Lab Assignment 11

Aim: To study and configure Firewalls using IP tables

Lab Outcome Attainment: LO6

Firewall:

A firewall is a system designed to prevent unauthorized access to or from a private network. You can implement a firewall in either hardware or software form, or a combination of both. Generally the firewall has two network interfaces: one for the external side of the network, one for the internal side. Its purpose is to control what traffic is allowed to traverse from one side to the other. As the most basic level, firewalls can block traffic intended for particular IP addresses or server ports.

TCP network traffic moves around a network in packets, which are containers that consist of a packet header—this contains control information such as source and destination addresses, and packet sequence information—and the data (also known as a payload). While the control information in each packet helps to ensure that its associated data gets delivered properly, the elements it contains also provides firewalls a variety of ways to match packets against firewall rules.

Types of Firewalls

Three basic types of network firewalls: packet filtering (stateless), stateful, and application layer.

Packet filtering, or stateless, firewalls work by inspecting individual packets in isolation. As such, they are unaware of connection state and can only allow or deny packets based on individual packet headers.

Stateful firewalls are able to determine the connection state of packets, which makes them much more flexible than stateless firewalls. They work by collecting related packets until the connection state can be determined before any firewall rules are applied to the traffic.

Application firewalls go one step further by analyzing the data being transmitted, which allows network traffic to be matched against firewall rules that are specific to individual services or applications. These are also known as proxy-based firewalls.

Basic of iptables

Iptables is a firewall, installed by default on all official Ubuntu distributions (Ubuntu, Kubuntu, Xubuntu). When you install Ubuntu, iptables is there, but it allows all traffic by default.

The rules in IPTables are written to deal 3 different scenarios:

1.Those packets entering your machine that are destined for your machine. (INPUT)

2.Those packets leaving your machine. (OUTPUT)

3.Those packets entering your machine, but are destined for another machine and will pass through your machine (FORWARD).

In Iptables, these scenarios are referred to as INPUT, OUTPUT, and FORWARD, respectively.

Once the traffic type has been specified, three actions may be taken:

1.ACCEPT allows packets to pass through the firewall.

2.DROP ignores the packet and sends no response to the request.

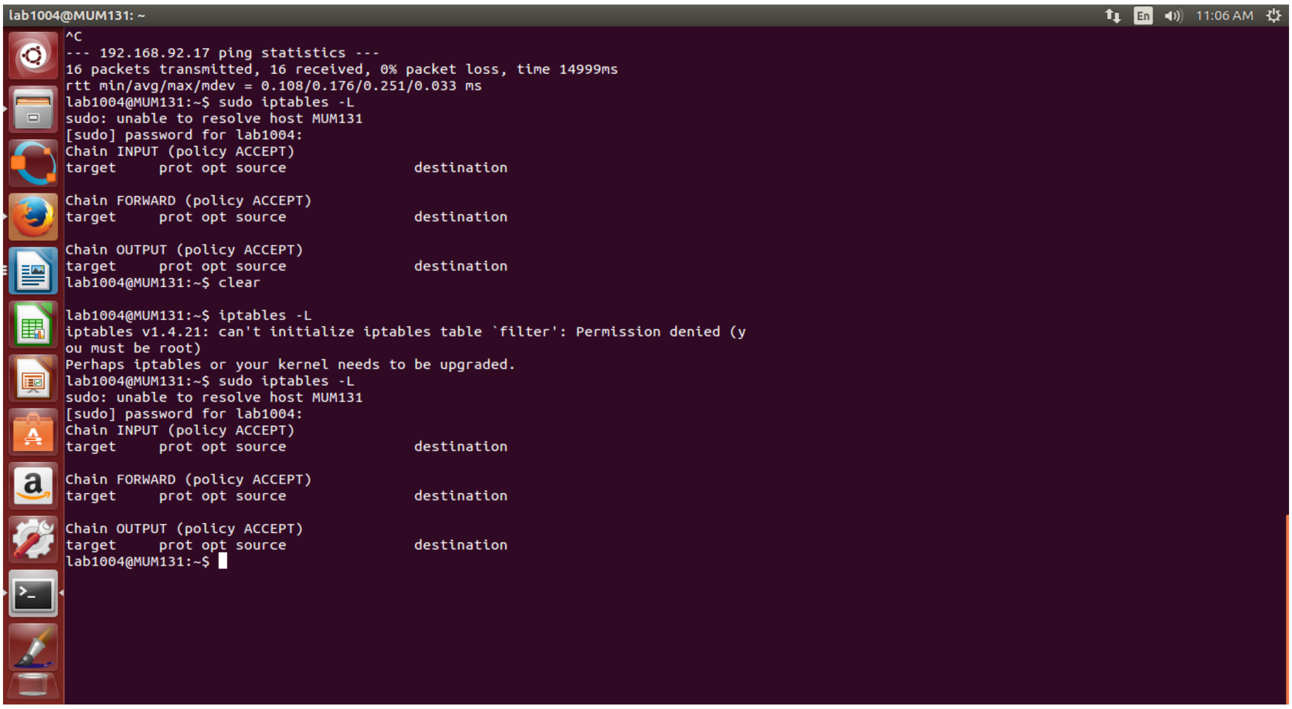
3.REJECT ignores the packet, but responds to the request with a packet denied message.

Basic commands

Typing

```
sudo iptables -L
```

```
(-L - List the current filter rules. )
```



```
^C
--- 192.168.92.17 ping statistics ---
16 packets transmitted, 16 received, 0% packet loss, time 14999ms
rtt min/avg/max/mdev = 0.108/0.176/0.251/0.033 ms
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
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
lab1004@MUM131:~$ clear

lab1004@MUM131:~$ iptables -L
iptables v1.4.21: can't initialize iptables table `filter': Permission denied (y
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
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
lab1004@MUM131:~$ 
```

As you can see, we have our three default chains (INPUT,OUTPUT, and FORWARD). We also can see each chain's default policy (each chain has ACCEPT as its default policy). We also see some column headers, but we don't see any actual rules. This is because Ubuntu doesn't ship with a default rule set.

Basic Iptables Options

- `-A` - Append this rule to a rule chain. Valid chains for what we're doing are INPUT, FORWARD and OUTPUT, but we mostly deal with INPUT in this tutorial, which affects only incoming traffic.
- `-p` - The connection protocol used.
- `--dport` - The destination port(s) required for this rule. A single port may be given, or a range may be given as `start:end`, which will match all ports from `start` to `end`, inclusive.
- `-j` - Jump to the specified target. By default, iptables allows four targets:
    - `ACCEPT` - Accept the packet and stop processing rules in this chain.
    - `REJECT` - Reject the packet and notify the sender that we did so, and stop processing rules in this chain.
    - `DROP` - Silently ignore the packet, and stop processing rules in this chain.
    - `LOG` - Log the packet, and continue processing more rules in this chain. Allows the use of the `--log-prefix` and `--log-level` options.
- `-i` - Only match if the packet is coming in on the specified interface.
- `-I` - Inserts a rule. Takes two options, the chain to insert the rule into, and the rule number it should be.
    - `-I INPUT 5` would insert the rule into the INPUT chain and make it the 5th rule in the list.
- `-v` - Display more information in the output. Useful for if you have rules that look similar without using `-v`.
- `-s --source` - address[/mask] source specification
- `-d --destination` - address[/mask] destination specification
- `-o --out-interface` - output name[+] network interface name ([+] for wildcard)

Allowing Incoming Traffic on Specific Ports

You could start by blocking traffic, but you might be working over SSH, where you would need to allow SSH before blocking everything else.

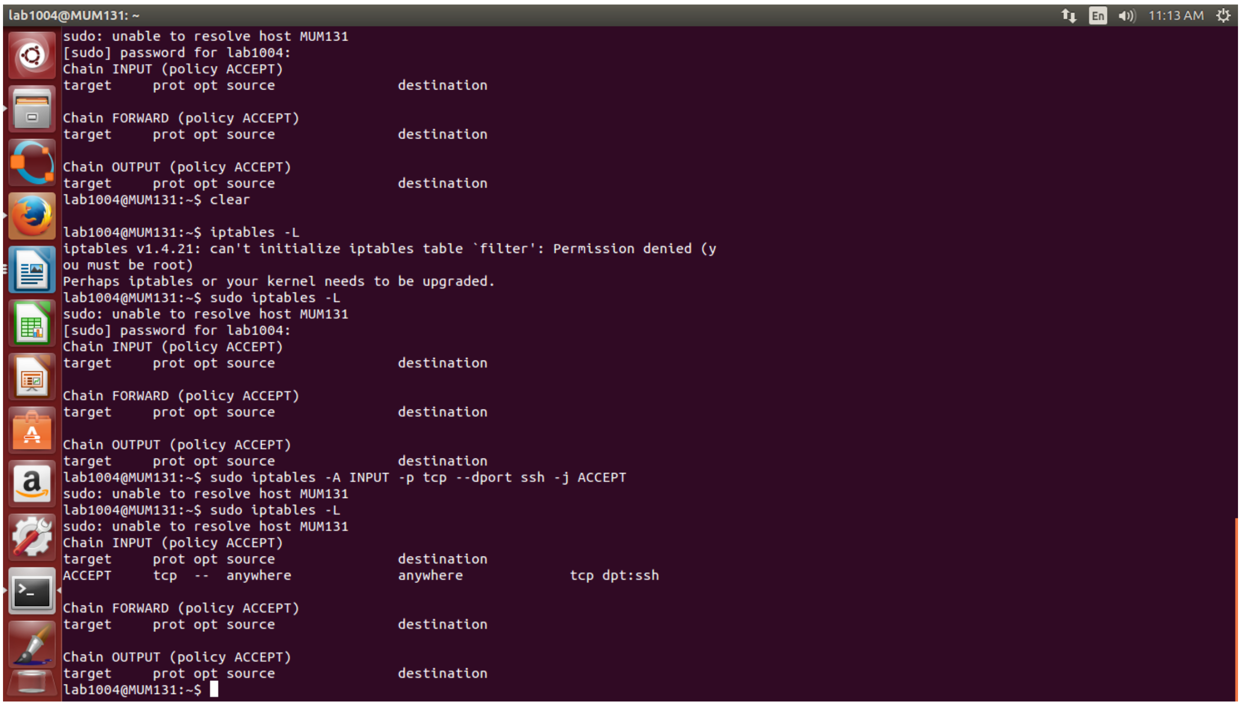
To allow incoming traffic on the default SSH port (22), you could tell iptables to allow all

TCP traffic on that port to come in.

```
sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
```

Referring back to the list above, you can see that this tells iptables:

- append this rule to the input chain (-A INPUT) so we look at incoming traffic
- check to see if it is TCP (-p tcp).
- if so, check to see if the input goes to the SSH port (--dport ssh).
- if so, accept the input (-j ACCEPT).

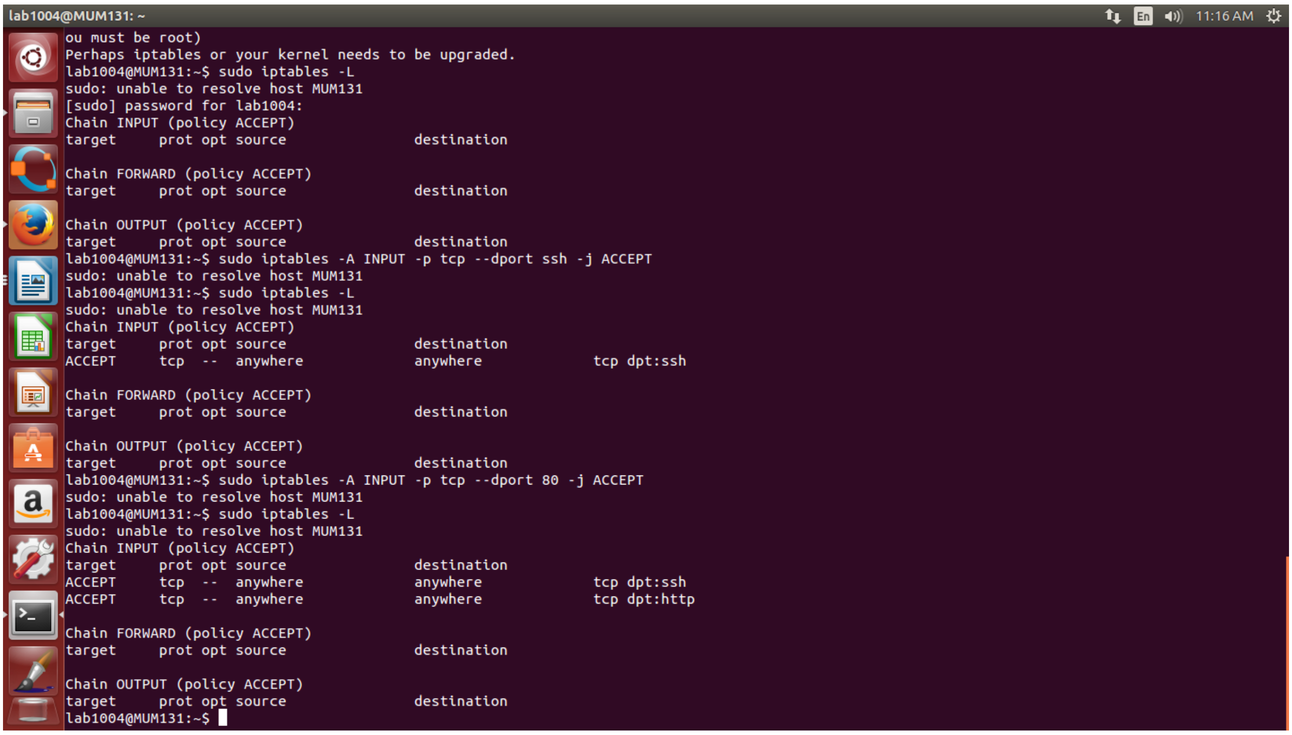


```
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ clear

lab1004@MUM131:~$ iptables -L
iptables v1.4.21: can't initialize iptables table `filter': Permission denied (y
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
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
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```

Now, let's allow all incoming web traffic

```
sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
```

We have specifically allowed tcp traffic to the ssh and web ports, but as we have not blocked anything, all traffic can still come in.



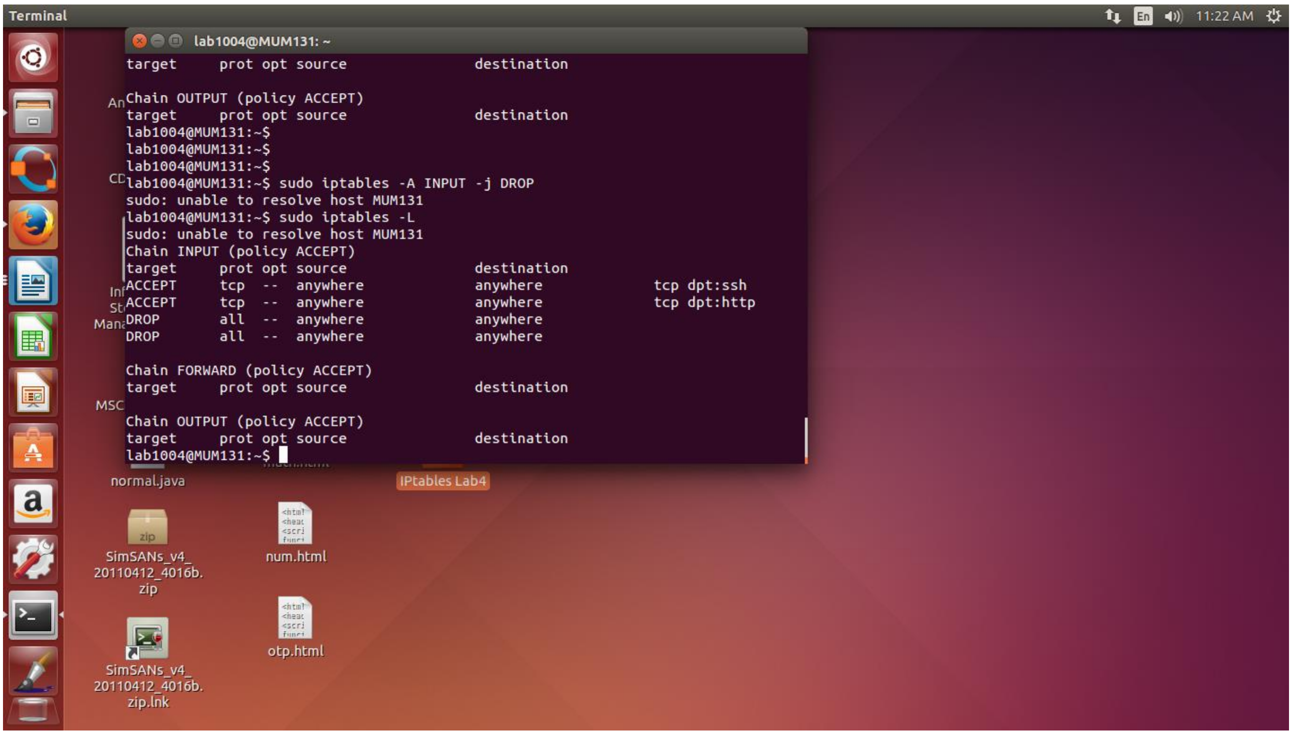
```
ou must be root)
Perhaps iptables or your kernel needs to be upgraded.
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
[sudo] password for lab1004:
Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport ssh -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp --dport 80 -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```

Blocking Traffic

Once a decision is made to accept a packet, no more rules affect it. As our rules allowing ssh and web traffic come first, as long as our rule to block all traffic comes after them, we can still accept the traffic we want. All we need to do is put the rule to block all traffic at the end.

```
sudo iptables -A INPUT -j DROP

sudo iptables -L
```

Because we didn't specify an interface or a protocol, any traffic for any port on any interface is blocked, except for web and ssh.

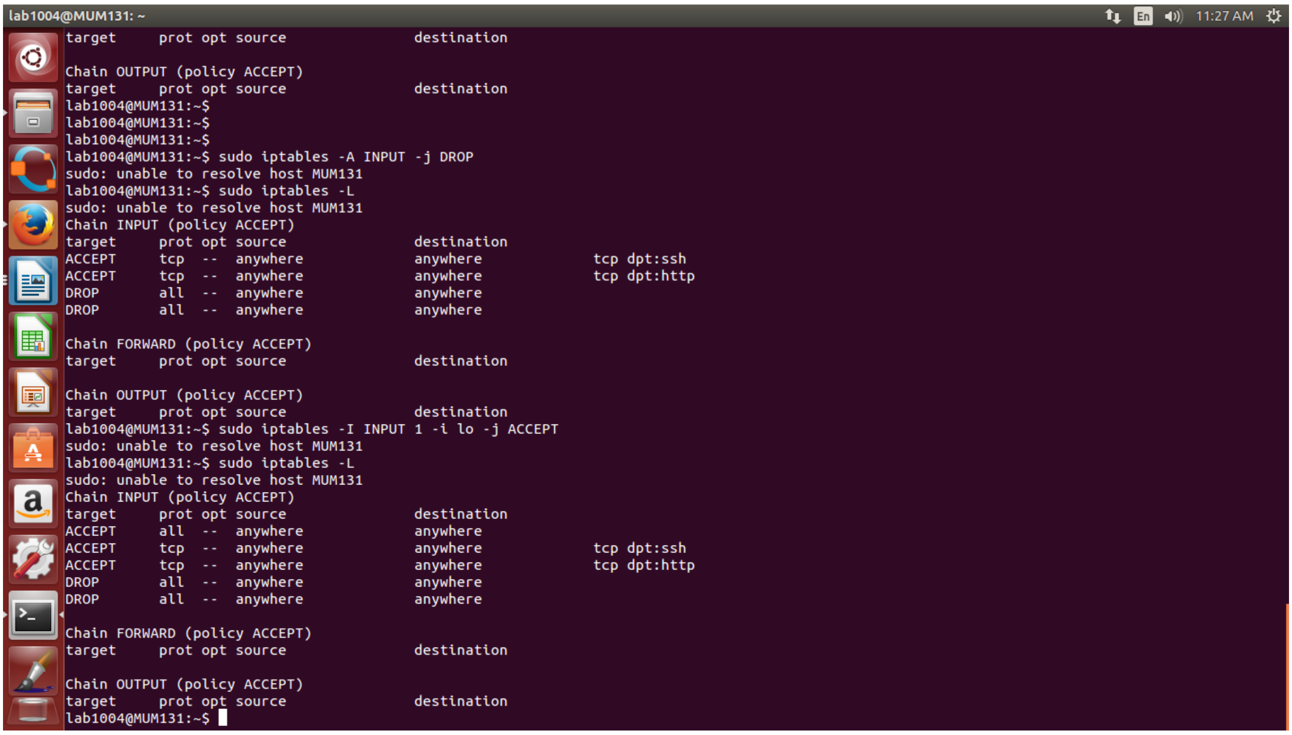


```
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$ sudo iptables -A INPUT -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```

Editing iptables

The only problem with our setup so far is that even the loopback port is blocked. We could have written the drop rule for just eth0 by specifying -i eth0, but we could also add a rule for the loopback. If we append this rule, it will come too late - after all the traffic has been dropped. We need to insert this rule before that. Since this is a lot of traffic, we'll insert it as the first rule so it's processed first.

```
sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo iptables -L
```

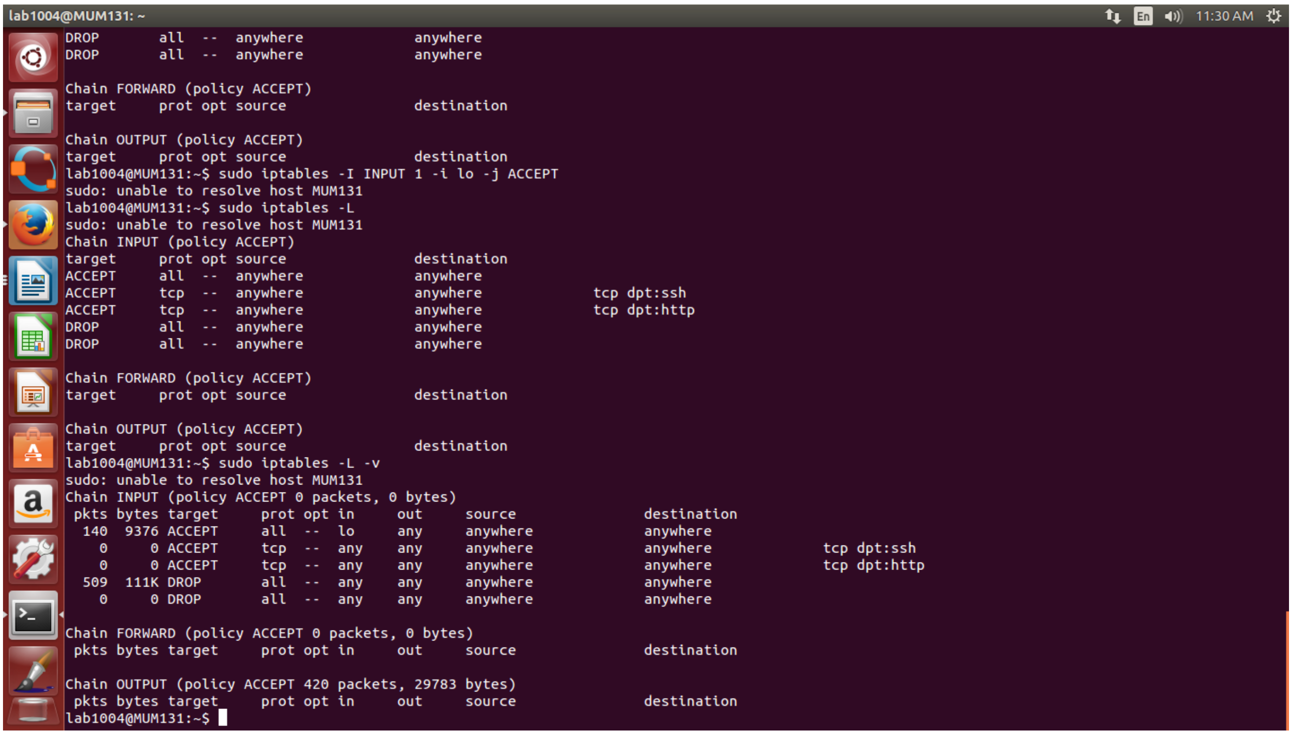


```
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$
lab1004@MUM131:~$ sudo iptables -A INPUT -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```

we will list iptables in greater detail.

```
sudo iptables -L -v
```

You can now see a lot more information. This rule is actually very important, since many programs use the loopback interface to communicate with each other.



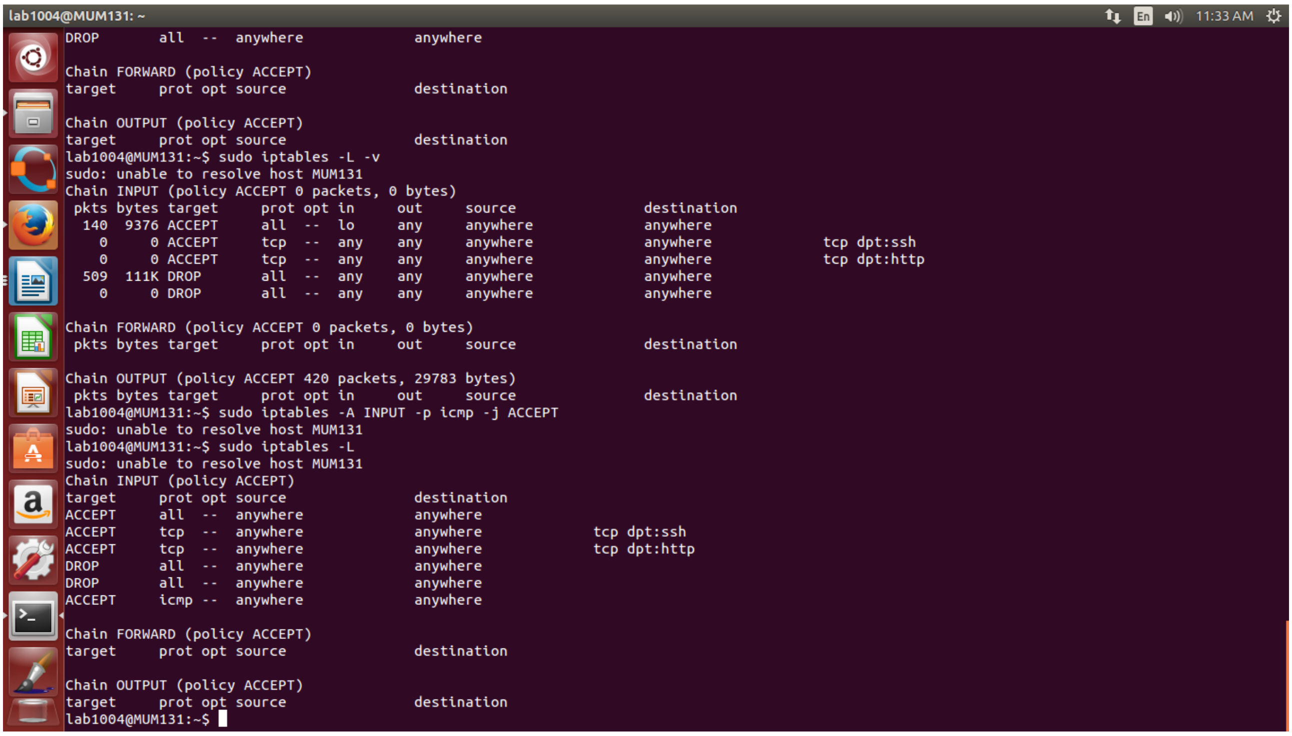
```
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -I INPUT 1 -i lo -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -L -v
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  140  9376 ACCEPT     all  --  lo     any     anywhere             anywhere
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:ssh
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:http
  509  111K DROP       all  --  any    any     anywhere             anywhere
    0     0 DROP       all  --  any    any     anywhere             anywhere

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

Chain OUTPUT (policy ACCEPT 420 packets, 29783 bytes)
 pkts bytes target     prot opt in     out     source               destination
lab1004@MUM131:~$
```

Allow traffic on ICMP port

sudo iptables -A INPUT -p icmp -j ACCEPT

now list the rules again..

sudo iptables -L

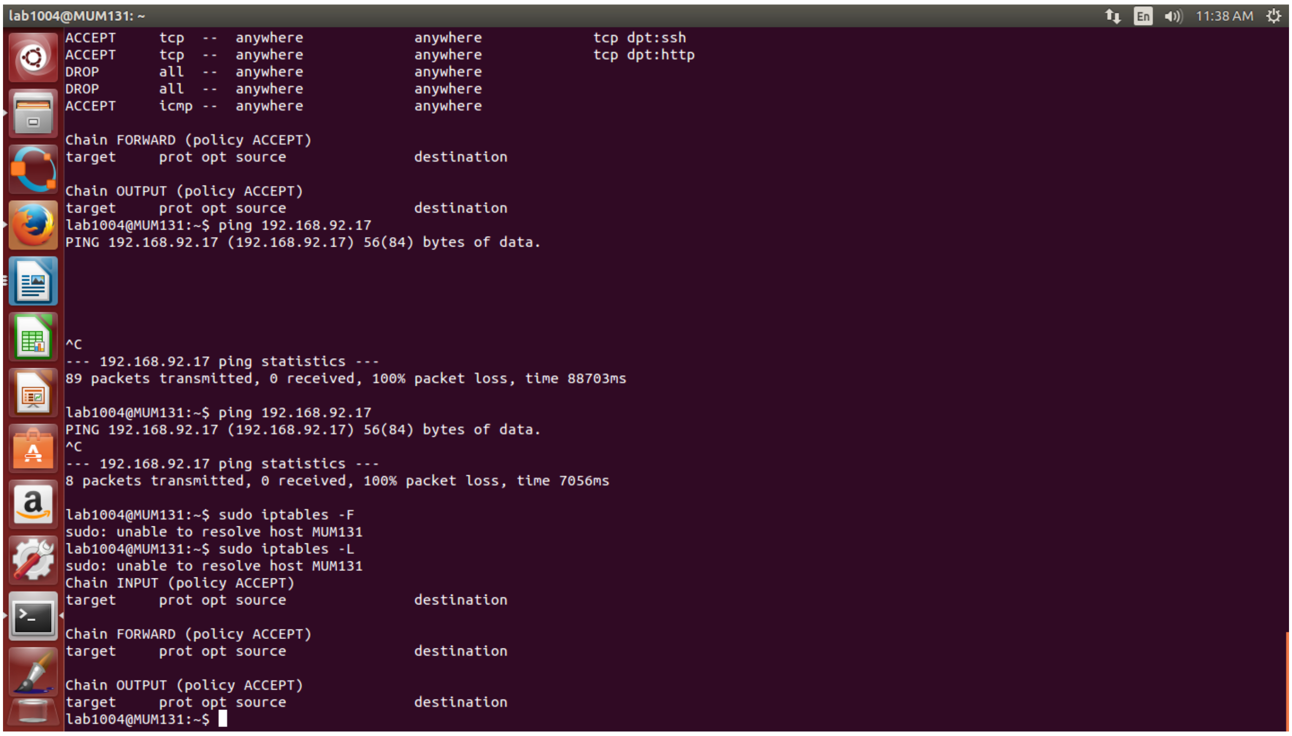


```
DROP       all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -L -v
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target     prot opt in     out     source               destination
  140  9376 ACCEPT     all  --  lo     any     anywhere             anywhere
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:ssh
    0     0 ACCEPT     tcp  --  any    any     anywhere             anywhere             tcp dpt:http
  509  111K DROP       all  --  any    any     anywhere             anywhere
    0     0 DROP       all  --  any    any     anywhere             anywhere

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

Chain OUTPUT (policy ACCEPT 420 packets, 29783 bytes)
 pkts bytes target     prot opt in     out     source               destination
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp -j ACCEPT
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere
ACCEPT     icmp --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```

clearing all rules

sudo iptables -F

sudo iptables -L



```
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere             tcp dpt:http
DROP       all  --  anywhere             anywhere
DROP       all  --  anywhere             anywhere
ACCEPT     icmp --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.





^C
--- 192.168.92.17 ping statistics ---
89 packets transmitted, 0 received, 100% packet loss, time 88703ms

lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
8 packets transmitted, 0 received, 100% packet loss, time 7056ms

lab1004@MUM131:~$ sudo iptables -F
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
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
lab1004@MUM131:~$ 
```

Dropping icmp packets

try to ping ur neighbour machine

ping 192.168.92.17

u can see the response packets received.

Now block incoming icmp packets from the neighbour using command:

sudo iptables -A **INPUT** -p icmp -j DROP

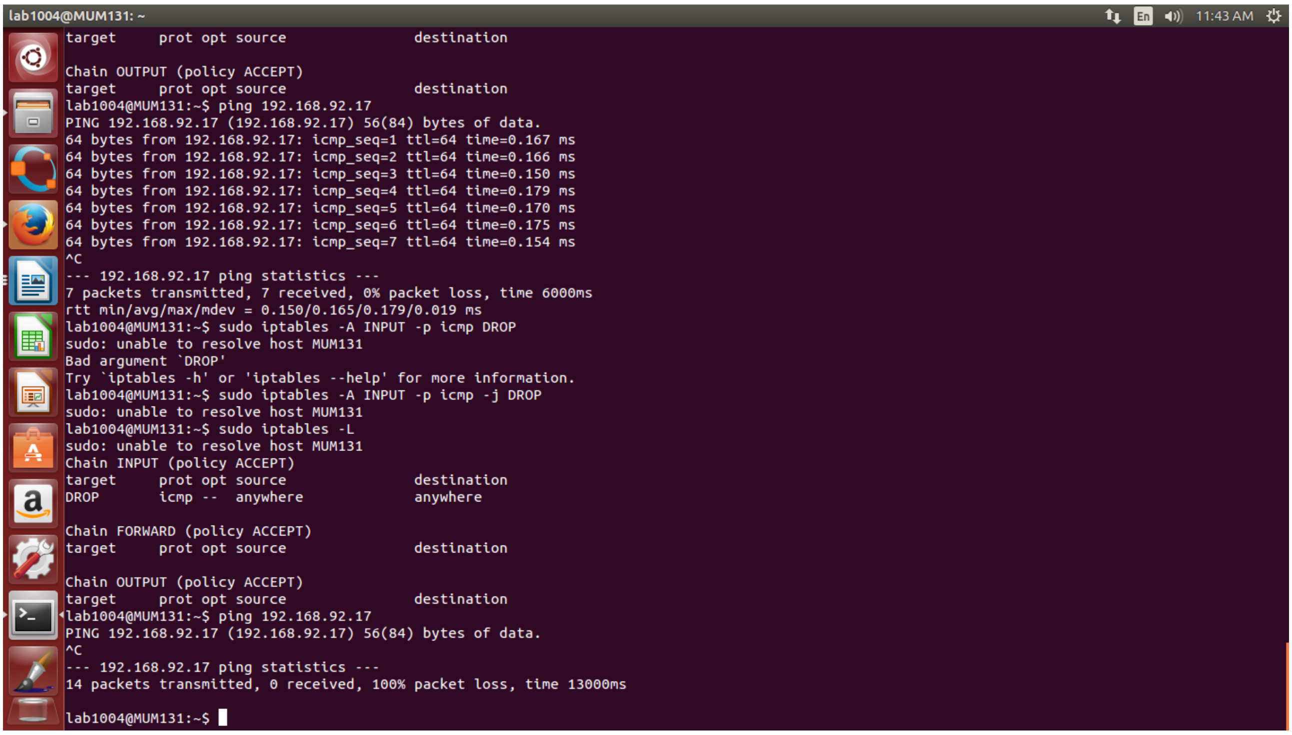
list the rule:

sudo iptables -L

try to ping ur neighbour machine again

ping 192.168.92.17

u can not see receive icmp echo reply packets..



```
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
64 bytes from 192.168.92.17: icmp_seq=1 ttl=64 time=0.167 ms
64 bytes from 192.168.92.17: icmp_seq=2 ttl=64 time=0.166 ms
64 bytes from 192.168.92.17: icmp_seq=3 ttl=64 time=0.150 ms
64 bytes from 192.168.92.17: icmp_seq=4 ttl=64 time=0.179 ms
64 bytes from 192.168.92.17: icmp_seq=5 ttl=64 time=0.170 ms
64 bytes from 192.168.92.17: icmp_seq=6 ttl=64 time=0.175 ms
64 bytes from 192.168.92.17: icmp_seq=7 ttl=64 time=0.154 ms
^C
--- 192.168.92.17 ping statistics ---
7 packets transmitted, 7 received, 0% packet loss, time 6000ms
rtt min/avg/max/mdev = 0.150/0.165/0.179/0.019 ms
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp DROP
sudo: unable to resolve host MUM131
Bad argument `DROP'
Try `iptables -h' or 'iptables --help' for more information.
lab1004@MUM131:~$ sudo iptables -A INPUT -p icmp -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
14 packets transmitted, 0 received, 100% packet loss, time 13000ms

lab1004@MUM131:~$
```

Now try to restrict outgoing icmp packets by adding rule

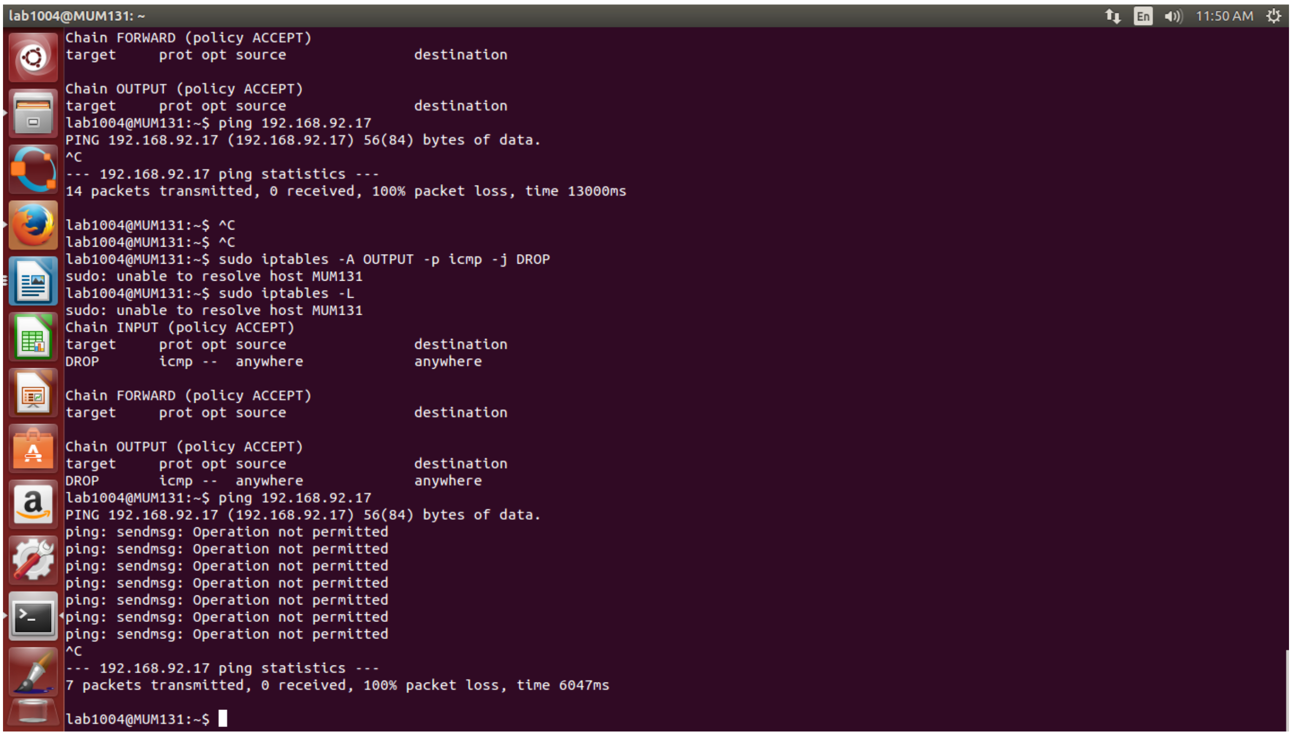
sudo iptables -A **OUTPUT** -p icmp -j DROP

List the rule:

sudo iptables -L

now try to ping neighbour

ping 192.168.92.17



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

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
^C
--- 192.168.92.17 ping statistics ---
14 packets transmitted, 0 received, 100% packet loss, time 13000ms

lab1004@MUM131:~$ ^C
lab1004@MUM131:~$ ^C
lab1004@MUM131:~$ sudo iptables -A OUTPUT -p icmp -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  anywhere             anywhere
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
ping: sendmsg: Operation not permitted
^C
--- 192.168.92.17 ping statistics ---
7 packets transmitted, 0 received, 100% packet loss, time 6047ms

lab1004@MUM131:~$
```

flush all rules and try to ping neighbor

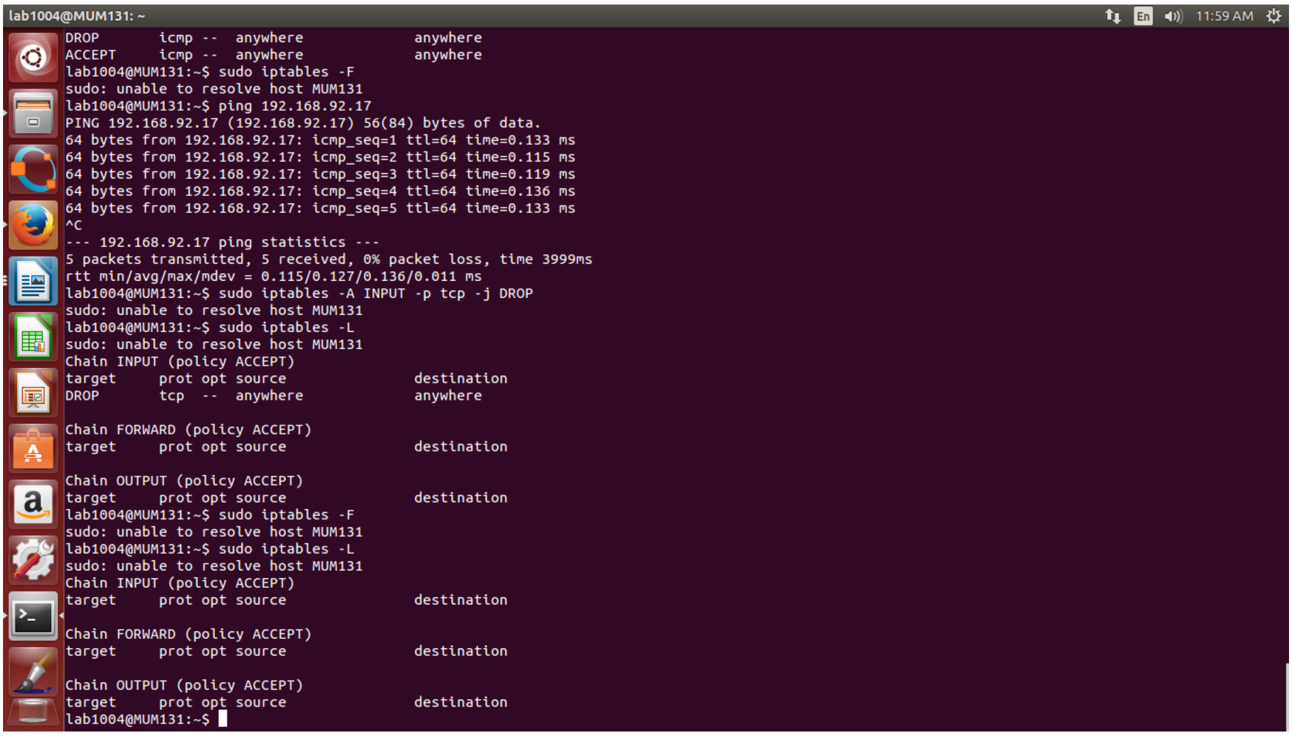
Blocking TCP port traffic will not allow u to browse the Internet

sudo iptables -A INPUT -p tcp -j DROP

List the rule:

sudo iptables -L

Now try to access the internet.. u can,t. Flush the rule n then try to acess internet.. u can.



```
DROP       icmp --  anywhere             anywhere
ACCEPT     icmp --  anywhere             anywhere
lab1004@MUM131:~$ sudo iptables -F
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ ping 192.168.92.17
PING 192.168.92.17 (192.168.92.17) 56(84) bytes of data.
64 bytes from 192.168.92.17: icmp_seq=1 ttl=64 time=0.133 ms
64 bytes from 192.168.92.17: icmp_seq=2 ttl=64 time=0.115 ms
64 bytes from 192.168.92.17: icmp_seq=3 ttl=64 time=0.119 ms
64 bytes from 192.168.92.17: icmp_seq=4 ttl=64 time=0.136 ms
64 bytes from 192.168.92.17: icmp_seq=5 ttl=64 time=0.133 ms
^C
--- 192.168.92.17 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.115/0.127/0.136/0.011 ms
lab1004@MUM131:~$ sudo iptables -A INPUT -p tcp -j DROP
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       tcp  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -F
sudo: unable to resolve host MUM131
lab1004@MUM131:~$ sudo iptables -L
sudo: unable to resolve host MUM131
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
lab1004@MUM131:~$
```

Blocking ICMP packets from specific source machine:

sudo iptables -A INPUT -s 192.168.92.17 -p icmp -j DROP

sudo iptables -L

ping 192.168.92.17

---does not allow (192.168.92.17 can not send u icmp packets)

ping any other machine:

ping 192.168.92.11

---allowed (192.168.92.11 can send u icmp packets)

Types of iptables:

I. IPTABLES TABLES and CHAINS

IPTables has the following 4 built-in tables.

1. Filter Table

Filter is default table for iptables. So, if you don’t define you own table, you’ll be using filter table. Iptables’s filter table has the following built-in chains.

- INPUT chain – Incoming to firewall. For packets coming to the local server.
- OUTPUT chain – Outgoing from firewall. For packets generated locally and going out of the local server.
- FORWARD chain – Packet for another NIC on the local server. For packets routed through the local server.

Type the following command and see the result

```
sudo iptables -t filter -L
```

2. NAT table

Iptable’s NAT table has the following built-in chains.

- PREROUTING chain – Alters packets before routing. i.e Packet translation happens immediately after the packet comes to the system (and before routing). This helps to translate the destination ip address of the packets to something that matches the routing on the local server. This is used for DNAT (destination NAT).
- POSTROUTING chain – Alters packets after routing. i.e Packet translation happens when the packets are leaving the system. This helps to translate the source ip address of the packets to something that might match the routing on the desintation server. This is used for SNAT (source NAT).
- OUTPUT chain – NAT for locally generated packets on the firewall.

Type the following command and see the result

```
sudo iptables -t nat -L
```

3. Mangle table

Iptables’s Mangle table is for specialized packet alteration. This alters QOS bits in the TCP header. Mangle table has the following built-in chains.

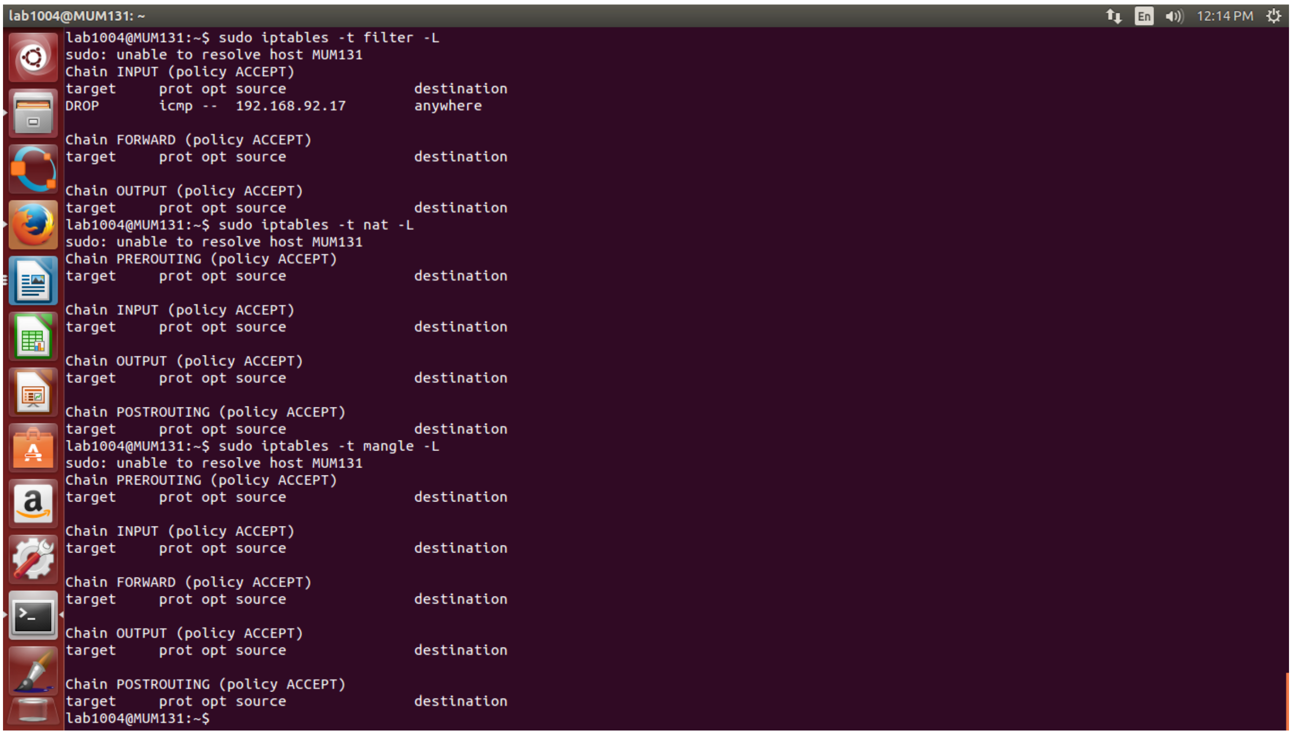
- PREROUTING chain
- OUTPUT chain
- FORWARD chain
- INPUT chain
- POSTROUTING chain

Type the following command and see the result sudo iptables -t nat -L

4. Raw table

Iptable’s Raw table is for configuration excemptions. Raw table has the following built-in chains.

- PREROUTING chain
- OUTPUT chain



```
lab1004@MUM131:~$ sudo iptables -t filter -L
sudo: unable to resolve host MUM131
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       icmp --  192.168.92.17        anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -t nat -L
sudo: unable to resolve host MUM131
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

Chain INPUT (policy ACCEPT)
target     prot opt source               destination

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$ sudo iptables -t mangle -L
sudo: unable to resolve host MUM131
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination

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

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
lab1004@MUM131:~$
```