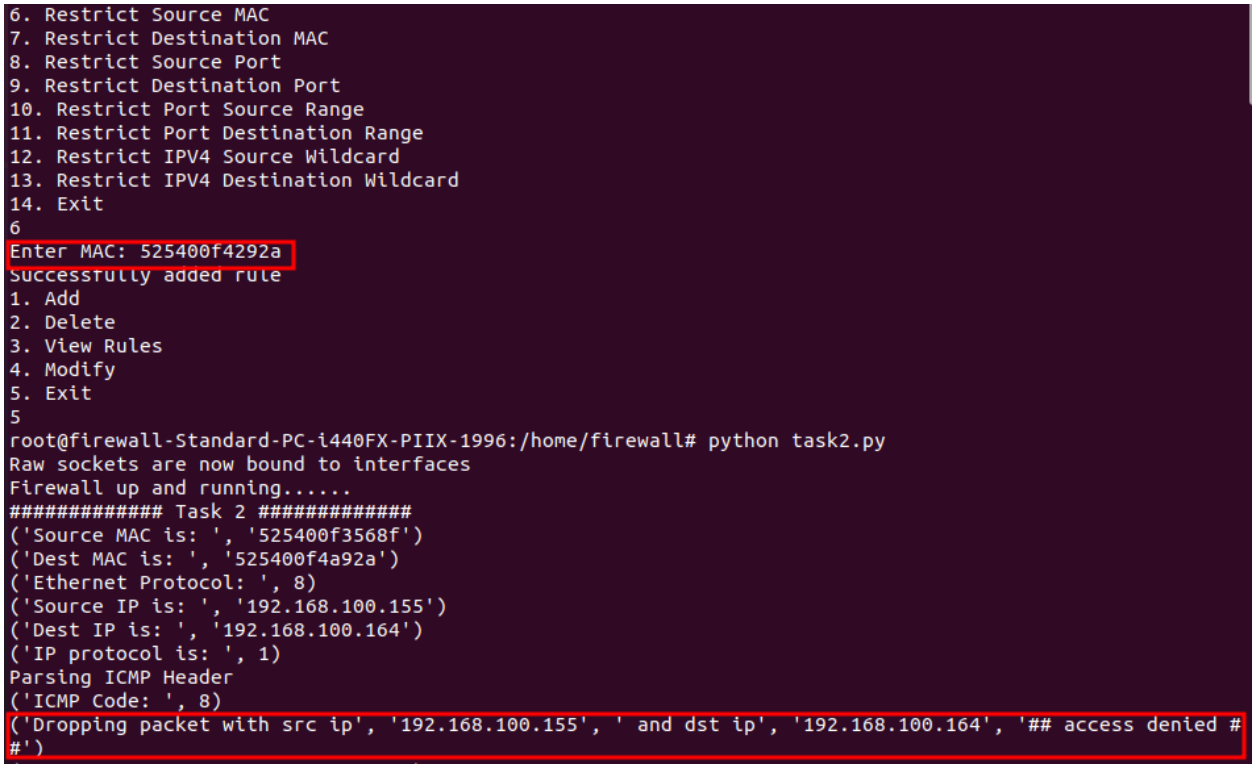


Fig: User entered a specified MAC address to be blocked as a new Rule. Successfully a packet with a given MAC address has also been blocked generated through PING.

*****Firewall supporting the aggregation of a rule - IPV4 wildcard & Range of Port Numbers*****

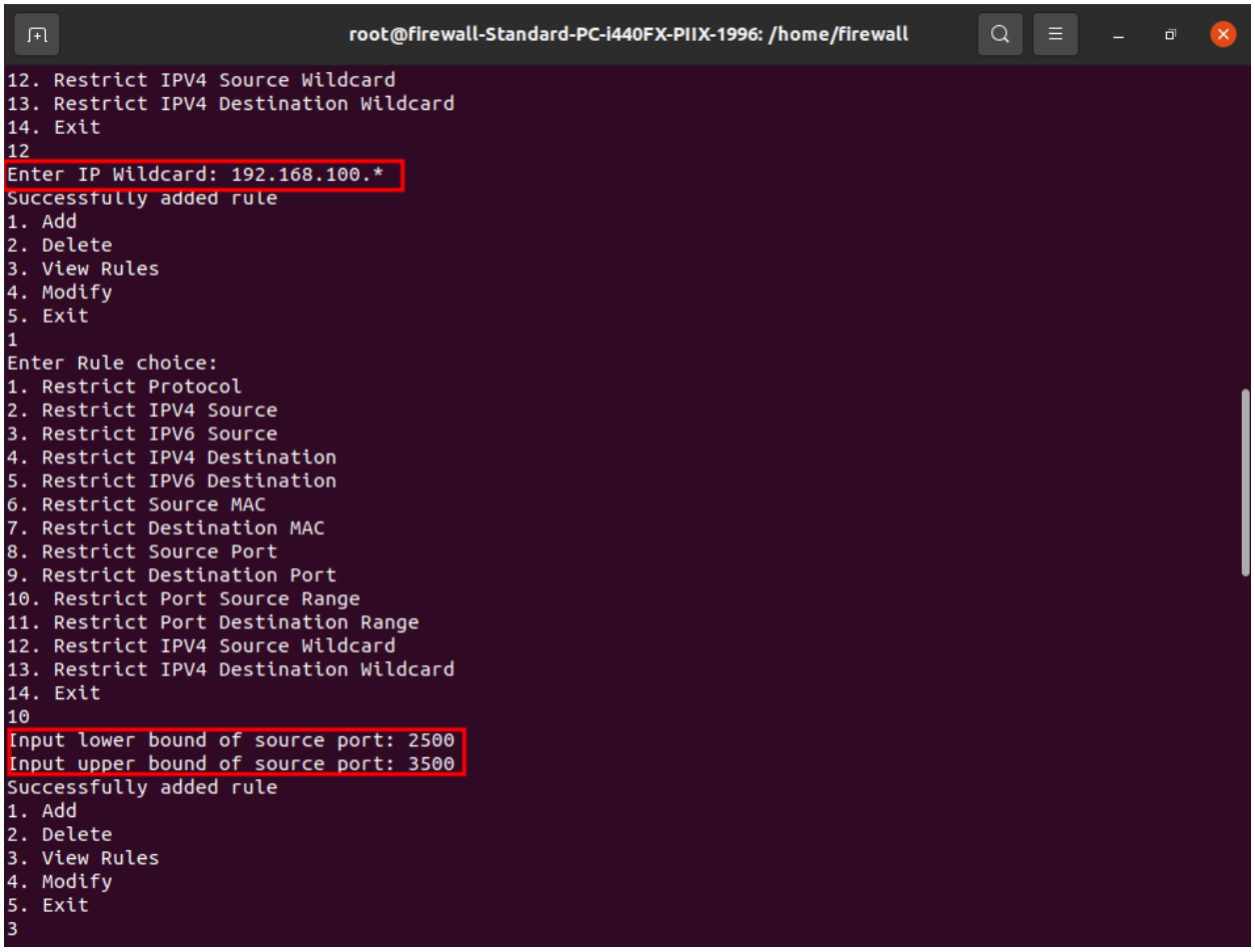
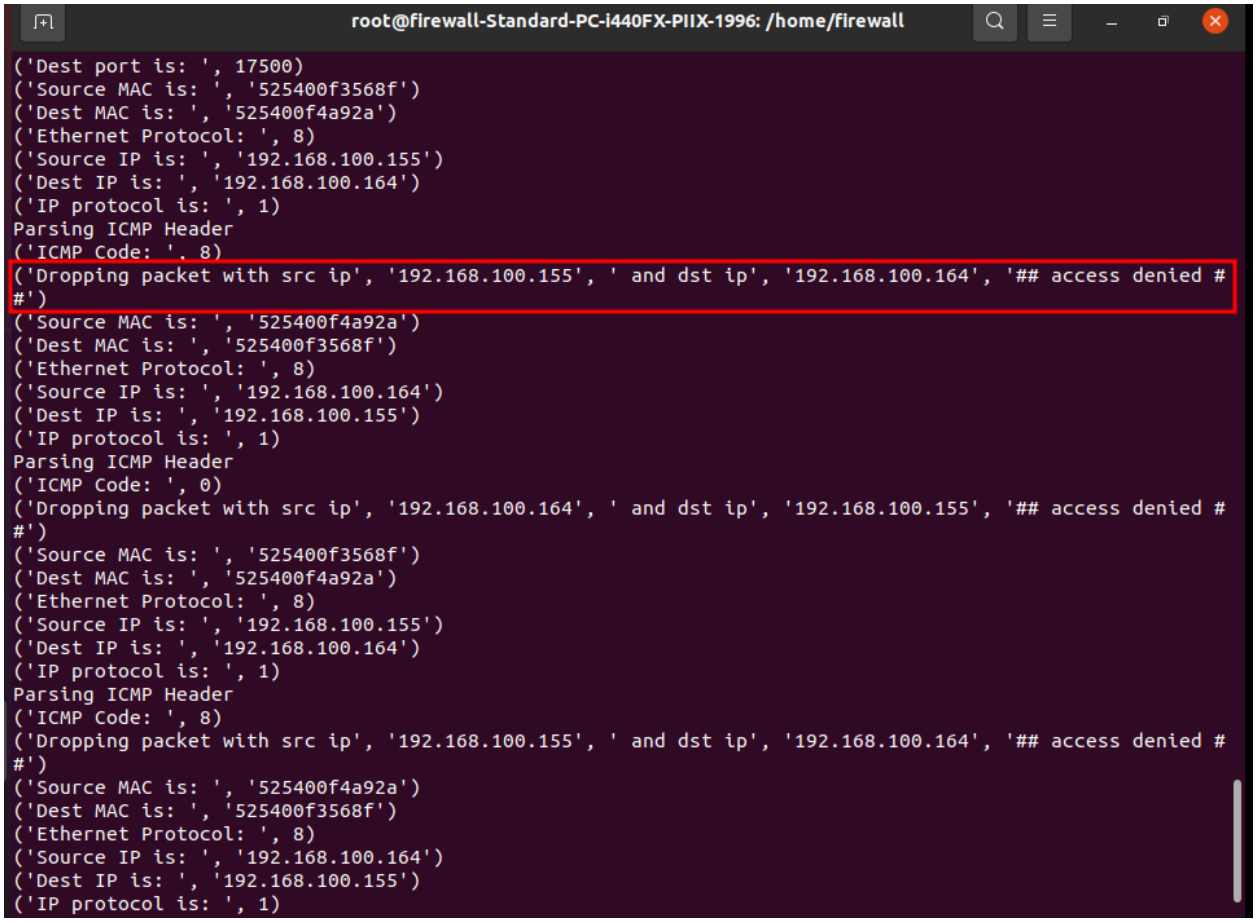


Fig: Our CLI supporting entry of IPV4 Wildcards and Port Range Based Rule Decision at Firewall.

After adding IPv4 wildcard entry, all the packets from Network 1 –> 192.168.100.0 and Port Ranges between 2500-3500 are dropped at the firewall.



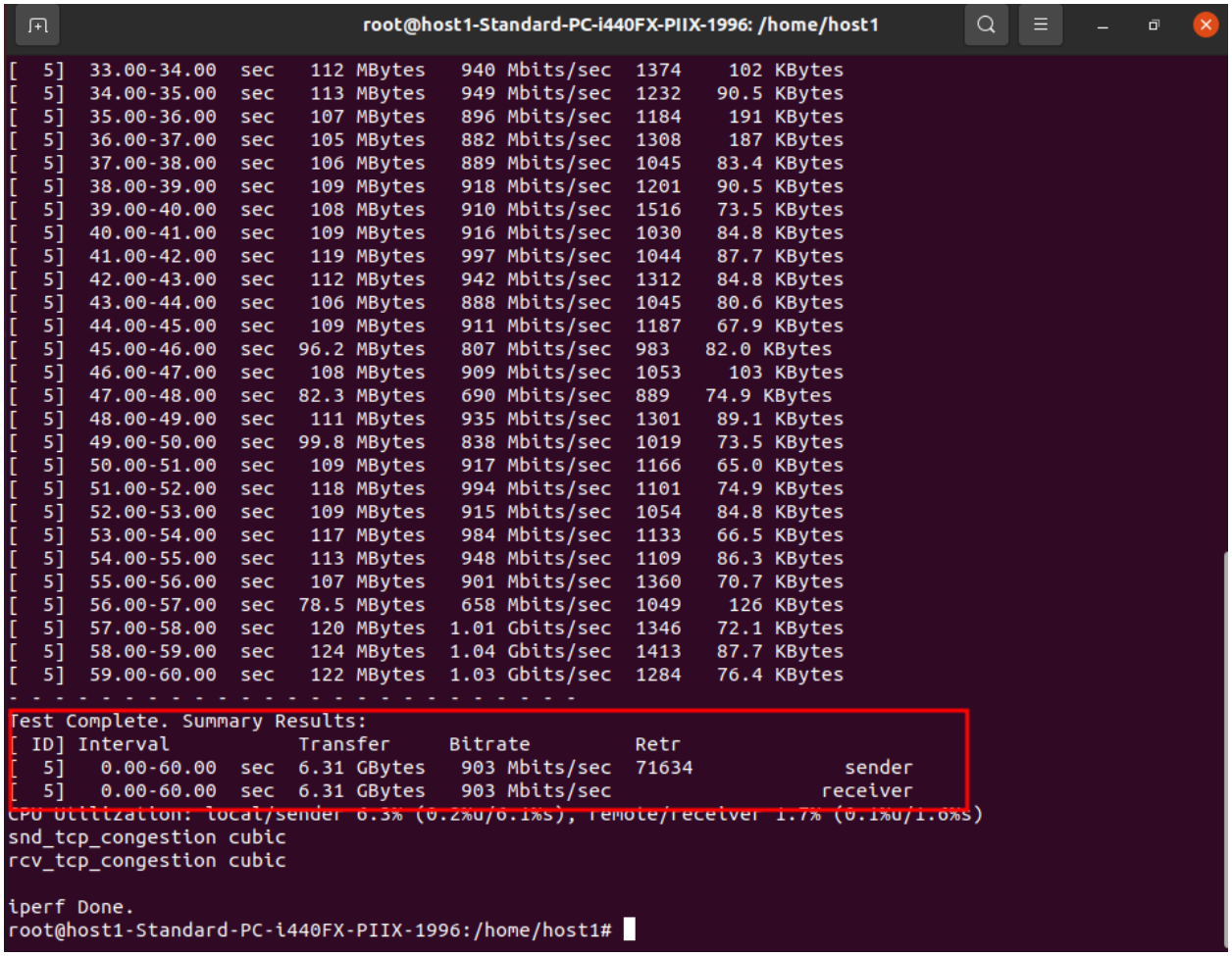
Packets from Network 1 are dropped based on the new rule added.

****

Task 3: Performance examination and improvement

For performance examination:

1. We are considering the TCP protocol since TCP requires handshake and state management and is heavier than UDP.
2. For generating a controlled TCP traffic, we have used the **iperf** tool for 60 secs as shown in the screenshot below.



3. Our firewall reads an incoming packet and stores it in a queue with a predefined size.
4. If the queue is full, the incoming packets are dropped eventually by the firewall and cannot further process the packets.
5. The packets are then taken from the queue and further processed by the firewall based on the rules present in the rules.json file.
6. To improve the performance, we have used one thread to read packets and another thread to process these read packets.

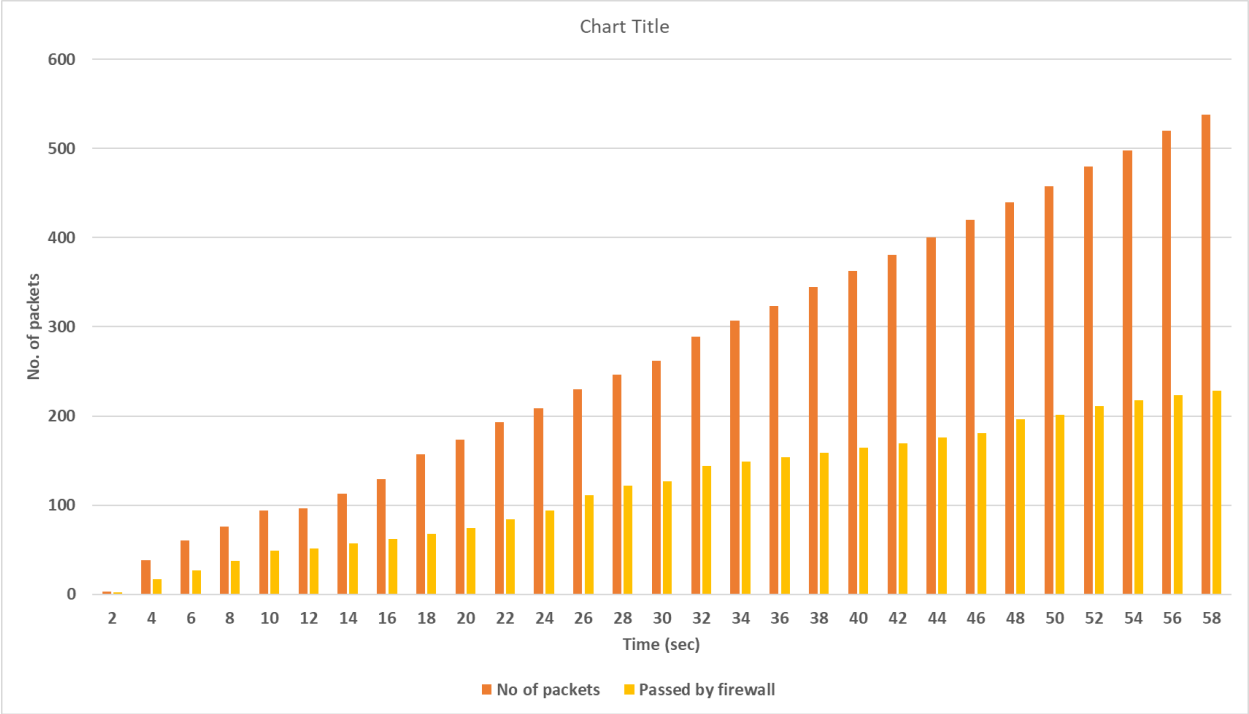
Test 1: Variable factor - Queue Size of firewall
Constant factor - Packet generated time - 60 secs (1 min)
The packets are generated at 0.5 M bits/sec using iperf

```
host1@host1-Standard-PC-i440FX-PIIX-1996:~$ sudo iperf3 -c 192.168.100.164 -p 3000 -f K -V --bandwidth
.5M -t 60
iperf 3.7
Linux host1-Standard-PC-i440FX-PIIX-1996 5.13.0-40-generic #45~20.04.1-Ubuntu SMP Mon Apr 4 09:38:31 UT
 2022 x86_64
Control connection MSS 1448
Time: Tue, 03 May 2022 19:31:07 GMT
Connecting to host 192.168.100.164, port 3000
      Cookie: rxvyeo6z7lobjfzlxfvns2fciw5ux5h4i7b5
      TCP MSS: 1448 (default)
      Target Bitrate: 500000
[  5] local 192.168.100.155 port 33994 connected to 192.168.100.164 port 3000
Starting Test: protocol: TCP, 1 streams, 131072 byte blocks, omitting 0 seconds, 60 second test, tos 0
[ ID] Interval           Transfer     Bitrate         Retr  Cwnd
[  5]   0.00-1.00   sec   128 KBytes   128 KBytes/sec    0    109 KBytes
[  5]   1.00-2.00   sec  0.00 Bytes  0.00 KBytes/sec    0    109 KBytes
[  5]   2.00-3.00   sec   128 KBytes   128 KBytes/sec    0    143 KBytes
[  5]   3.00-4.00   sec  0.00 Bytes  0.00 KBytes/sec    0    143 KBytes
[  5]   4.00-5.00   sec   128 KBytes   128 KBytes/sec    0    136 KBytes
[  5]   5.00-6.00   sec  0.00 Bytes  0.00 KBytes/sec    0    136 KBytes
[  5]   6.00-7.00   sec   128 KBytes   128 KBytes/sec    0    136 KBytes
[  5]   7.00-8.00   sec  0.00 Bytes  0.00 KBytes/sec    0    136 KBytes
[  5]   8.00-9.00   sec   128 KBytes   128 KBytes/sec    0    136 KBytes
[  5]   9.00-10.00  sec  0.00 Bytes  0.00 KBytes/sec    0    136 KBytes
[  5]  10.00-11.00  sec   128 KBytes   128 KBytes/sec    0    115 KBytes
[  5]  11.00-12.00  sec  0.00 Bytes  0.00 KBytes/sec    0    115 KBytes
[  5]  12.00-13.00  sec   128 KBytes   128 KBytes/sec    0    143 KBytes
[  5]  13.00-14.00  sec  0.00 Bytes  0.00 KBytes/sec    0    143 KBytes
[  5]  14.00-15.00  sec   128 KBytes   128 KBytes/sec    0    113 KBytes
[  5]  15.00-16.00  sec  0.00 Bytes  0.00 KBytes/sec    0    113 KBytes
[  5]  16.00-17.00  sec   128 KBytes   128 KBytes/sec    0    143 KBytes
[  5]  17.00-18.00  sec  0.00 Bytes  0.00 KBytes/sec    0    143 KBytes
[  5]  18.00-19.00  sec   128 KBytes   128 KBytes/sec    0    136 KBytes
[  5]  19.00-20.00  sec  0.00 Bytes  0.00 KBytes/sec    0    136 KBytes
```

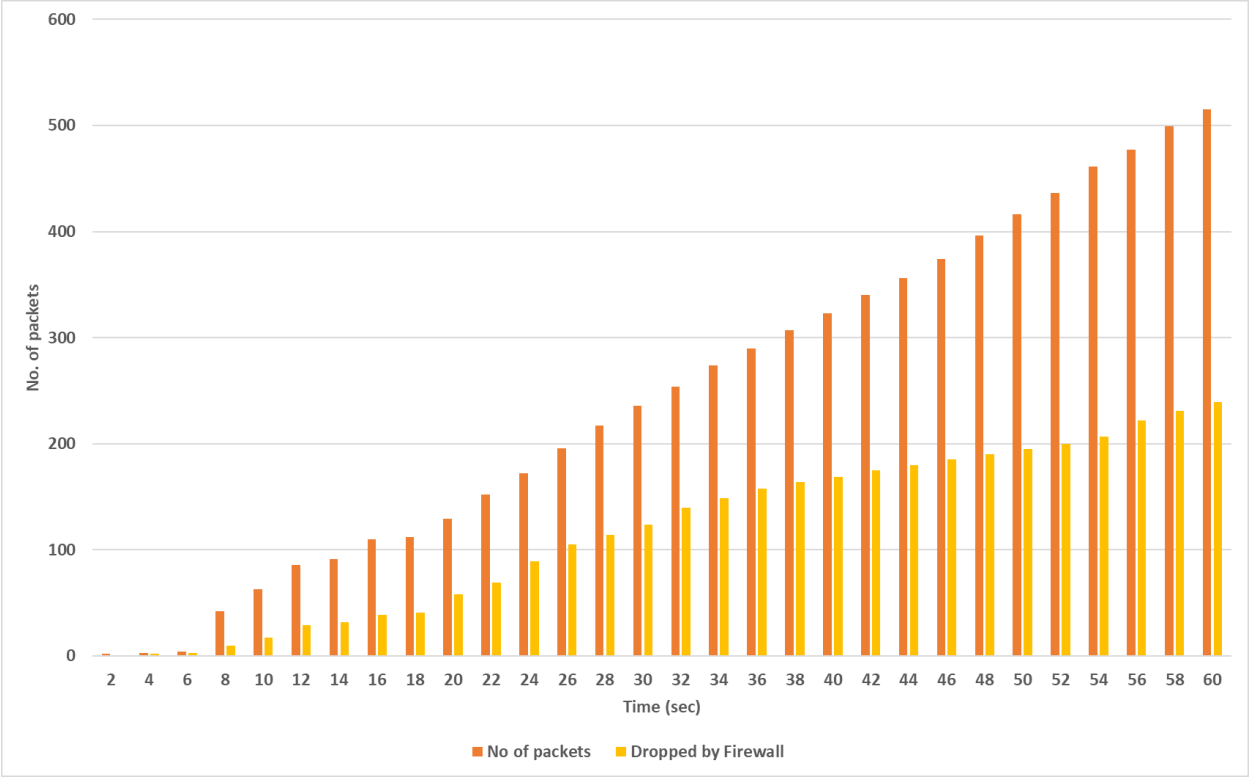
Graph for packets per seconds handled: [Packets sent v/s Packets processed (drop/allow) by firewall]

1. Queue size: 5

Action Pass all packets:

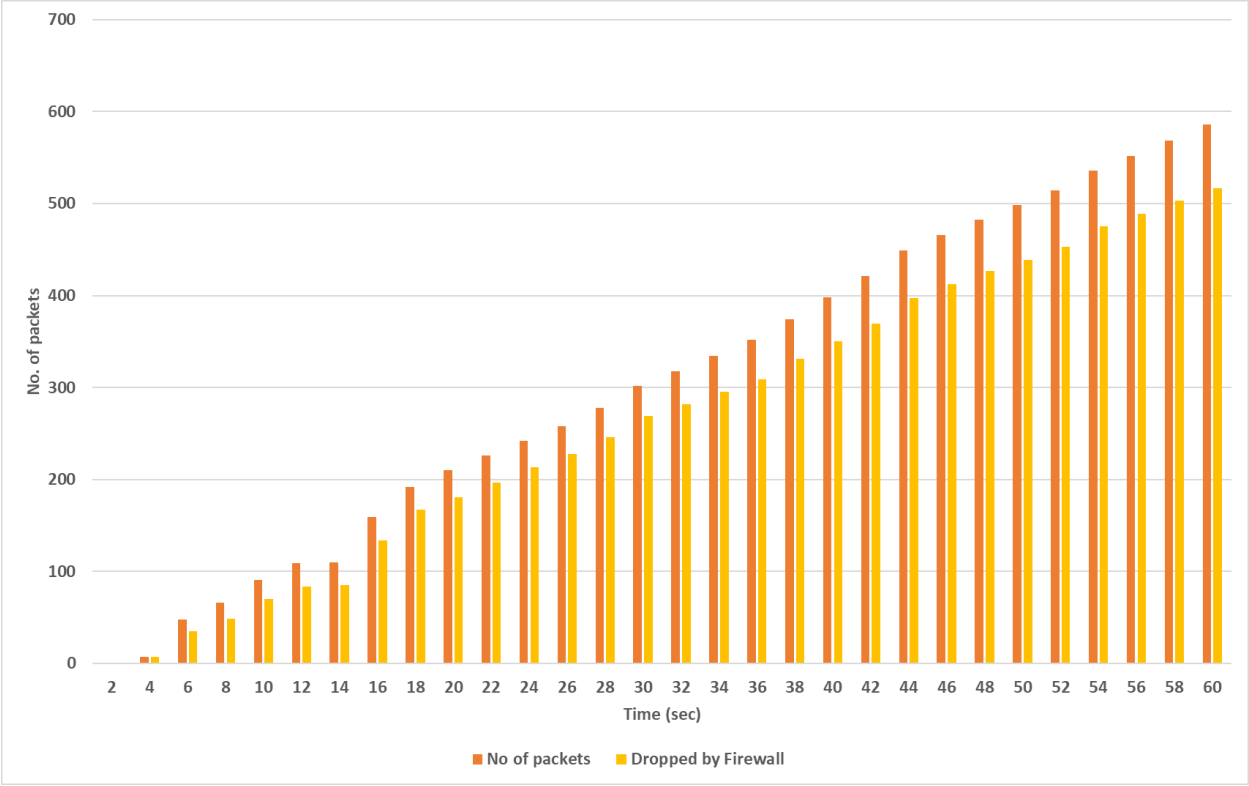


Action Drop all packets:

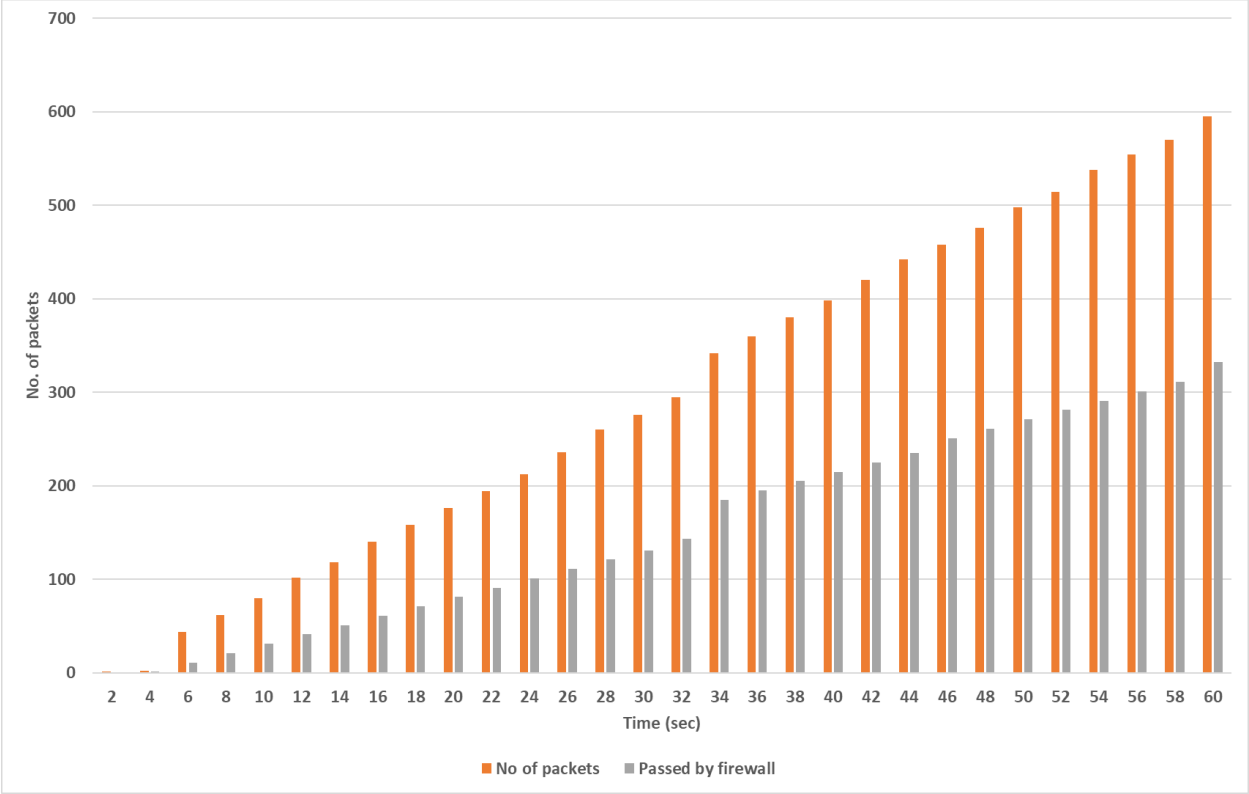


2. Queue size: 10

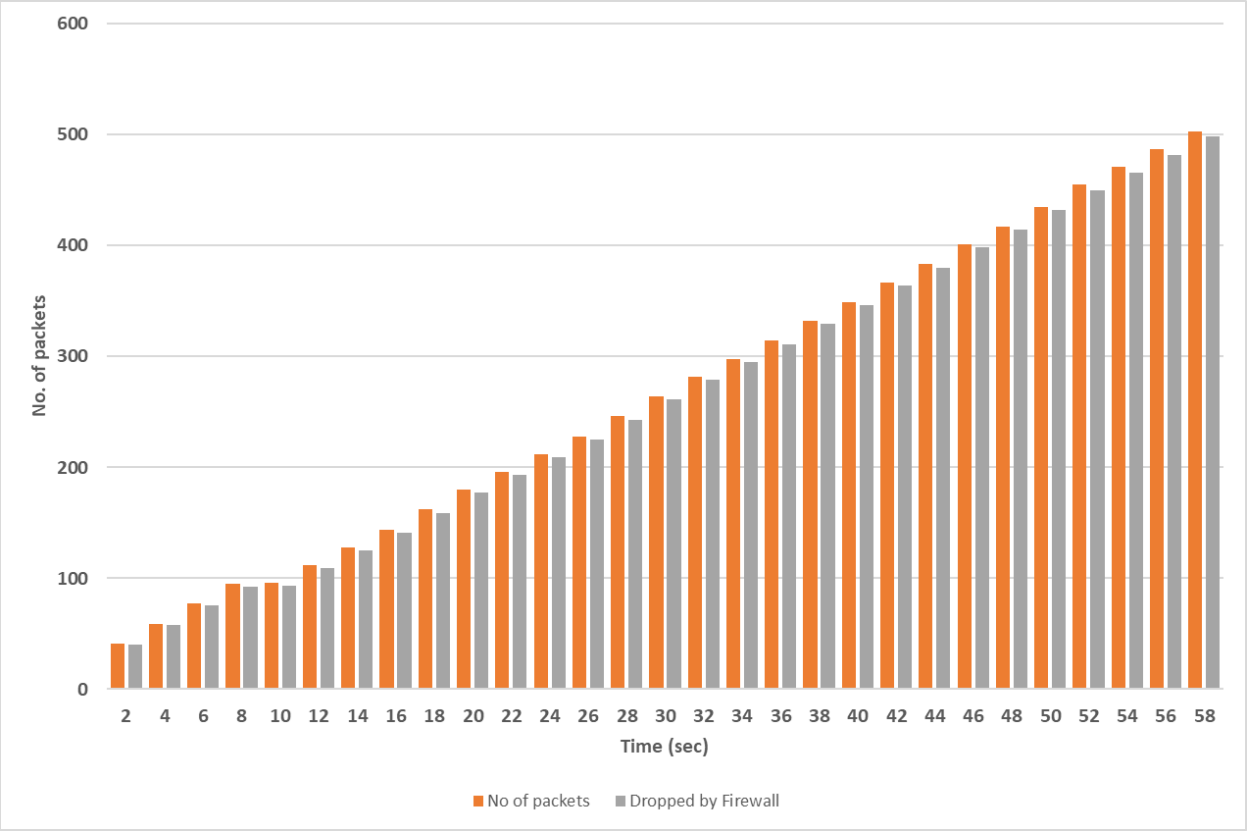
Action Drop all packets:



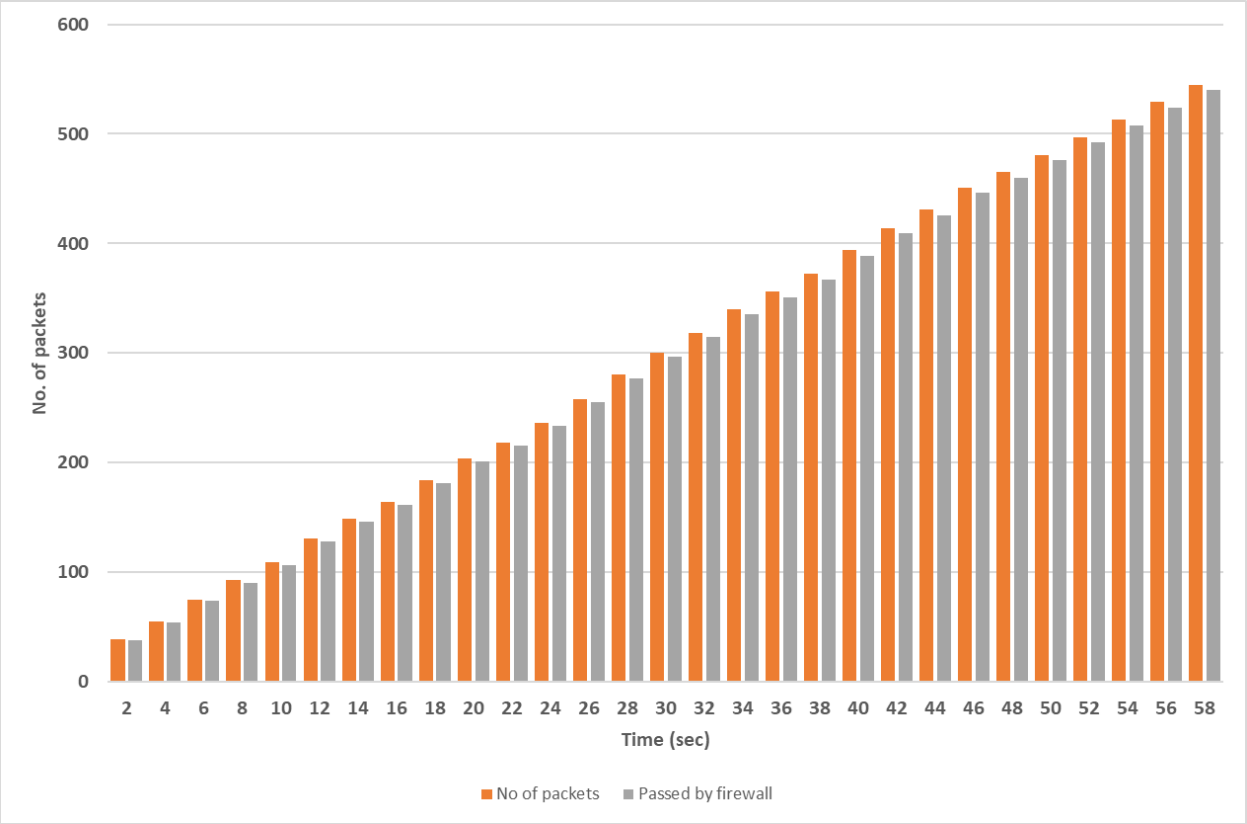
Action Pass all packets:



3. Queue size: 10000
   Action Drop all packets:



Action Pass all packets:



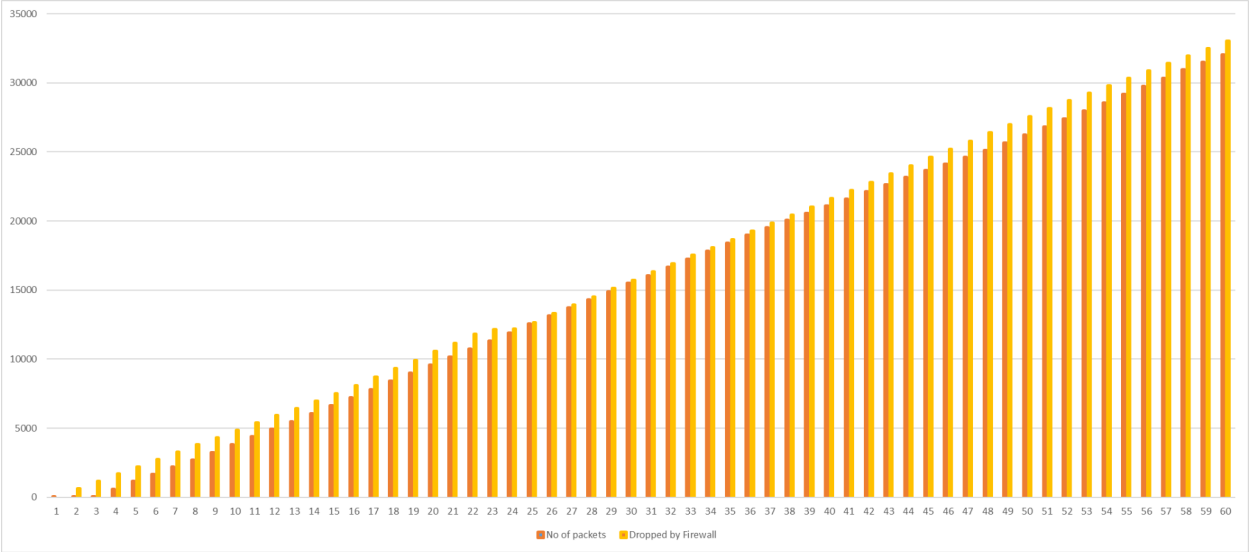
Test 2: Rule matching - Drop all matched rules

Graph for packets per seconds handled by our firewall [Packets sent v/s Packets processed (drop/allow) by firewall]

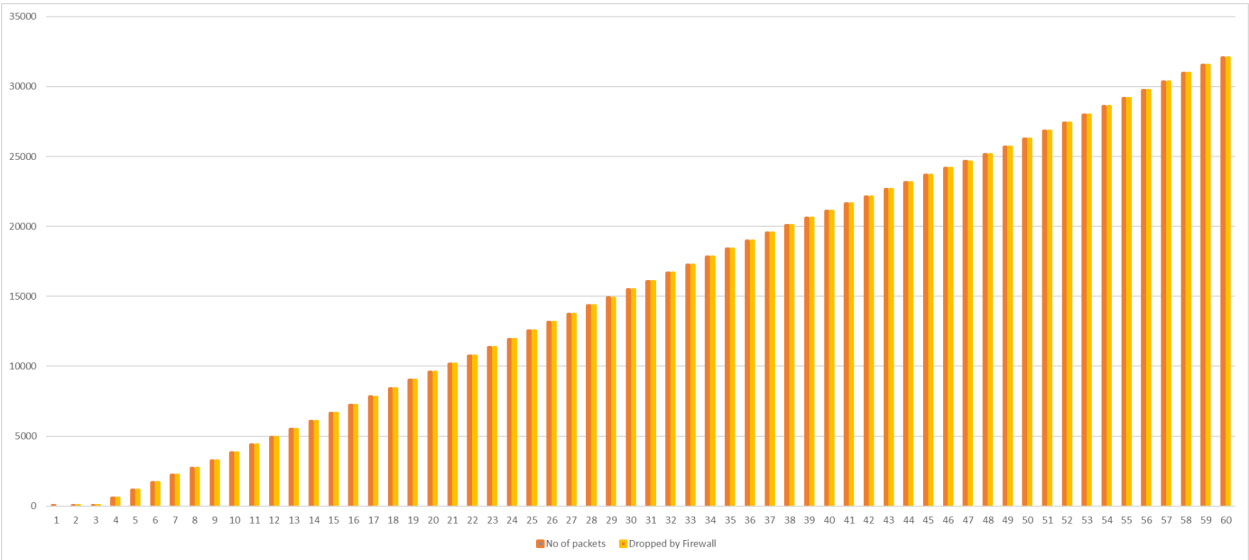
Time: 120 secs

Bandwidth: 50Mbits/sec

1. 10 rules:[9 unmatched IP with 1 matched IP at last]



2. 100 rules:[99 unmatched IP with 1 matched IP at last]

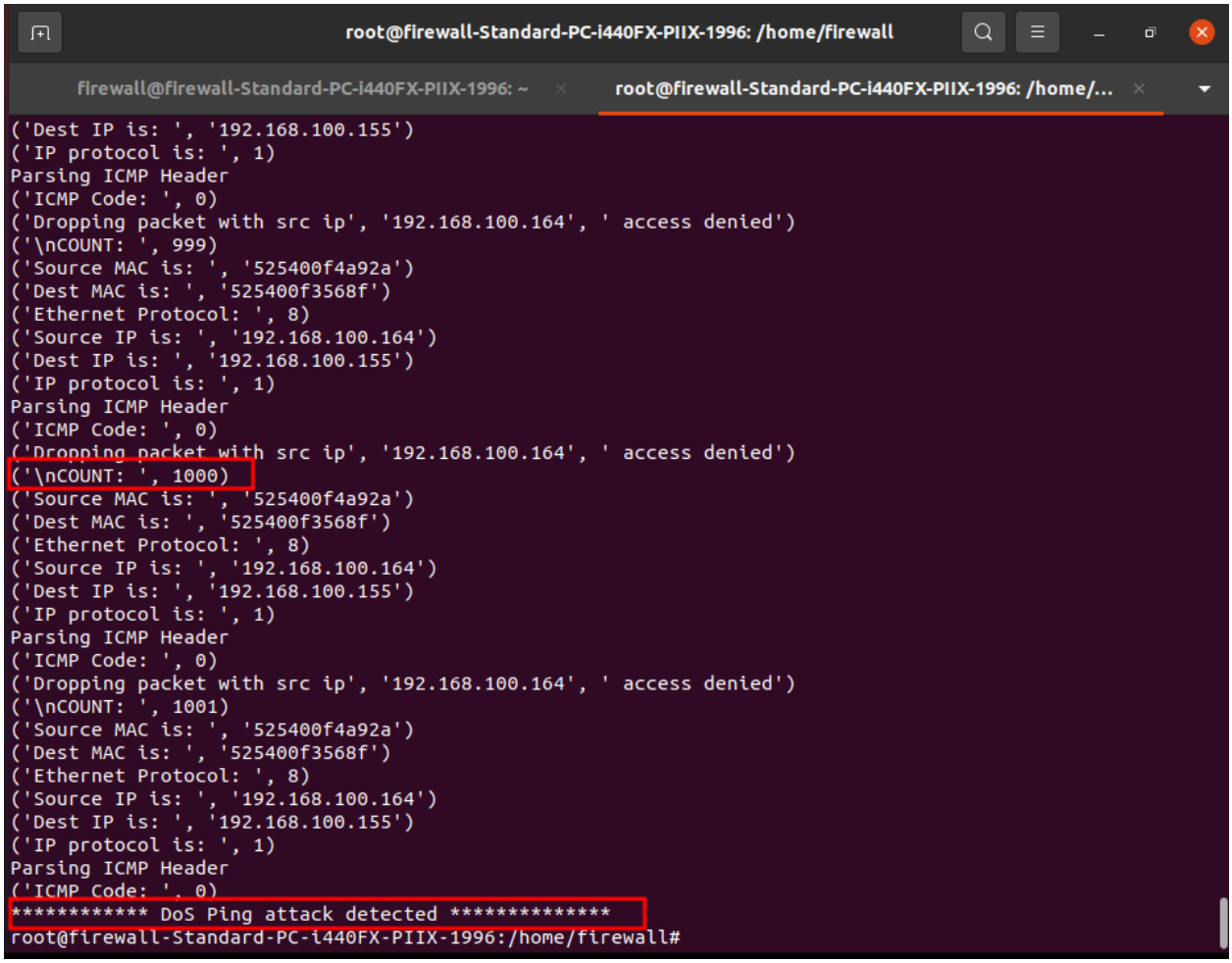


Task 4(b): DoS attack detection

- **Attack considered:** Ping Flood attack
- **Tool used for simulating attack:** hping3
- In order to emulate Ping Flood Attack, we have generated ping packets using hping3 on host 1.
- In order to detect a ping flood attack, we will count the number of times we get ICMP packets, by counting the number of times we got “1” in the ipv4_protocol field. (1 denotes ICMP packet).
- We have kept the packet threshold as 1000 (can be changed accordingly). If the counter is higher than this threshold value, we can consider as a DoS Ping flood attack

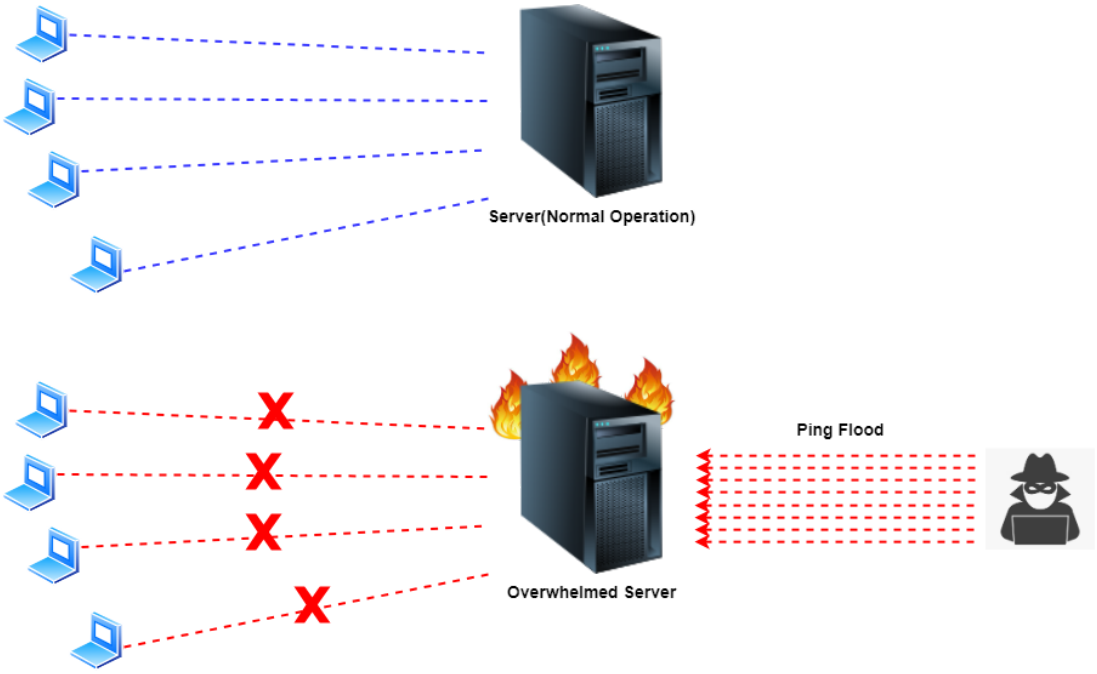
On host1:

```
host1@host1-Standard-PC-i440FX-PIIX-1996:~$ sudo hping3 --icmp --flood 192.168.100.164
HPING 192.168.100.164 (ens3 192.168.100.164): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
```



Ping Flood attack:

It is a type of DoS/DDoS attack which can destabilize or overwhelm the server and make the server unavailable for legitimate users. Ping is generally an ICMP echo request to which the server responds with a response. As we know, each ICMP echo request will get an equal number of packets in reply. The attacker has to know about the IP address of the target. The attacker sends many ping requests (ICMP packets) to the target, which is enough to overwhelm the target and affect the legitimate users. During the attack, the target is flooded with the ping requests and the resources of the target get occupied in responding to the ping request. Below figure illustrates the Ping Flood attack:



References:

- https://python-can.readthedocs.io/en/1.5.2/_modules/can/interfaces/socketcan_ctypes.html
- https://www.opensourceforu.com/2015/03/a-guide-to-using-raw-sockets/
- https://stackoverflow.com/questions/6067405/python-sockets-enabling-promiscuous-mode-in-linux