

Project 1: Firewall Setup using iptables

Submitted by: Shantanu Bhusari (1211213728)

1. Summary

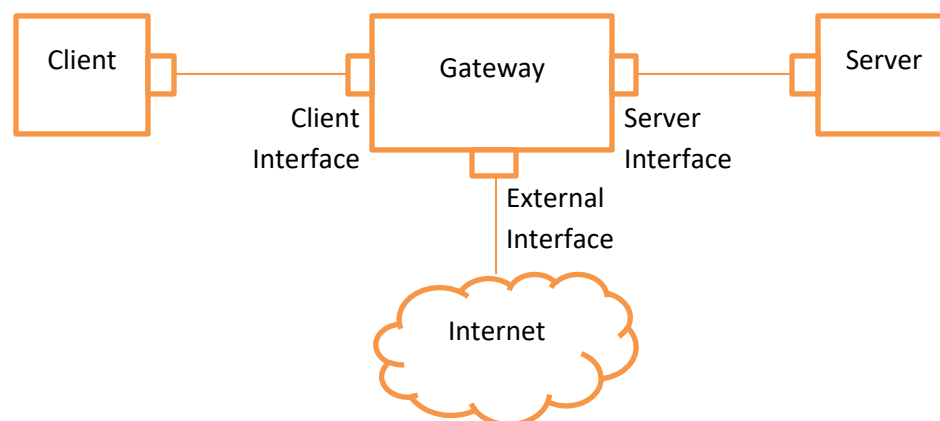
Iptables is a package provided by Linux distribution which helps us to create a stateful firewall by filtering and allowing packets using different rules. In this project, I will demonstrate the usage of firewall which will satisfy certain rules of packet filtering and connection establishment. Given network configuration have three Linux hosts (client, server, and gateway). A gateway is a machine which can allow communication between client and server, as well as to the external network. The rules that are implemented using iptables are as following:

- 1) Allow all types of traffic initiated from server to the Internet.
- 2) Only allow the client to access the web page (HTTP) on the server.
- 3) Only allow the server to ping the client's IP address and gateways' IP addresses.
- 4) The default firewall policy is DROP, which means disable all other traffic except the allowed ones required in 1, 2, and 3.

A script will enable all these rules in gateway host.

2. List of software packages used

- Linux Ubuntu 14.04
- Apache2 (Apache server package on server host)
- Iptables



Setup Diagram with terminologies used in the description

3. Description

In the given setup, the gateway host acts as a gateway which allows communication between server and client, also between internal network and internet. Initially, the routes on server and client are incorrect, so correct gateway addresses were provided in respective routing tables so that server and client can communicate with each other. Further, no NAT setup is done on the gateway host blocking client and server to communicate with the external network.

```
Terminal
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 172.16.0.5 0.0.0.0 UG 0 0 0 eth0
169.254.169.254 172.16.0.3 255.255.255.255 UGH 0 0 0 eth0
172.16.0.0 0.0.0.0 255.240.0.0 U 1 0 0 eth0
ubuntu@ubuntu:~$ ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data:
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=1.27 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.650 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.617 ms
^C
--- 10.0.0.4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.617/0.846/1.273/0.303 ms
ubuntu@ubuntu:~$ ping 192.168.0.5
PING 192.168.0.5 (192.168.0.5) 56(84) bytes of data:
64 bytes from 192.168.0.5: icmp_seq=1 ttl=64 time=0.664 ms
64 bytes from 192.168.0.5: icmp_seq=2 ttl=64 time=0.610 ms
^C
--- 192.168.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1000ms
rtt min/avg/max/mdev = 0.610/0.637/0.664/0.027 ms
ubuntu@ubuntu:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2014ms
ubuntu@ubuntu:~$
```

Client status after fixing routing table

```
Terminal File Edit View Search Terminal Help
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ route -n
Kernel IP routing table
Destination Gateway Genmask Flags Metric Ref Use Iface
0.0.0.0 10.0.0.4 0.0.0.0 UG 0 0 0 eth0
10.0.0.0 0.0.0.0 255.255.255.0 U 1 0 0 eth0
169.254.169.254 10.0.0.3 255.255.255.255 UGH 0 0 0 eth0
ubuntu@ubuntu:~$ ping 172.16.0.4
PING 172.16.0.4 (172.16.0.4) 56(84) bytes of data:
64 bytes from 172.16.0.4: icmp_seq=1 ttl=63 time=2.74 ms
64 bytes from 172.16.0.4: icmp_seq=2 ttl=63 time=1.80 ms
^C
--- 172.16.0.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.804/2.272/2.741/0.470 ms
ubuntu@ubuntu:~$ ping 192.168.0.5
PING 192.168.0.5 (192.168.0.5) 56(84) bytes of data:
64 bytes from 192.168.0.5: icmp_seq=1 ttl=64 time=1.28 ms
64 bytes from 192.168.0.5: icmp_seq=2 ttl=64 time=0.633 ms
^C
--- 192.168.0.5 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 0.633/0.960/1.288/0.328 ms
ubuntu@ubuntu:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data:
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3024ms
ubuntu@ubuntu:~$
```

Server status after fixing routing table

Next, I created a script which enables iptables rules to fulfill project requirements. Initially, in the script, the variables are declared which stores the IP address, network address, broadcast address and interface name on each host. All required modules are imported to setup iptables rules. List of those modules is given in the script provided with this report.

Now, the firewall is created in following steps:

1. Flush all the existing rules. This can be done using following commands:

```
$ iptables -F  
$ iptables -t nat -F (to flush nat setup, if any)
```
2. Before we set any new rules, IPv4 forwarding should be enabled on the gateway to allow packet forwarding:

```
$ echo "1" > /proc/sys/net/ipv4/ip_forward
```
3. The default policy is first set to DROP, to disable all traffic except required:

```
$ iptables -P INPUT DROP  
$ iptables -P OUTPUT DROP  
$ iptables -P FORWARD DROP
```
4. The NAT is enabled on the interface of the gateway which is connected to the external network (eth0), to translate packets between internal network and external network:

```
$ iptables -t nat -A POSTROUTING -o <interface name> -j MASQUERADE
```

Masquerade translates the source IP address of the packet generated from internal network to the IP address of firewall's outgoing interface.
5. In this step, all the traffic generated from the server to the internet is allowed to forward. Here, as the server makes the request, the reply that request should also be accepted to allow communication with the internet:

```
$ iptables -A FORWARD -p tcp -i <server interface> -o <gateway external interface> -j ACCEPT  
$ iptables -A FORWARD -p udp -i <server interface> -o <gateway external interface> -j ACCEPT  
$ iptables -A FORWARD -p tcp -i <gateway external interface> -o <server interface> -j ACCEPT  
$ iptables -A FORWARD -p udp -i <gateway external interface> -o <server interface> -j ACCEPT
```

6. In next step, only client is allowed to access the webpage on the server. The webpage has a simple 'Hello World' message and being hosted on the apache server on the server host. This policy can be implemented using following rules:

```
$ iptables -A FORWARD -p tcp --dport 80 -s <client IP address> -d <server IP address> -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
```

```
$ iptables -A FORWARD -p tcp --sport 80 -s <server IP address> -d <client IP address> -m state --state NEW,RELATED,ESTABLISHED -j ACCEPT
```

Port 80 indicates that the connection is over HTTP. These rules allow to create a new connection or communicate using established connection.

7. In last step, only server is allowed to ping client's IP address or gateway's IP address. Gateway host is directly connected to the server, so there is no need of forwarding. However, ICMP packets should be forwarded between server and client. To allow only server to ping, ICMP packets of type 8 (request) are forwarded from server to client, and ICMP packets of type 0 and 11 (response) are forwarded from client to server:

```
$ iptables -A FORWARD -p ICMP -i <server interface> -o <client interface> -s <server IP address> -d <client IP address> --icmp-type 8 -j ACCEPT
```

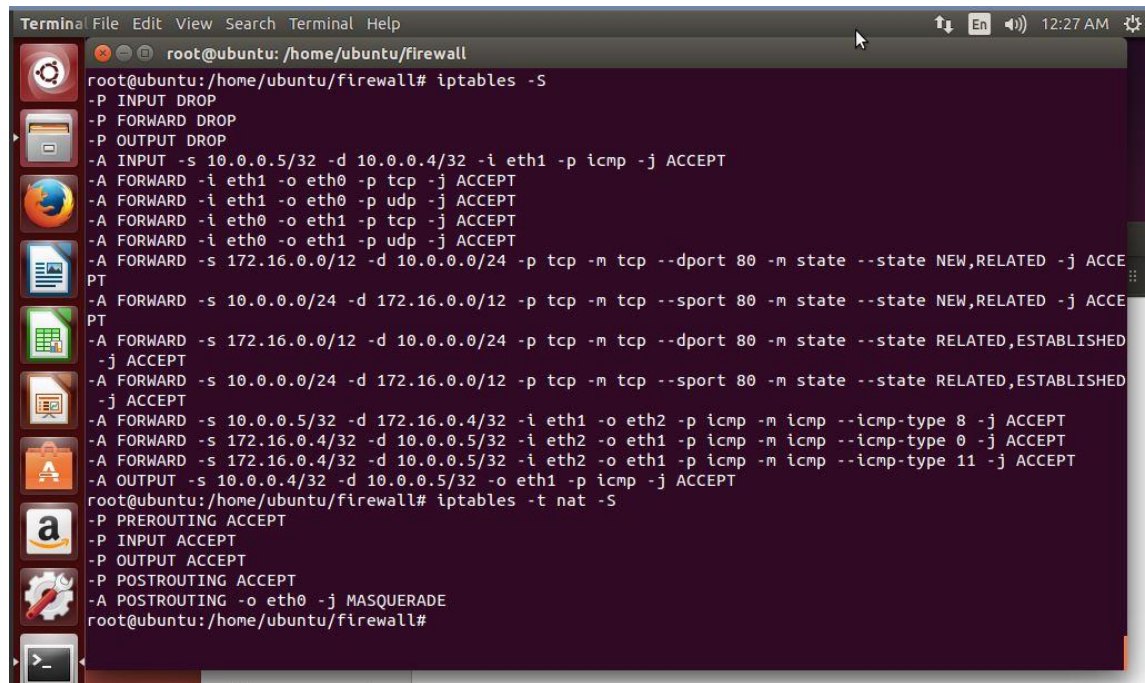
```
$ iptables -A FORWARD -p ICMP -i <client interface> -o <server interface> -s <client IP address> -d <server IP address> --icmp-type 0 -j ACCEPT
```

```
$ iptables -A FORWARD -p ICMP -i <client interface> -o <server interface> -s <client IP address> -d <server IP address> --icmp-type 11 -j ACCEPT
```

```
$ iptables -A INPUT -p ICMP -i <server interface> -s <server IP address> -d <server interface IP on gateway> -j ACCEPT
```

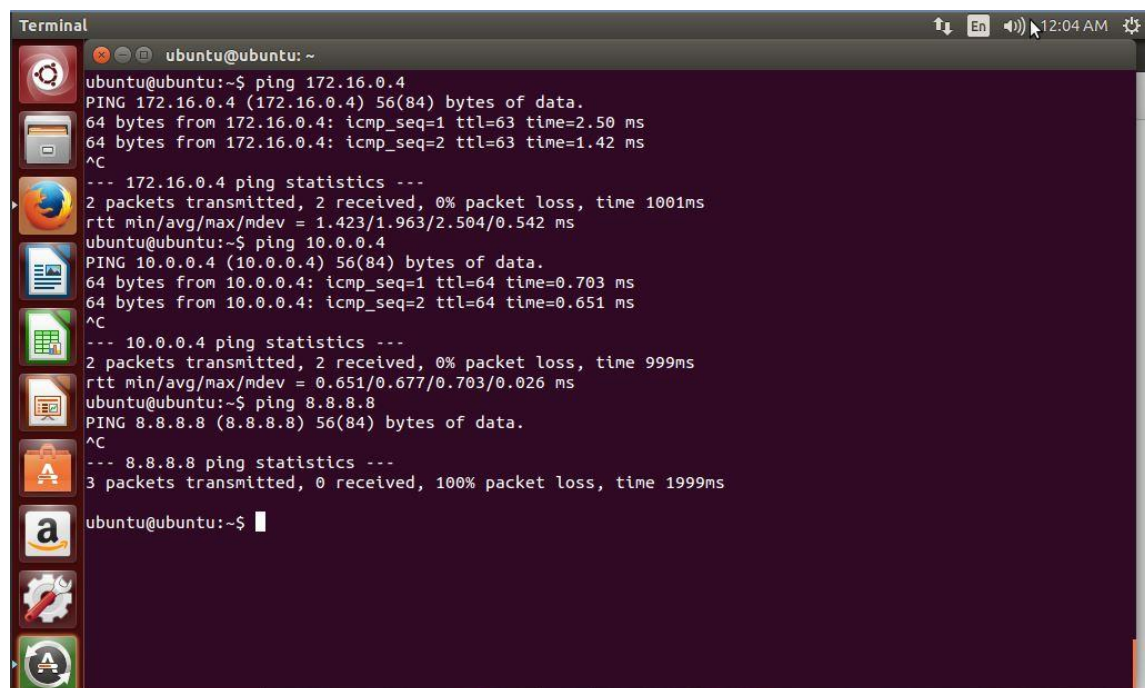
```
$ iptables -A OUTPUT -p ICMP -o <server interface> -s <server interface IP on gateway> -d <server IP address> -j ACCEPT
```

4. Results



```
Terminal File Edit View Search Terminal Help
root@ubuntu: /home/ubuntu/firewall
root@ubuntu:/home/ubuntu/firewall# iptables -S
-P INPUT DROP
-P FORWARD DROP
-P OUTPUT DROP
-A INPUT -s 10.0.0.5/32 -d 10.0.0.4/32 -i eth1 -p icmp -j ACCEPT
-A FORWARD -i eth1 -o eth0 -p tcp -j ACCEPT
-A FORWARD -i eth1 -o eth0 -p udp -j ACCEPT
-A FORWARD -i eth0 -o eth1 -p tcp -j ACCEPT
-A FORWARD -i eth0 -o eth1 -p udp -j ACCEPT
-A FORWARD -s 172.16.0.0/12 -d 10.0.0.0/24 -p tcp -m tcp --dport 80 -m state --state NEW,RELATED -j ACCEPT
-A FORWARD -s 10.0.0.0/24 -d 172.16.0.0/12 -p tcp -m tcp --sport 80 -m state --state NEW,RELATED -j ACCEPT
-A FORWARD -s 172.16.0.0/12 -d 10.0.0.0/24 -p tcp -m tcp --dport 80 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -s 10.0.0.0/24 -d 172.16.0.0/12 -p tcp -m tcp --sport 80 -m state --state RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -s 10.0.0.5/32 -d 172.16.0.4/32 -i eth1 -o eth2 -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A FORWARD -s 172.16.0.4/32 -d 10.0.0.5/32 -i eth2 -o eth1 -p icmp -m icmp --icmp-type 0 -j ACCEPT
-A FORWARD -s 172.16.0.4/32 -d 10.0.0.5/32 -i eth2 -o eth1 -p icmp -m icmp --icmp-type 11 -j ACCEPT
-A OUTPUT -s 10.0.0.4/32 -d 10.0.0.5/32 -o eth1 -p icmp -j ACCEPT
root@ubuntu:/home/ubuntu/firewall# iptables -t nat -S
-P PREROUTING ACCEPT
-P INPUT ACCEPT
-P OUTPUT ACCEPT
-P POSTROUTING ACCEPT
-A POSTROUTING -o eth0 -j MASQUERADE
root@ubuntu:/home/ubuntu/firewall#
```

Policies applied using iptables



```
Terminal
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ ping 172.16.0.4
PING 172.16.0.4 (172.16.0.4) 56(84) bytes of data.
64 bytes from 172.16.0.4: icmp_seq=1 ttl=63 time=2.50 ms
64 bytes from 172.16.0.4: icmp_seq=2 ttl=63 time=1.42 ms
^C
--- 172.16.0.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.423/1.963/2.504/0.542 ms
ubuntu@ubuntu:~$ ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=0.703 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.651 ms
^C
--- 10.0.0.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.651/0.677/0.703/0.026 ms
ubuntu@ubuntu:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 1999ms
ubuntu@ubuntu:~$
```

Only server can ping client and gateway


```
Terminal
ubuntu@ubuntu: ~
ubuntu@ubuntu:~$ ping 10.0.0.4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
^C
--- 10.0.0.4 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3022ms

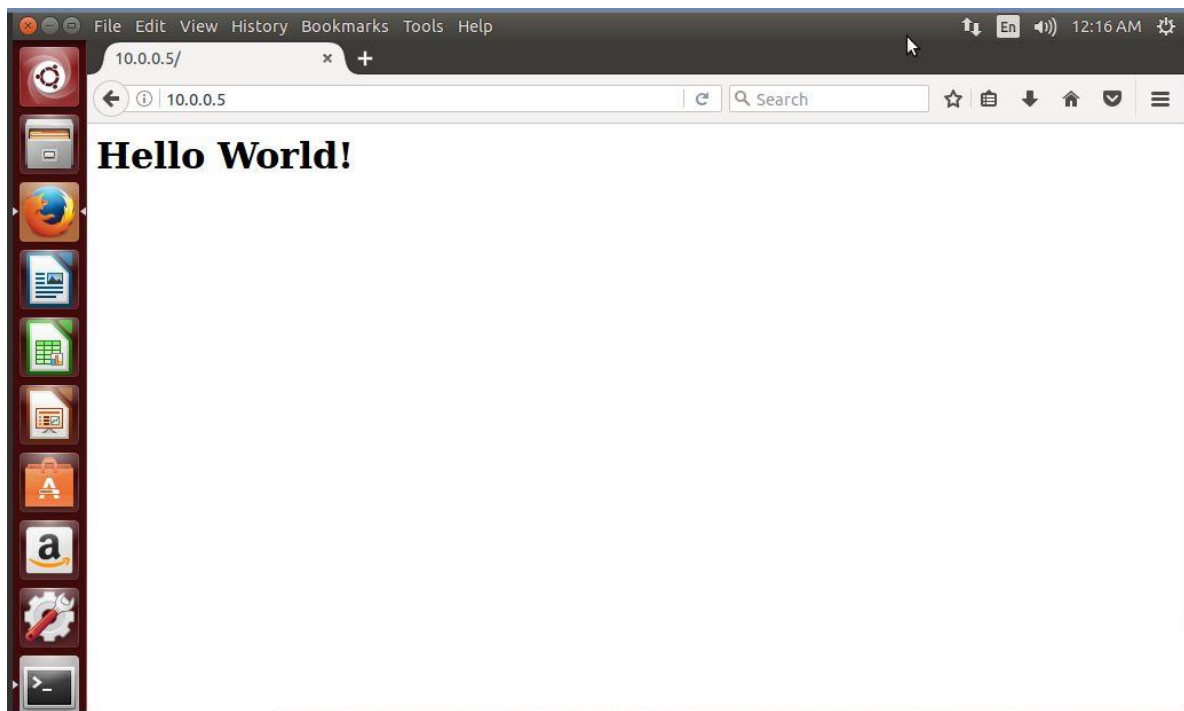
ubuntu@ubuntu:~$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7055ms

ubuntu@ubuntu:~$
```

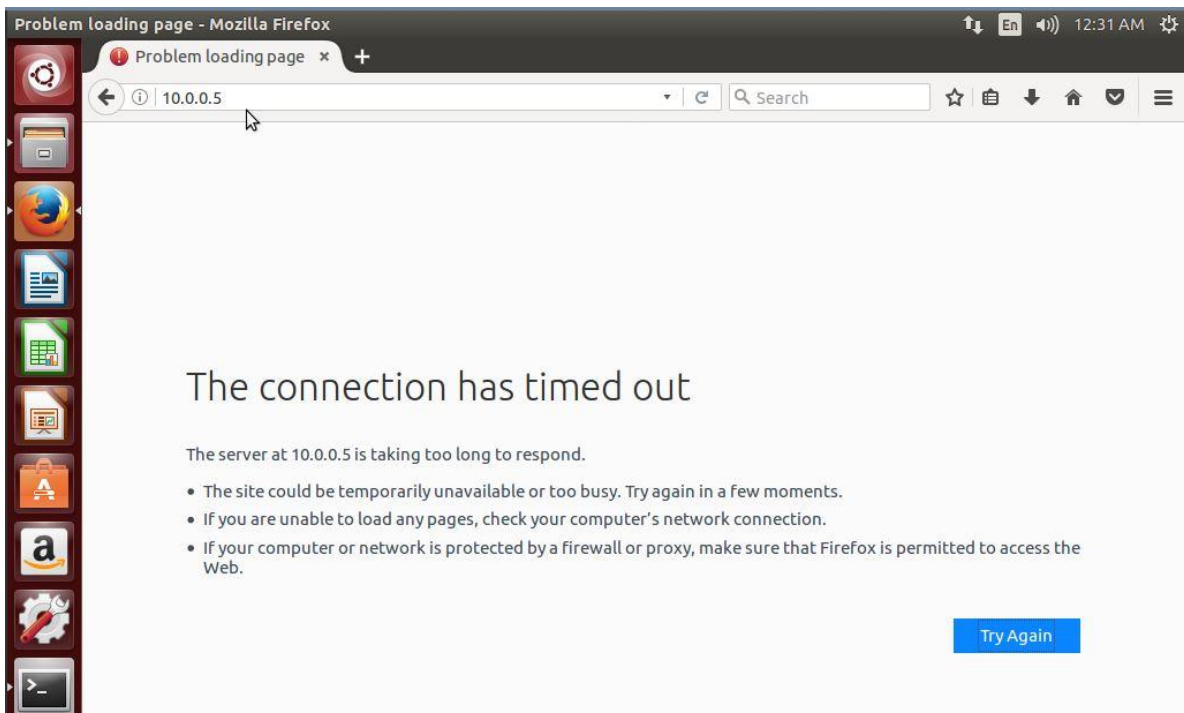
```
Terminal File Edit View Search Terminal Help
root@ubuntu: /home/ubuntu/firewall
root@ubuntu:/home/ubuntu/firewall# ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 0 received, 100% packet loss, time 2015ms

root@ubuntu:/home/ubuntu/firewall#
```

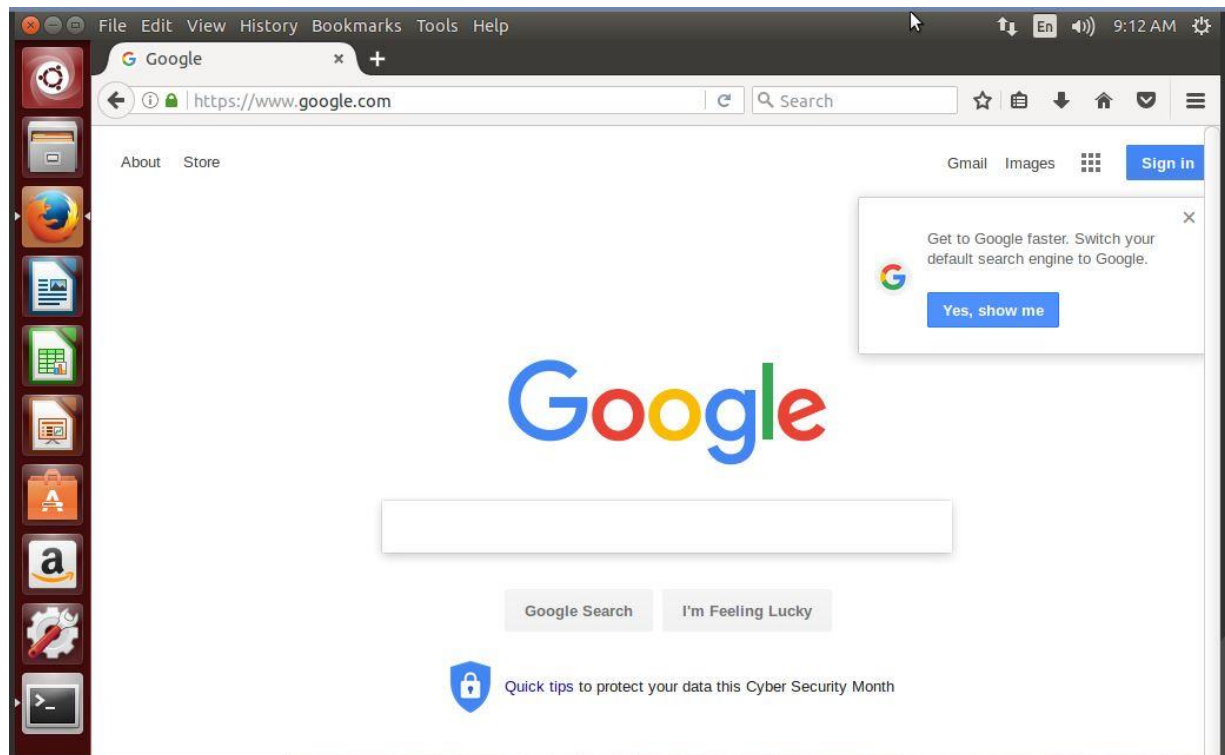
Client and gateway are not allowed to ping anywhere.



Only Client can access 'Hello World' webpage on the server



Gateway trying to access webpage on the server



Connection initiated from the server to the internet is established.

5. Conclusion

Given setup of firewall policies is successfully performed using iptables. It can be observed that the iptables not only filter packets using network and transport layer protocols but also based on connection state, making it a stateful firewall.

6. Appendix

- rc.firewall – script to enable policies using iptables
- Link for the demonstration of project on Youtube: <https://youtu.be/W0NzP6MOPRs>