

LAB – 1 Networking and Firewalls

Version 2

By

Nasirali Kovvuru
nako17@student.bth.se
P No 960521-3231

Changes in this Version according to comments:

Comments:

Task-1:

- 10.0.2.0/24 is not an IP address.

Changes done: Yes, that is not IP address, it is Subnet Mask and I have made corresponding changes in the task.

Task-2:

- Not a “Randomly assigned Ip by the Virtualbox”. It is configured to get that. Check the configuration file.**

Changes done: Yes, I have checked to Configuration file and made changes in the task.

Task-5:

- “193.11.185.57” or “193.11.185.75”?**
- What is the gateway that your host is able to reach to?

Changes done: after referring to screenshot, I found my mistake and changed it to 193.11.185.75. The host gateway is also changes after referring to screenshot, the reason for my mistake will be parallax error while writing the report.

Task-6:

- **“gateway” doesn’t assign ip addresses. Correct your statement.**

Changes done: from the above comment I found wrong in my statement and corrected.

Task 18, 23:

- The problem with that implementation is, if the IP address is changed, **then the rules won’t apply anymore. Think from ports perspective.**

Changes done: yes sir, I did a mistake and corresponding changes are done in those tasks by thinking on ports perspective.

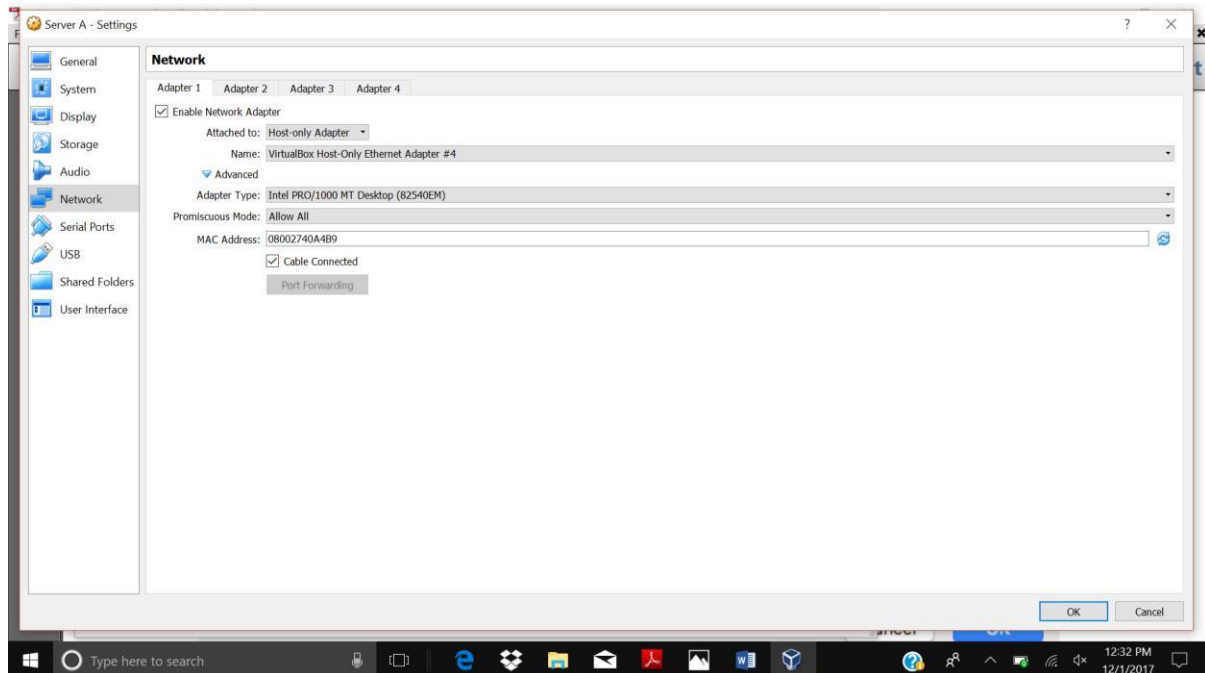
Task 1: To know IP and Mac addresses of each Network card.

Server A

Adapter 1 → ip:192.168.60.1 → Mac:08002740A4B9

Adapter 2 → Subnet:10.0.2.0/24 → Mac:080027868510

Adapter 3 → ip:192.168.70.1 → Mac:080027B666E4



Task2: To know the name of the interface and type of the interface by using Mac addresses from Task1.

Adapter 1 → Host only → ip:192.168.60.1 → interface: enpos3 → Assigned IP by the Virtualbox:192.168.60.100

Adapter 2 → NAT → Subnet:10.0.2.0/24 → interface: enpos9 → supports DHCP:10.0.98.100

Adapter 3 → Host only → ip:192.168.70.1 → interface: enpos8 → Assigned Ip by the Virtualbox:192.168.70.5

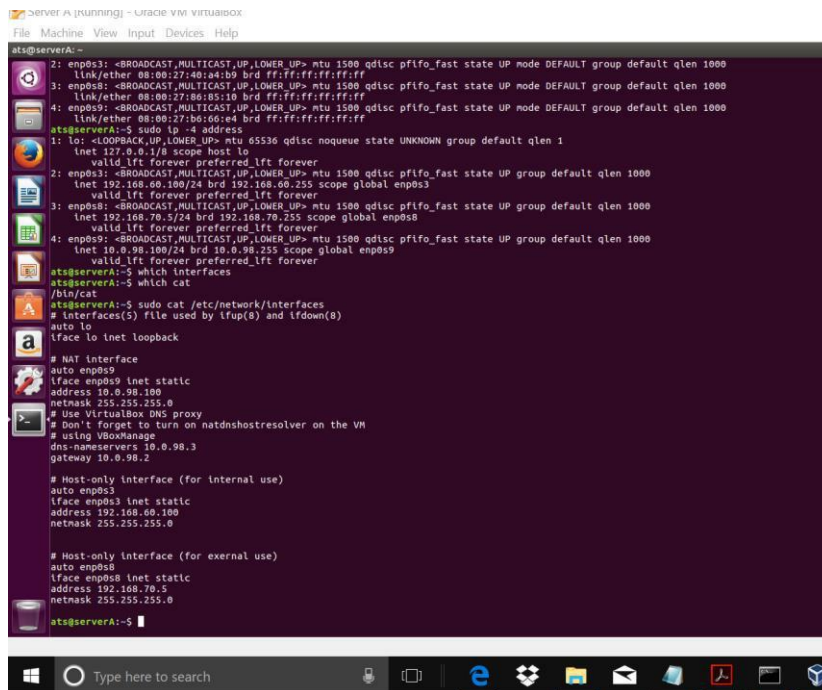
Task3:

```
#sudo cat /etc/network/interfaces
```

```
➔ #NAT interface; enpos9; ip:10.0.98.100 255.255.255.0
```

```
➔ dns-nameservers 10.0.98.3; gateway:10.0.98.2
```

- ➔ NAT interface allows your virtual machine to communicate to outer world, over the NAT interface have their source ip address replaced with gateway
- ➔ DNS name server (10.0.98.3)
- ➔ When virtual box receives DNS requests on this address from the virtual machine, it passes them on to DNS proxy which will convert that IP to corresponding IP address by seeing the Table.
- ➔ When packets are sent to Gateway 10.0.98.2, here virtual box uses its own **NAT engine** to do network address translation. And asks the host to forward the packets to the destination



```

ats@serverA:~$ ifconfig -a
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:40:a4:b9 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:40:b5:19 brd ff:ff:ff:ff:ff:ff
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT group default qlen 1000
    link/ether 08:00:27:b6:66:e4 brd ff:ff:ff:ff:ff:ff
ats@serverA:~$ sudo ip -4 address
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 192.168.60.100/24 brd 192.168.60.255 scope global enp0s3
        valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 192.168.70.5/24 brd 192.168.70.255 scope global enp0s8
        valid_lft forever preferred_lft forever
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    inet 10.0.98.100/24 brd 10.0.98.255 scope global enp0s9
        valid_lft forever preferred_lft forever
ats@serverA:~$ which cat
/usr/bin/cat
ats@serverA:~$ sudo cat /etc/network/interfaces
# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

# NAT interface
auto enp0s9
iface enp0s9 inet static
    address 10.0.98.100
    netmask 255.255.255.0

# Use VirtualBox DNS proxy
# Don't forget to turn on natdnshostresolver on the VM
# using VBoxManage
dns-nameservers 10.0.98.3
gateway 10.0.98.2

# Host-only interface (for internal use)
auto enp0s3
iface enp0s3 inet static
    address 192.168.60.100
    netmask 255.255.255.0

# Host-only interface (for external use)
auto enp0s8
iface enp0s8 inet static
    address 192.168.70.5
    netmask 255.255.255.0
ats@serverA:~$
  
```

Task4:

- ➔ By seeing the result from command (`$sudo ifconfig -a`), we can see that there are different ip's which are not configured at host-only network
- ➔ In my case, I have configured host only networks and NAT networks with 192.168.60.1(255.255.255.0), 192.168.70.1(255.255.255.0), 10.0.2.0/24(255.255.255.0) respectively but I observed that all the addresses are varied after switching on the virtual machine.
- ➔ From this result, by moving to internal virtual guest O.S settings, I came to conclusion that, our **virtual box is assigning different ip addresses to different guests to make it possible to use the same card(adapter) by different guests.**
- ➔ Very important observation is that, **Mac address assignment is constant, irrespective of dynamic IP address assignment, so**

from the above observation I practically came to conclusion that **observing Mac address instead host-only network IP address is preferable.**

➔ And, from my base machine command prompt I found that **Host-only network IP addresses are same as I configured at first.**

➔ Output:

enpos3 is interface card is connected to host only adapter#4

enpos9 interface card is connected to host only adapter#5

enpos3 interface card is connected to NAT Adapter

```
Select C:\WINDOWS\system32\cmd.exe
Ethernet adapter VirtualBox Host-Only Network #4:

Connection-specific DNS Suffix  . : 
Link-local IPv6 Address . . . . . : fe80::e41d:f3ca:670d:204a%46
IPv4 Address. . . . . : 192.168.60.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 

Ethernet adapter VirtualBox Host-Only Network #5:

Connection-specific DNS Suffix  . : 
Link-local IPv6 Address . . . . . : fe80::3867:eb11:563e:1ac9%50
IPv4 Address. . . . . : 192.168.70.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 

Ethernet adapter VirtualBox Host-Only Network #6:

Connection-specific DNS Suffix  . : 
Link-local IPv6 Address . . . . . : fe80::dd8c:7980:d479:490c%54
IPv4 Address. . . . . : 192.168.80.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 

Ethernet adapter VirtualBox Host-Only Network #2:

Connection-specific DNS Suffix  . : 
Link-local IPv6 Address . . . . . : fe80::b9e1:b4b:6e9b:6ac0%8
IPv4 Address. . . . . : 192.168.1.1
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 

Ethernet adapter VirtualBox Host-Only Network #3:

Connection-specific DNS Suffix  . : 
Link-local IPv6 Address . . . . . : fe80::8597:842c:a032:10e2%14
IPv4 Address. . . . . : 192.168.2.2
Subnet Mask . . . . . : 255.255.255.0
Default Gateway . . . . . : 

Wireless LAN adapter Local Area Connection* 3:

Type here to search
```

TASK 5: Routing table in the host Operating system

➔ in my case, I can reach the default gateway through Wifi LAN Adapter from the Host os(Windows) is through its ip address 193.11.184.1, and the interface is 193.11.185.75

➔ In the output from command (#route -4 PRINT), I got it as **193.11.184.1.**

```

C:\WINDOWS\system32\cmd.exe
C:\Users\Nasir Ali>route -4 PRINT
=====
Interface List
=====
46...0a 00 27 00 00 2e .....VirtualBox Host-Only Ethernet Adapter #4
50...0a 00 27 00 00 32 .....VirtualBox Host-Only Ethernet Adapter #5
54...0a 00 27 00 00 36 .....VirtualBox Host-Only Ethernet Adapter #6
8...0a 00 27 00 00 08 .....VirtualBox Host-Only Ethernet Adapter #2
14...0a 00 27 00 00 0e .....VirtualBox Host-Only Ethernet Adapter #3
12...30 e3 7a 0d e0 95 .....Microsoft Wi-Fi Direct Virtual Adapter
36...00 50 56 c0 00 01 .....VMware Virtual Ethernet Adapter for VMnet1
13...30 e3 7a 0d e0 94 .....Intel(R) Dual Band Wireless-AC 3168
1.....Software Loopback Interface 1
21...00 00 00 00 00 00 e0 Microsoft Teredo Tunneling Adapter
=====

IPv4 Route Table
=====
Active Routes:
Network Destination        Netmask          Gateway             Interface          Metric
0.0.0.0                    0.0.0.0          193.11.184.1        193.11.185.57      55
127.0.0.0                  255.0.0.0        On-link             127.0.0.1          331
127.0.0.1                  255.255.255.255 On-link             127.0.0.1          331
127.255.255.255            255.255.255.255 On-link             127.0.0.1          331
192.168.1.0                 255.255.255.0    On-link             192.168.1.1        281
192.168.1.1                 255.255.255.255 On-link             192.168.1.1        281
192.168.1.255               255.255.255.255 On-link             192.168.1.1        281
192.168.2.0                 255.255.255.0    On-link             192.168.2.2        281
192.168.2.2                 255.255.255.255 On-link             192.168.2.2        281
192.168.2.255               255.255.255.255 On-link             192.168.2.2        281
192.168.60.0                255.255.255.0    On-link             192.168.60.1       281
192.168.60.1                255.255.255.255 On-link             192.168.60.1       281
192.168.60.255              255.255.255.255 On-link             192.168.60.1       281
192.168.70.0                255.255.255.0    On-link             192.168.70.1       281
192.168.70.1                255.255.255.255 On-link             192.168.70.1       281
192.168.70.255              255.255.255.255 On-link             192.168.70.1       281
192.168.80.0                255.255.255.0    On-link             192.168.80.1       281
192.168.80.1                255.255.255.255 On-link             192.168.80.1       281
192.168.80.255              255.255.255.255 On-link             192.168.80.1       281
192.168.180.0               255.255.255.0    On-link             192.168.180.1      291
192.168.180.1               255.255.255.255 On-link             192.168.180.1      291
192.168.180.255             255.255.255.255 On-link             192.168.180.1      291

```

TASK 6: Routing table in the guest Operating system

➔ From the command (`$route -n`), I found my default gateway (DNS proxy) at 10.0.98.2 to reach that adapter through the interface **enps9**.

```

Server A [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

ats@serverA:~$ netstat -4 -rn
Kernel IP routing table
Destination        Gateway            Genmask           Flags   MSS Window  irtt Iface
0.0.0.0            10.0.98.2         0.0.0.0           UC      0 0        0 enps9
10.0.98.0          0.0.0.0           255.255.255.0     U        0 0        0 enps9
109.254.0.0        0.0.0.0           255.255.0.0       U        0 0        0 enps8
192.168.60.0        0.0.0.0           255.255.255.0     U        0 0        0 enps3
192.168.70.0        0.0.0.0           255.255.255.0     U        0 0        0 enps8

```

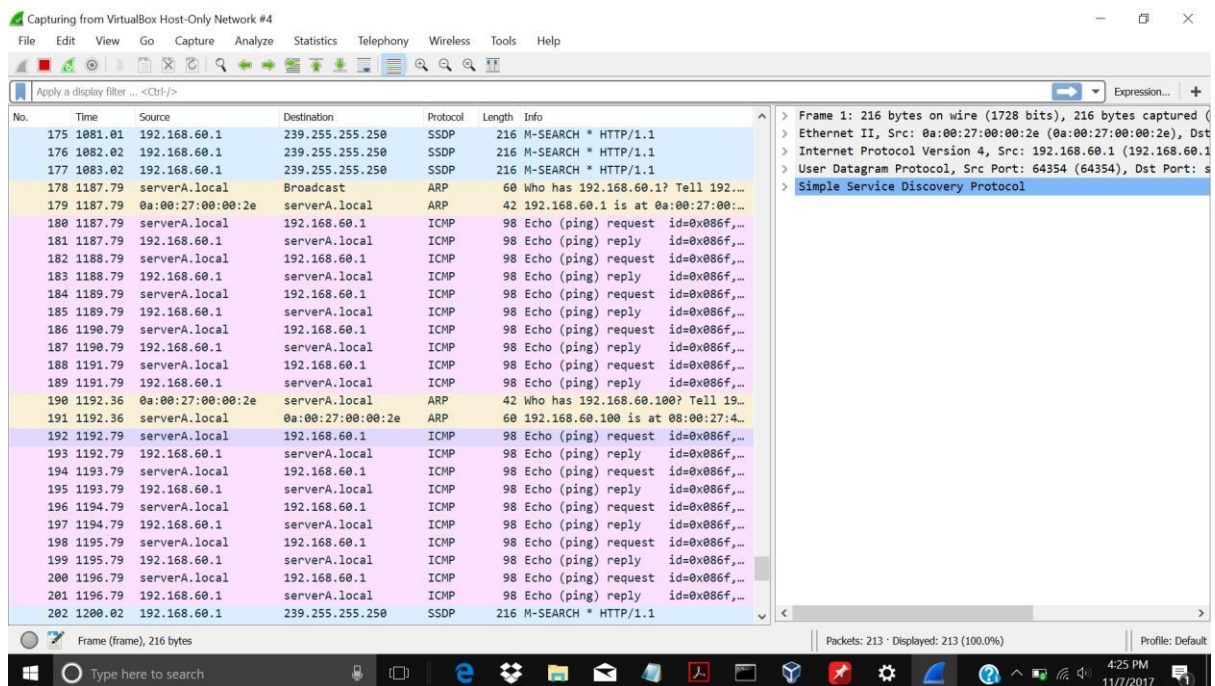
➔ From the above result I can conclude my exploration that, NAT server acts like a **switch to all the virtual machines** to communicate, but whereas to send the packets to outer world it should make use of **Proxy server as a gateway**

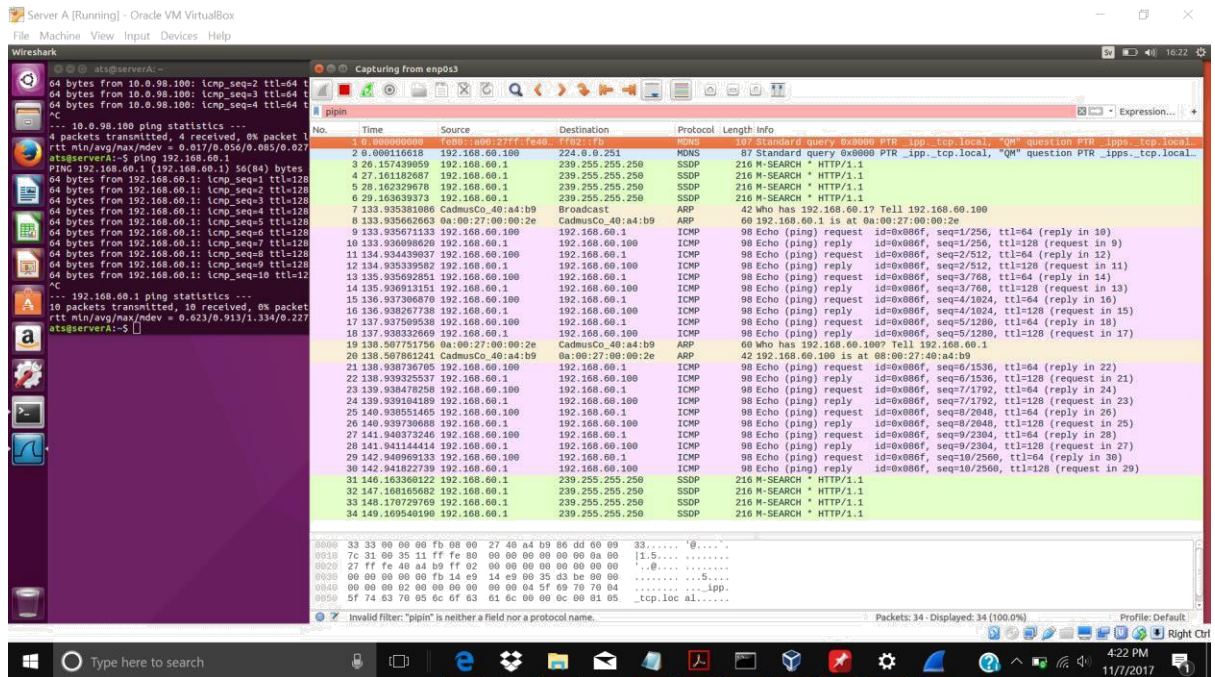
➔ We can conclude from the above screenshot that through host-only interface the guest OS can reach the default gateway for host OS.

➔ Here, **we can reach to gateway via NAT interface only, because where our DNS server is configured.**

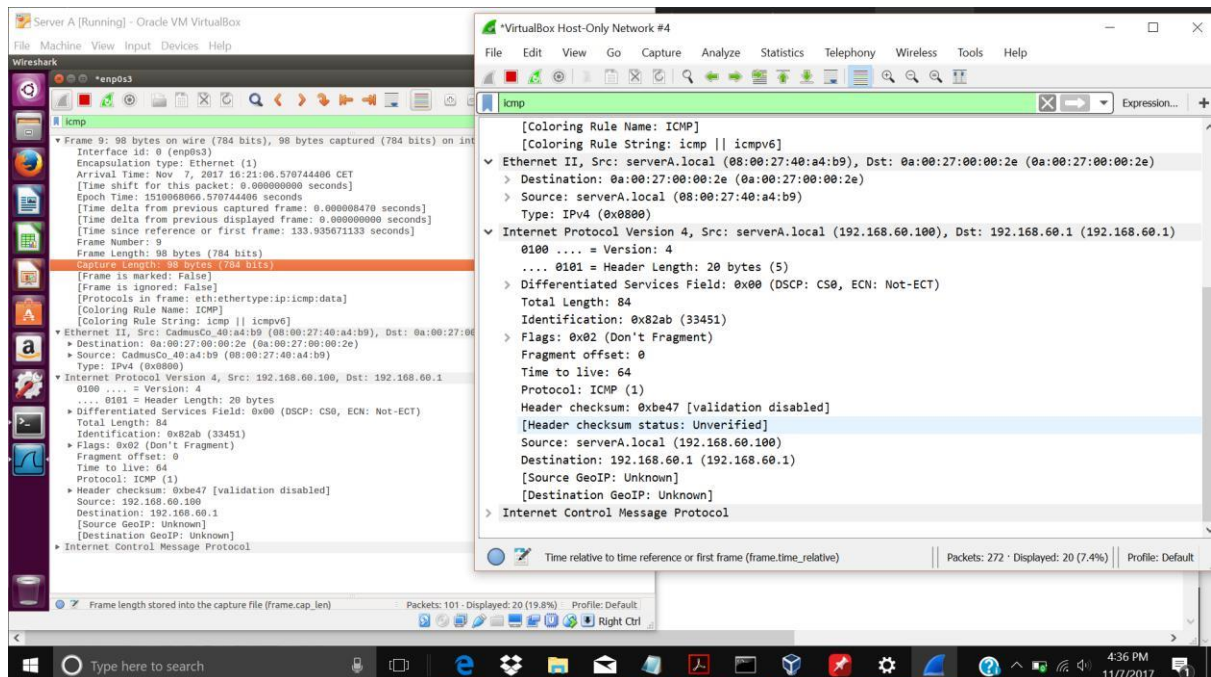
➔ Wireshark Traffic:

Packets are captured at both guest and host PC's for host-only interface 192.168.60.100.

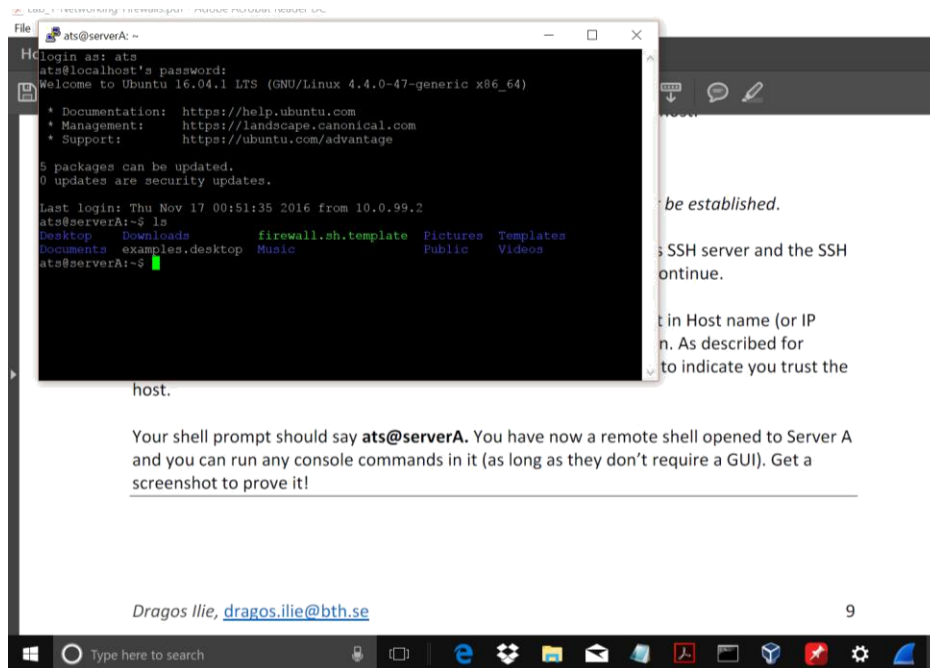




TASK 7: we can see that the Wireshark ICMP traffic is same at both ends.

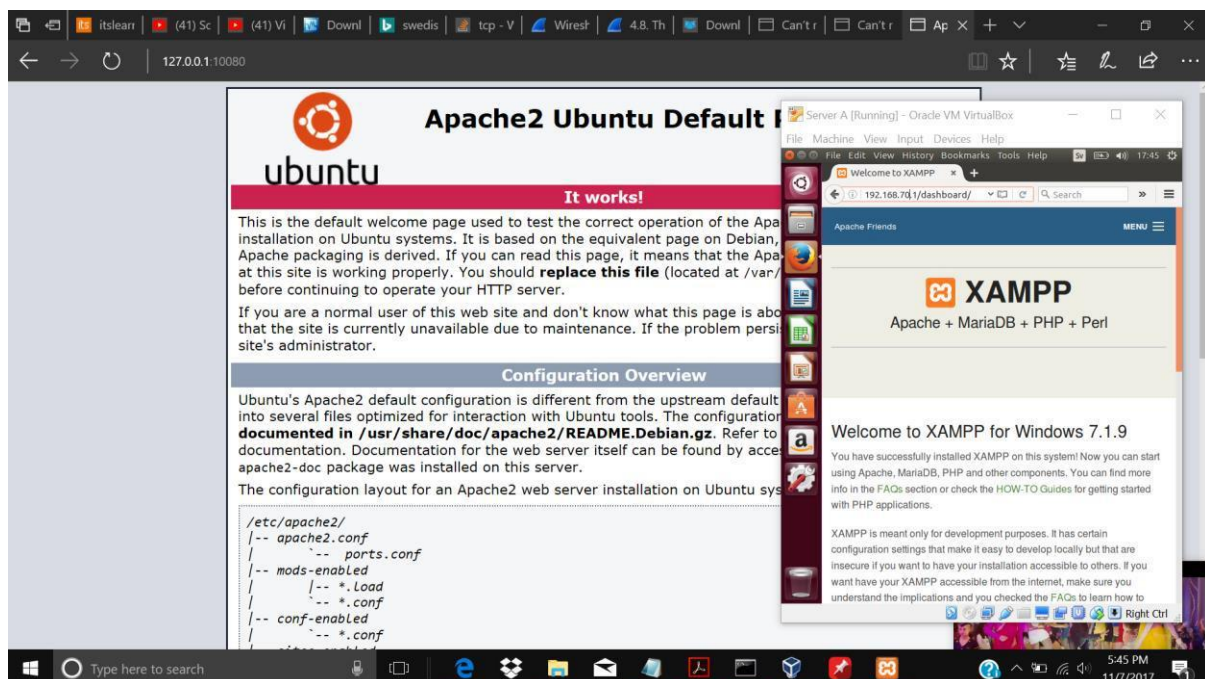


Task 8: SSH configuration



Task9: add forwarding rules for http and https in virtual box.

For http → 10080, and for https → 10443, these port numbers are used



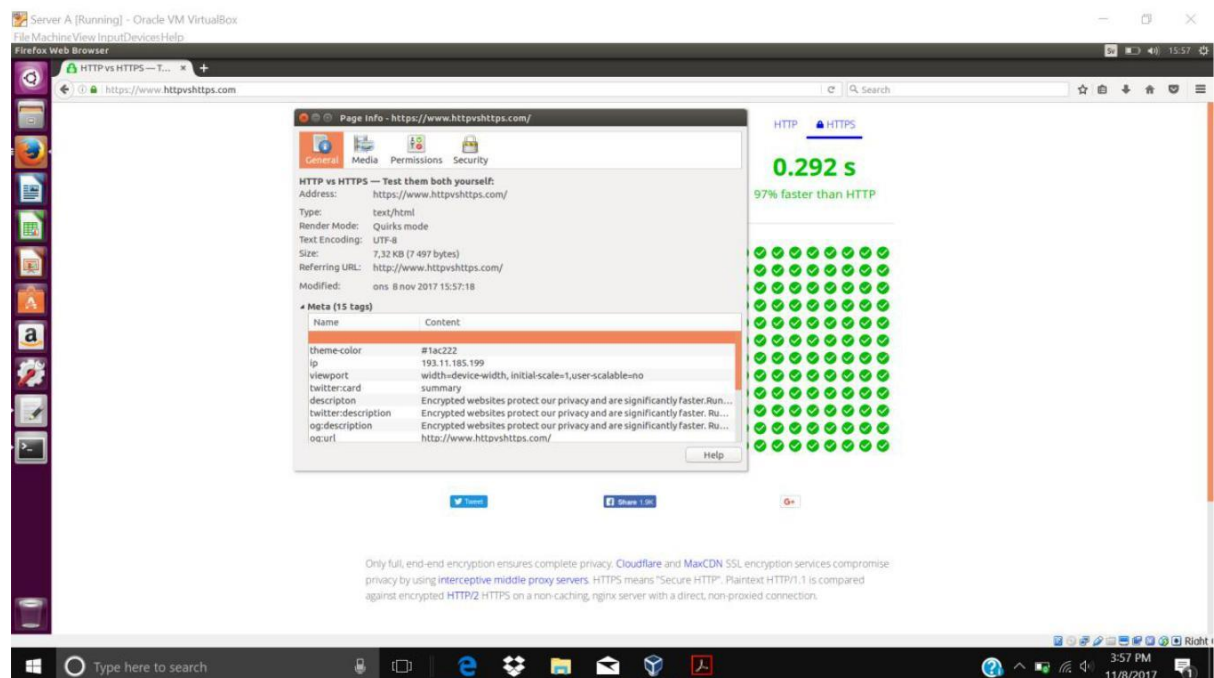
Task 10: Default firewall policy and rules.

\$iptables tool is used to find the default policies, the following screenshot shows them. By default all the filters and chains are in accept list.

The following commands helps in listing the default firewall rules:

\$sudo iptables - -list → by default it shows filter table policies.

`$sudo iptables -t raw - -list` → this command shows raw table policies.

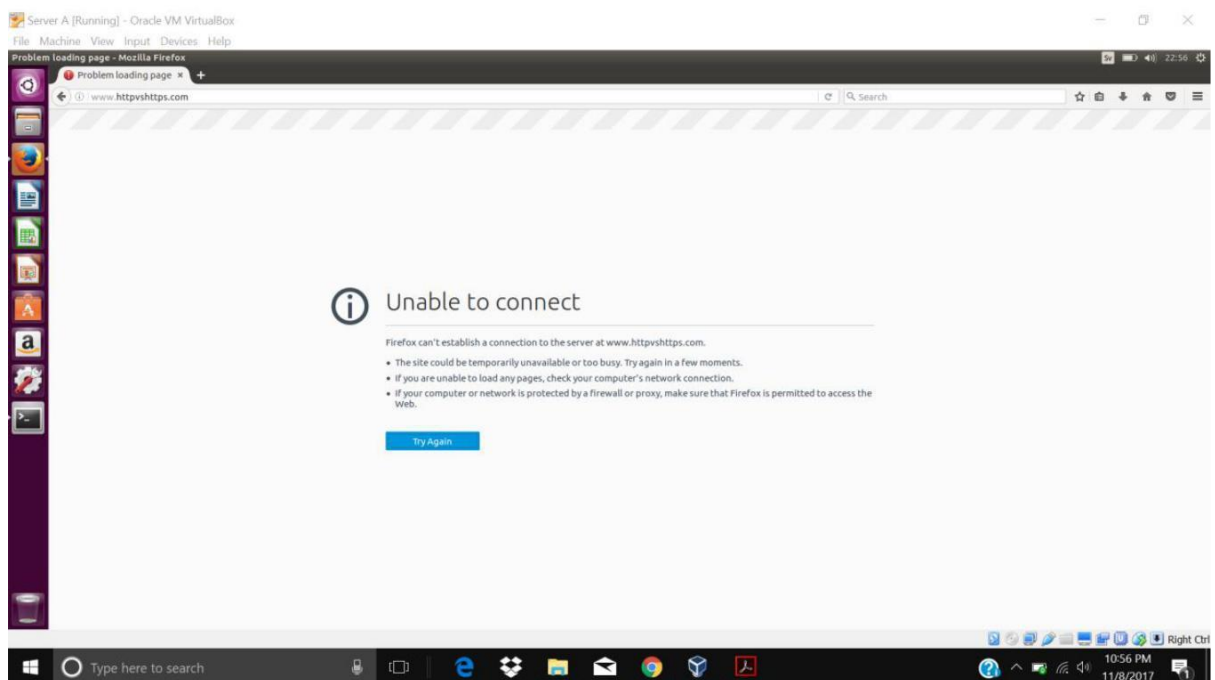


```
$sudo iptables -A INPUT -p tcp -dport 80 -j REJECT
```

- ➔ This command rejects all the http browsing.
- ➔ **Note:** the above command only affects the filter table, output chain; the reason behind is, it is the default table for iptables.

```

ats@serverA:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j REJECT
iptables v1.6.0: can't initialize iptables table 'filter': Table does not exist (do you need to insmod?)
perhaps iptables or your kernel needs to be upgraded.
***
ats@serverA:~$ sudo iptables -A OUTPUT -p tcp --dport 80 -j REJECT
[sudo] password for ats:
ats@serverA:~$ sudo iptables -L
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
ats@serverA:~$ sudo iptables -t nat --list
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
ats@serverA:~$ sudo iptables -t mangle --list
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
ats@serverA:~$ sudo iptables -t raw --list
Chain PREROUTING (policy ACCEPT)
target prot opt source destination
Chain OUTPUT (policy ACCEPT)
target prot opt source destination
ats@serverA:~$
  
```

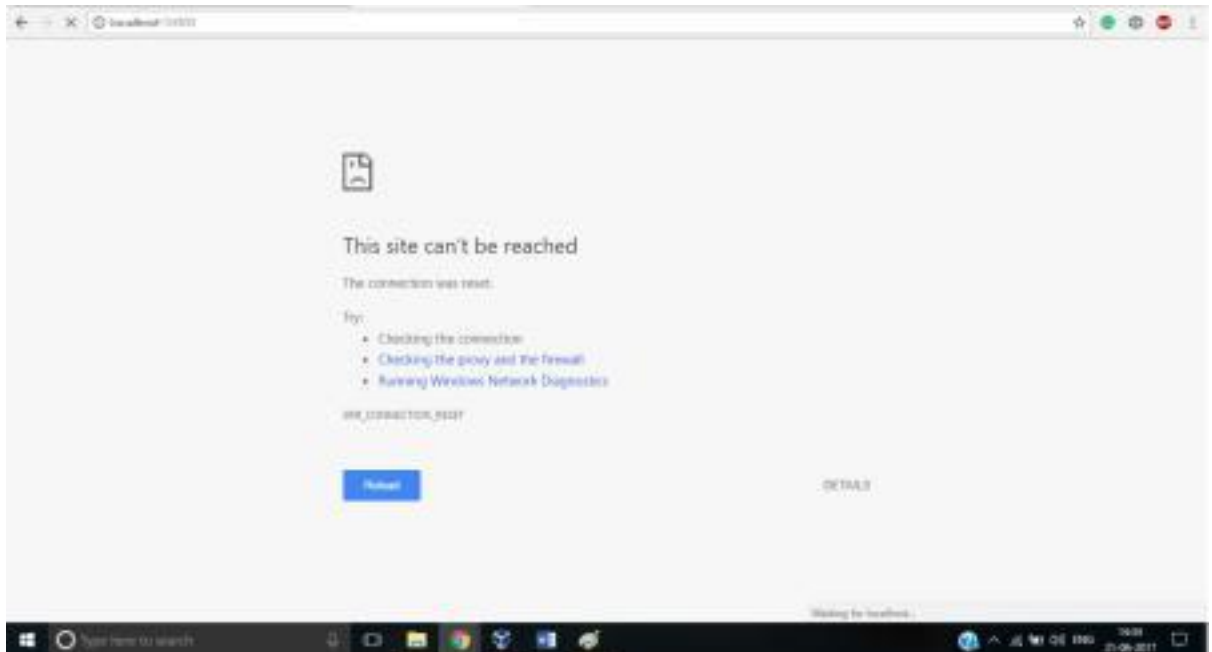


TASK 12: Block Apache web server from serving content over http

- ➔ I have achieved the above task by rejecting the output traffic from firewall. Or in other way we can reject the traffic going away from

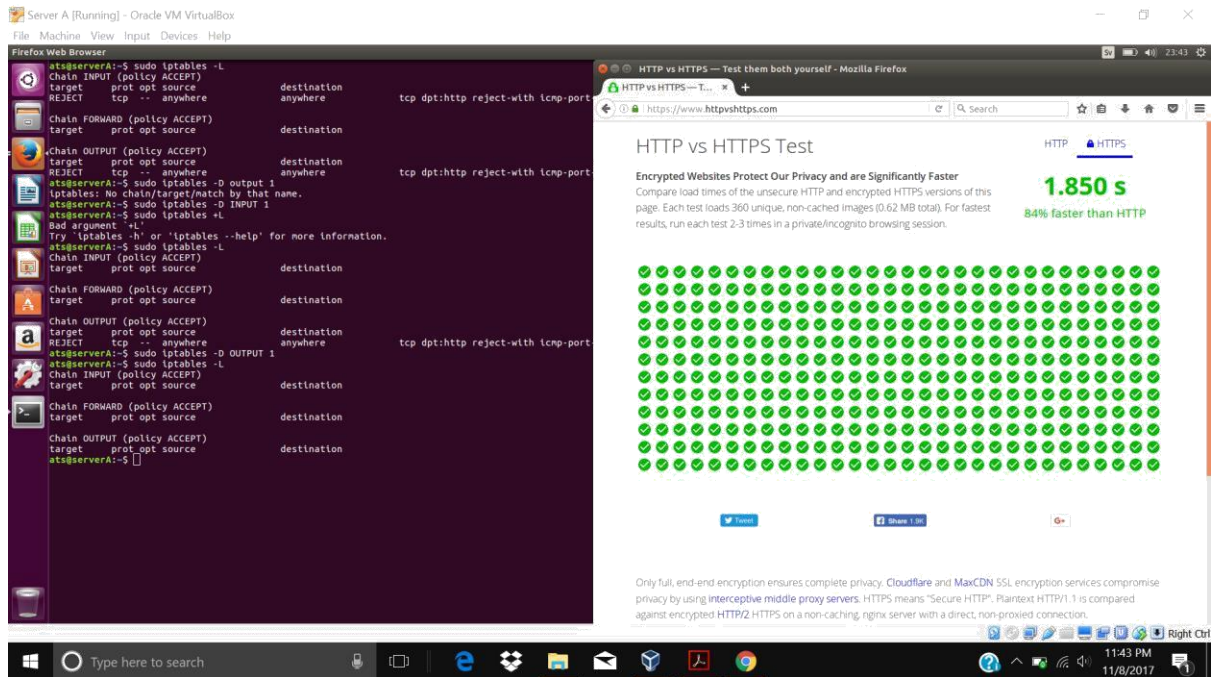
loopback interface as the apache server resides there. as shown in the following screenshot.

- ➔ `$sudo iptables -A OUTPUT -p tcp --dport 80 -j Reject`
- ➔ (or) `$sudo iptables -A OUTPUT -p tcp -s 127.0.0.1/24 -dport 80 -j Reject`.



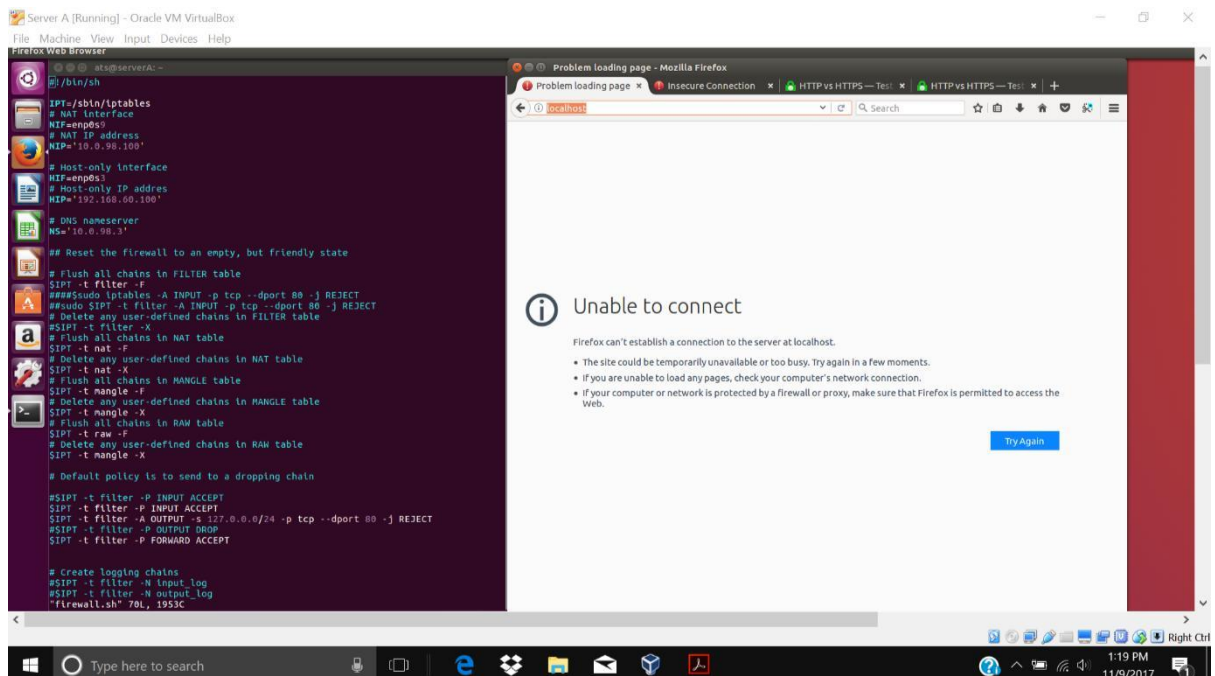
Task 13: Unblock HTTP browsing in the guest O.S

- ➔ Undo tasks 11 and 12
- ➔ The following commands helps
- ➔ `$sudo iptables -D OUTPUT1 → -D to Delete`
- ➔ `$sudo iptables -D INPUT1`



TASK 14: use firewall.sh to configure the firewall

- ➔ Modify the script to such that guest OS can view http and https services but apache2 service should be blocked.
- ➔ The following screenshot shows such behaviour
- ➔ `$ iptables -A OUTPUT -p tcp -s 127.0.0.1/24 -dport 80 -j Reject.`



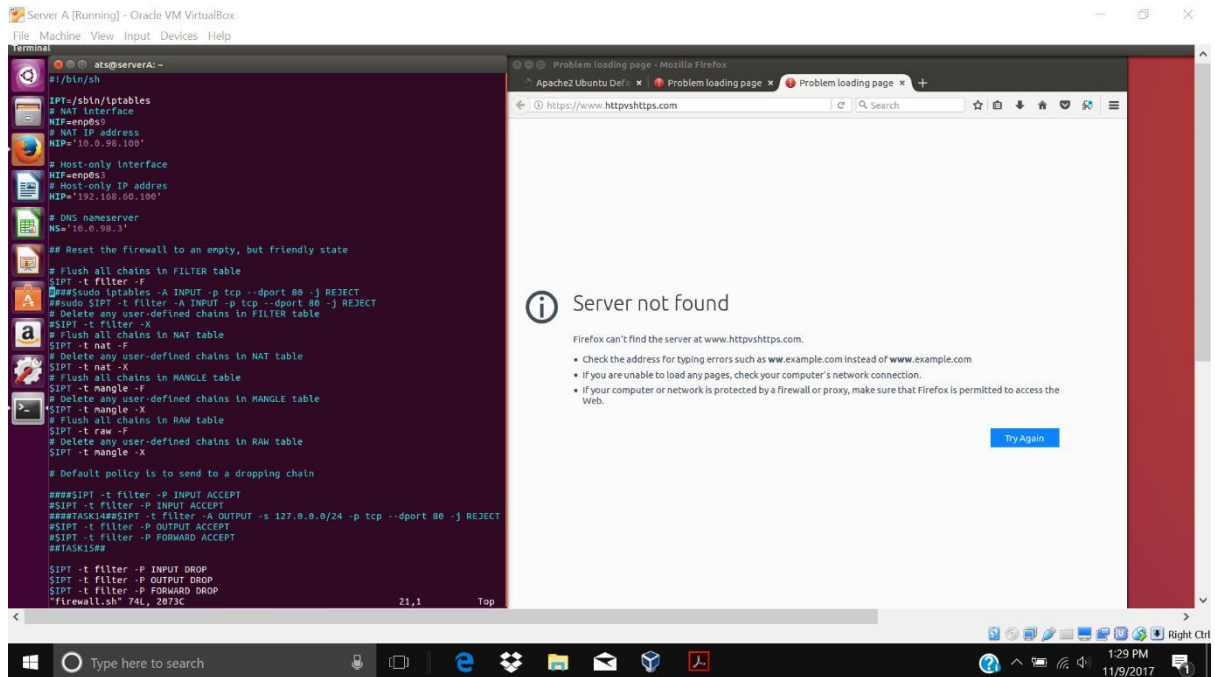
TASK 15: change default firewall policy to

drop ➔ `$iptables -t filter -P INPUT DROP`

→ `$iptables -t filter -P OUTPUT`

`DROP` → `$iptables -t filter -P`

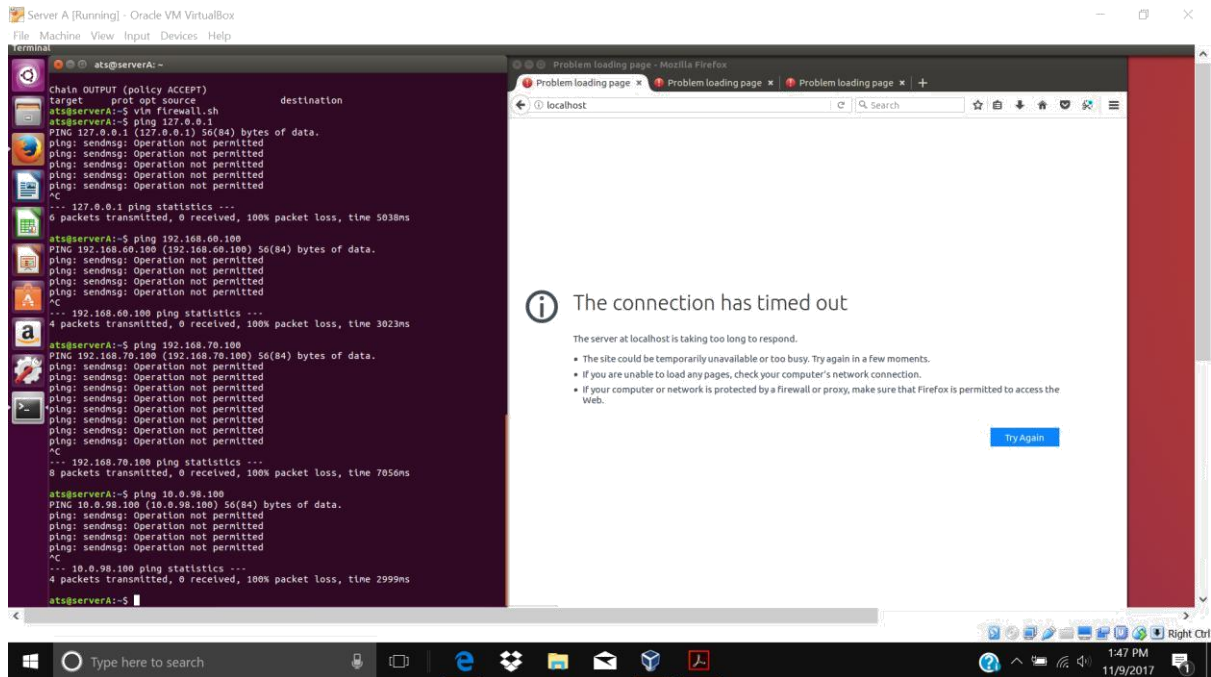
`FORWARD DROP`



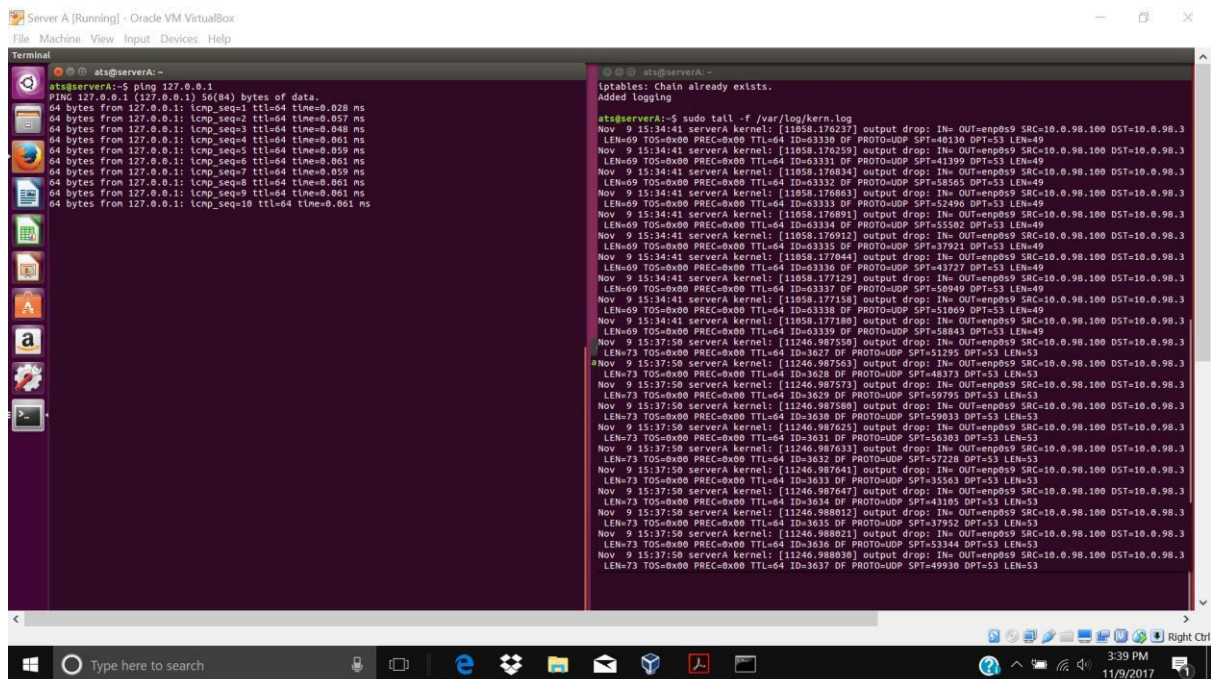
Result → localhost not loaded, http, https are not responding.

Execute the script to install the new policy to ping outside world from the virtual machine, try to ping the loopback interface.

- ➔ Following rules make the above requirement fulfil.
- ➔ `$ iptables -I OUTPUT -p icmp -j ACCEPT`
- ➔ `$iptables -I INPUT -p icmp -j ACCEPT`



➔ Live logs from the kernel is shown below.



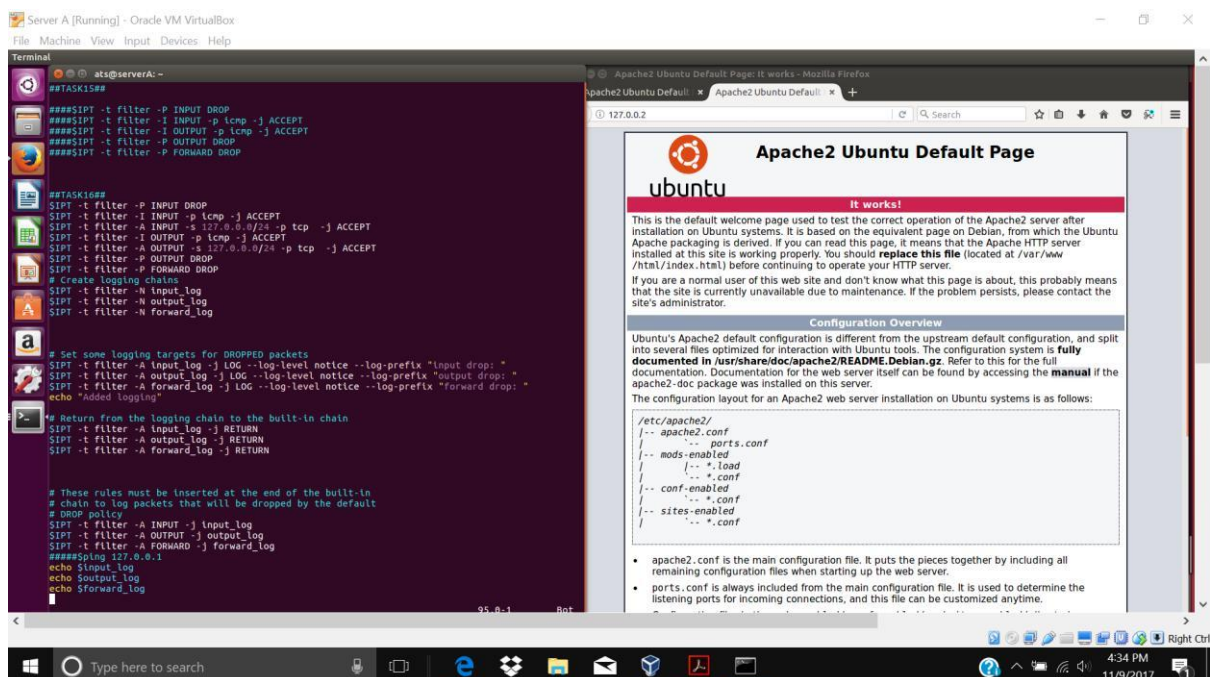
TASK 16: Logging Dropped packets

- ➔ By executing the command `$sudo tail -f var/log/kern.log`
- ➔ And by adding commands: `echo $input_log`, `echo $output_log`, `echo $forward_log`
- ➔ We can see the kernel log
- ➔ In the output we can find that out ping messages are blocking
- ➔ **What do you think is blocking your ping?**

- ➔ In my case my I am getting some error message as in description screenshot (output drop) and in my case source and destination addresses are in between NAT's DHCP and its DNS nameserver.
- ➔ The Possible reason behind it, will be; we already know from previous tasks that NAT interface is something out from the virtual machine and it stays on virtualbox hypervisor and the script which tells to DROP all the output packets except ICMP but here other management messages are going in a particular interval gap; so they are blocking by the firewall.
- ➔ So that packets got dropped.

TASK 17: Enable Traffic from Loopback interface

- ➔ `$ iptables -I OUTPUT -s 127.0.0.1/24 -p tcp -j ACCEPT`
- ➔ `$ iptables -I INPUT -s 127.0.0.1/24 -p tcp -j ACCEPT`
- ➔ The above commands are used to enable only TCP traffic but if you want to enable all types of traffic then following command helps.
- ➔ `$ iptables -I OUTPUT -s 127.0.0.1/24 -j ACCEPT`
- ➔ `$ iptables -I INPUT -s 127.0.0.1/24 -j ACCEPT`



TASK 18: Allow server A to ping the other interfaces.

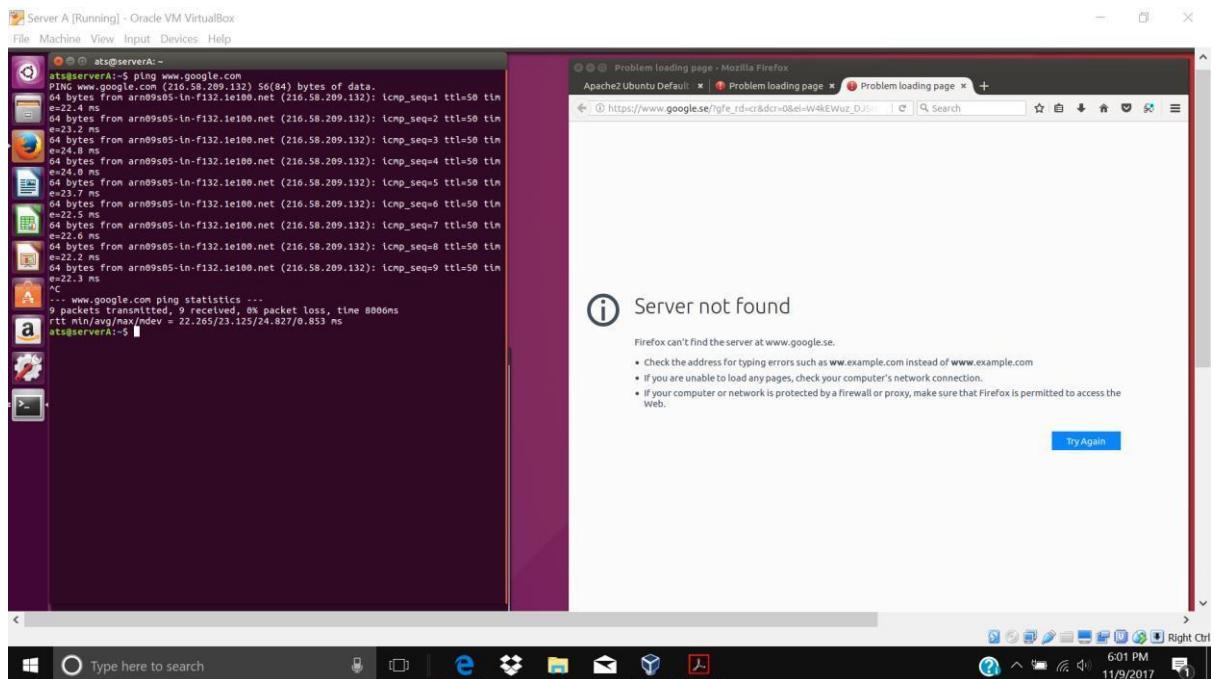
- ➔ `$ iptables -I OUTPUT -p icmp -j ACCEPT`
- ➔ `$ iptables -I INPUT -p icmp -j ACCEPT`
- ☐ To ping with www.google.se we should accept DNS name server

➔ We can go to DNS name server only via NAT engine, so it is essential to allow both to communicate with us such we have the service.

➔ Following commands helps in that work

```
$IPT -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
```

```
$IPT -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

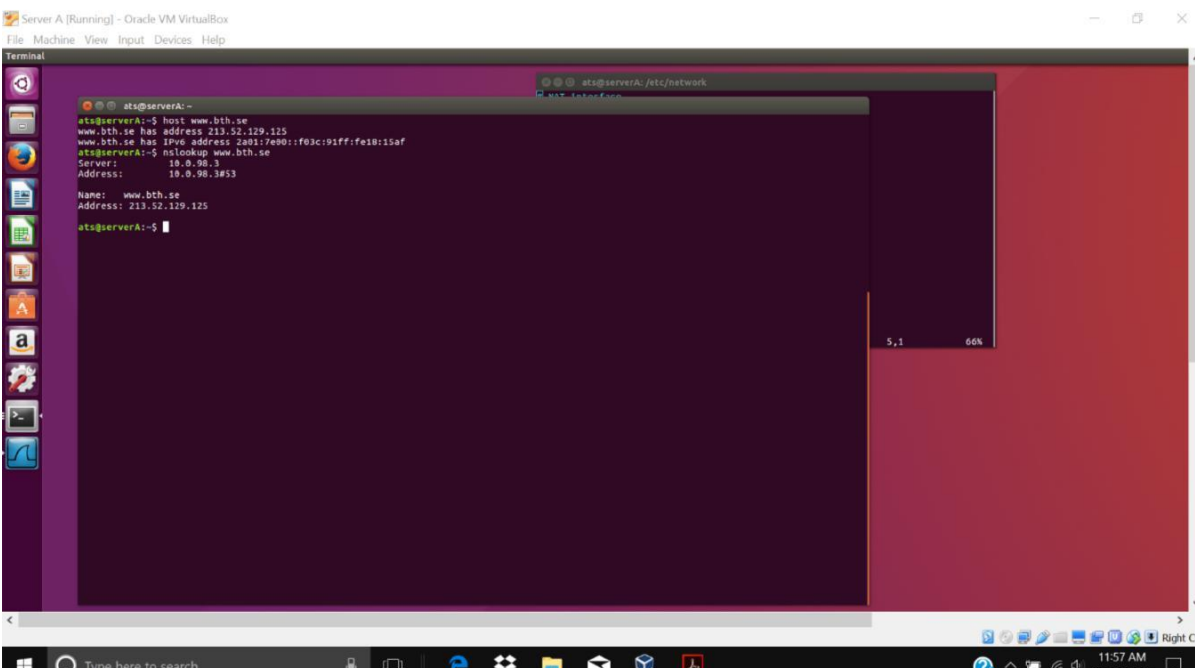


TASK 19: Allow server A to ping all hosts

This task is bit like before task, but to accomplish this task I have used a debugging and sniffing tool called wireshark to know where my packets got dropped and to add appropriate rules.

To allow the server to ping all hosts. By adding the following rules to the firewall.sh script and executing it, we are allowing the firewall to accept the outgoing ICMP traffic to any server and corresponding ICMP replies.

```
$IPT -A OUTPUT -p udp -m conntrack --ctstate NEW,ESTABLISHED  
- j ACCEPT
```

Server A [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Terminal

ats@serverA:~

ats@serverA:~\$ host www.bth.se
www.bth.se has address 213.52.129.125
www.bth.se has ipvc address 2a01:fe00::f03c:91ff:fe10:15af

ats@serverA:~\$ nslookup www.bth.se
Server: 19.0.96.3
Address: 19.0.96.3833

Name: www.bth.se
Address: 213.52.129.125

ats@serverA:~\$

ats@serverA:/etc/network

5.1 66%

Type here to search

11:57 AM
11/10/2017

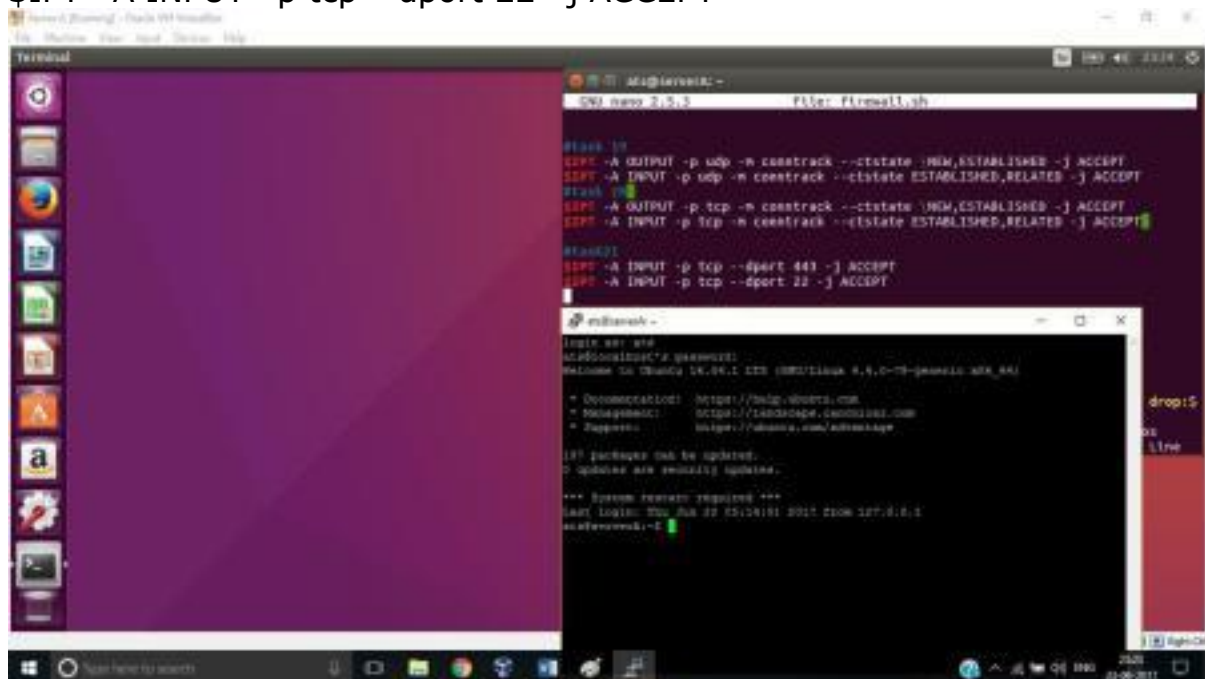
- ➔ To make the state full firewall, following two commands are used
- ➔ `$iptables -t filter -A INPUT -p tcp -m Conntrack --cstate ESTABLISHED,RELATED -j ACCEPT`

- ➔ It is for filtering the input traffic which is already in established or establishing modes
- ➔ `$iptables -t filter -I OUTPUT -p tcp -m conntrack --cstate NEW,ESTABLISHED -j ACCEPT`
- ➔ This command will filter, all the other requests going out except NEW and ESTABLISHED modes of operation.
- ➔ To disable SSH we should comment the command that is for accepting all the TCP traffic.
- ➔ `#iptables -t filter -I OUTPUT -p tcp -j ACCEPT`
- ➔ `#iptables -t filter -I INPUT -p tcp -j ACCEPT`

TASK 21: Enable SSH and HTTPS contents from apache2 server for web browser on host (base machine).

To Enable SSH and HTTPS content from apache2 server for web browser on HOST, add the following commands.

```
$IPT -A INPUT -p tcp --dport 443 -j ACCEPT
$IPT -A INPUT -p tcp --dport 22 -j ACCEPT
```



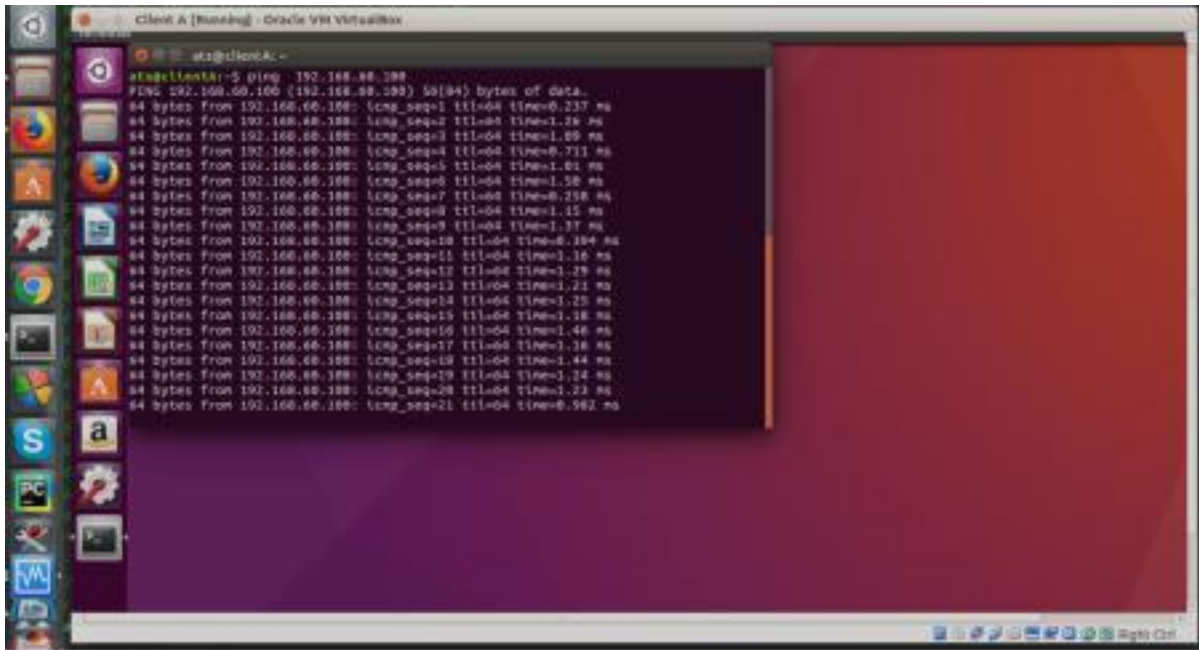
Task 22:

To add the firewall rules to ping server A from client A. add the following rules in firewall.sh

```
$IPT -t filter -A INPUT -p icmp -s 192.168.60.111 -j ACCEPT
$IPT -t filter -A OUTPUT -p icmp -s 192.168.60.111 -j ACCEPT
```

(or)

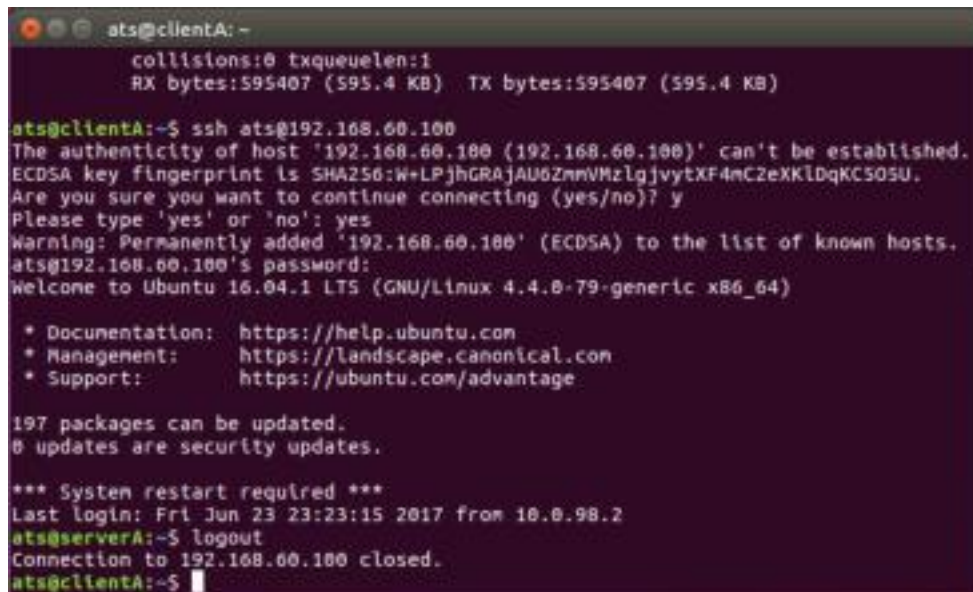
```
$iptables -A INPUT -p icmp --icmp-type echo-request -j ACCEPT
$IPTABLES -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
```



Task 23:

To fix the firewall rules such that we can SSH from Client A to Server A. Add the following rules to the firewall.

```
sudo iptables -I INPUT -s 192.168.60.111 -p tcp -m tcp --dport 10022 -j ACCEPT
```

A terminal window titled 'ats@clientA: ~' showing network statistics at the top: 'collisions:0 txqueuelen:1' and 'RX bytes:595407 (595.4 KB) TX bytes:595407 (595.4 KB)'. The user runs 'ssh ats@192.168.60.100'. The terminal shows a warning about the host's authenticity, a fingerprint, and a confirmation to add it to known hosts. The user enters 'y'. The terminal then shows the Ubuntu 16.04.1 LTS login screen with the username 'ats'. It lists documentation, management, and support links. It also shows that 197 packages can be updated and 0 security updates are available. A system restart is required. The last login is from 10.0.98.2. The user enters 'logout'. The terminal shows 'Connection to 192.168.60.100 closed.' and the user returns to the 'ats@clientA: ~' prompt.

```
ats@clientA: ~
collisions:0 txqueuelen:1
RX bytes:595407 (595.4 KB) TX bytes:595407 (595.4 KB)

ats@clientA:~$ ssh ats@192.168.60.100
The authenticity of host '192.168.60.100 (192.168.60.100)' can't be established.
ECDSA key fingerprint is SHA256:W+LPjhGRAJAU6ZmmVMzlgjvytXF4mC2eXKlDqKC505U.
Are you sure you want to continue connecting (yes/no)? y
Please type 'yes' or 'no': yes
Warning: Permanently added '192.168.60.100' (ECDSA) to the list of known hosts.
ats@192.168.60.100's password:
Welcome to Ubuntu 16.04.1 LTS (GNU/Linux 4.4.0-79-generic x86_64)

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

197 packages can be updated.
0 updates are security updates.

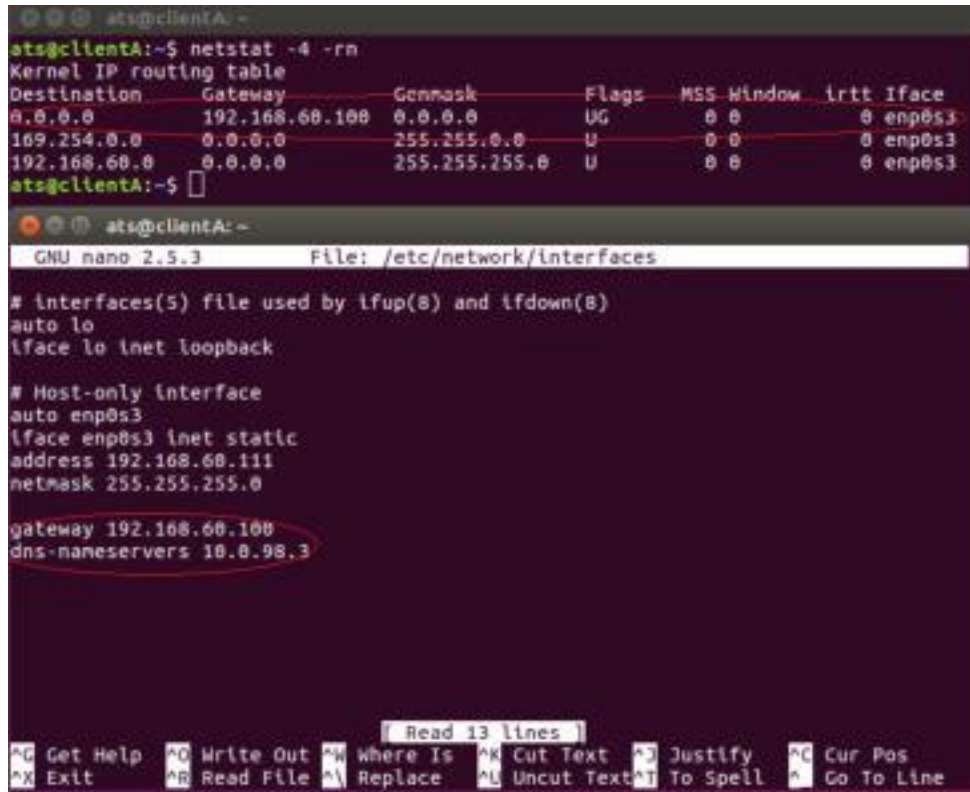
*** System restart required ***
Last login: Fri Jun 23 23:23:15 2017 from 10.0.98.2
ats@serverA:~$ logout
Connection to 192.168.60.100 closed.
ats@clientA:~$
```

For ssh I used below command.

```
Sudo ssh -p 10022 ats@192.168.60.100
```

Task 24:

In this task and the gateway and dns-nameserver in **/etc/network/interface** file in client A, so that we can able to add gateway and dns-servername to client A.



The screenshot shows two windows from a terminal session on client A. The top window displays the output of the `netstat -4 -rn` command, showing the kernel IP routing table. The bottom window shows the `/etc/network/interfaces` file being edited in nano, with the gateway and dns-nameservers lines highlighted.

```
ats@clientA:~$ netstat -4 -rn
Kernel IP routing table
Destination      Gateway         Genmask         Flags         MSS Window  irtt Iface
0.0.0.0          192.168.68.100 0.0.0.0         UG            0 0        0 enp0s3
169.254.0.0      0.0.0.0         255.255.0.0     U             0 0        0 enp0s3
192.168.68.0     0.0.0.0         255.255.255.0   U             0 0        0 enp0s3
ats@clientA:~$
```

```
GNU nano 2.5.3 File: /etc/network/interfaces

# interfaces(5) file used by ifup(8) and ifdown(8)
auto lo
iface lo inet loopback

# Host-only interface
auto enp0s3
iface enp0s3 inet static
address 192.168.68.111
netmask 255.255.255.0

gateway 192.168.68.100
dns-nameservers 10.0.98.3
```

Task 25:

To execute following command in the terminal of Server A so that IP forwarding is enabled on the Server A. This will forward the packets from enp0s3 to enp0s9

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo sysctl -p
```

Task 26:

To change the iptables rules to forward packets. add the following rules to forward packets from enp0s3 to enp0s9.

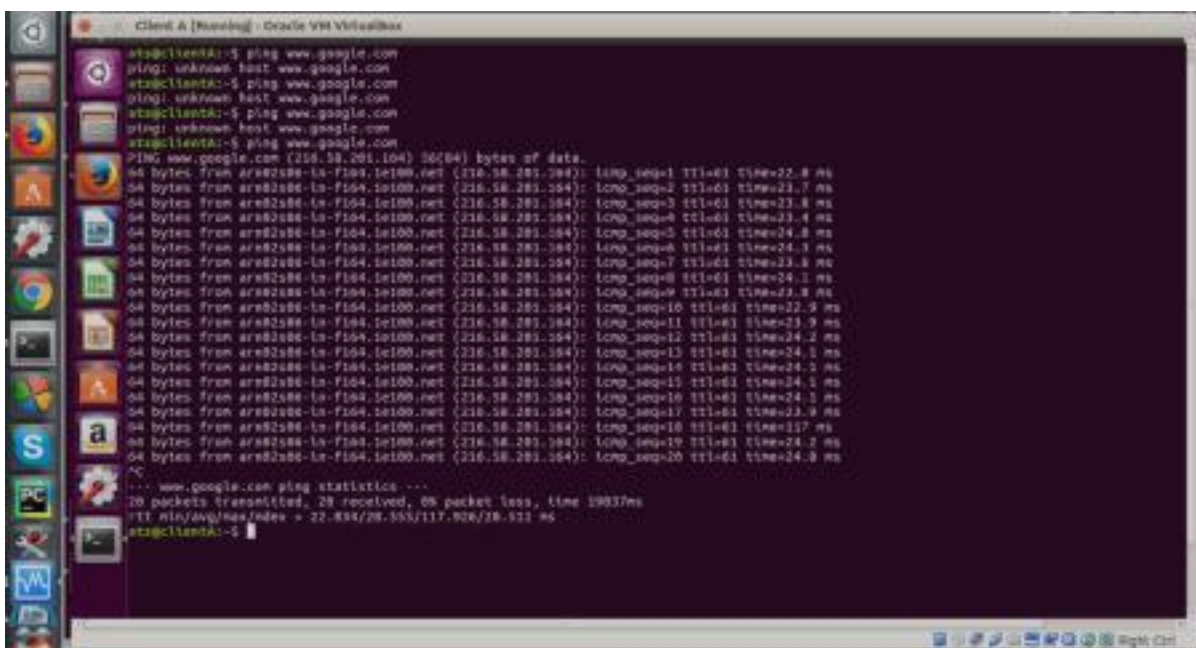
```
$IPT -t filter -A FORWARD -i $HIF -j ACCEPT
$IPT -t filter -A FORWARD -i $NIF -m conntrack --ctstate ESTABLISHED,
RELATED -j ACCEPT
```

After changing these rules the packets are forwarded to the NAT interface. But here the problem is that Client A uses private address (192.168.60.111) and all routers will have a basic default rule to drop packets coming from the private addresses. So we need to tell Server A to use NAT (more specifically Source NAT - SNAT). In order to do this we need to enable the SNAT on Server A.

Task 27:

To fix the problem outlined above you need to tell Server A to do SNAT on the NAT interface. You must add the following iptables rule.

```
$IPT -t nat -A POSTROUTING -j SNAT -o $NIF --to $NIP
```



```
Client A [Running] - Oracle VM VirtualBox
stratoclientA:~$ ping www.google.com
ping: unknown host www.google.com
stratoclientA:~$ ping www.google.com
ping: unknown host www.google.com
stratoclientA:~$ ping www.google.com
ping: unknown host www.google.com
stratoclientA:~$ ping www.google.com
PING www.google.com (216.58.201.104) 32(64) bytes of data:
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=1 ttl=61 time=22.8 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=2 ttl=61 time=23.7 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=3 ttl=61 time=23.8 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=4 ttl=61 time=23.4 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=5 ttl=61 time=24.8 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=6 ttl=61 time=24.3 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=7 ttl=61 time=23.8 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=8 ttl=61 time=24.1 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=9 ttl=61 time=23.8 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=10 ttl=61 time=22.9 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=11 ttl=61 time=23.9 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=12 ttl=61 time=24.2 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=13 ttl=61 time=24.1 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=14 ttl=61 time=24.2 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=15 ttl=61 time=24.1 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=16 ttl=61 time=24.1 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=17 ttl=61 time=23.9 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=18 ttl=61 time=21.7 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=19 ttl=61 time=21.2 ms
64 bytes from ar802000-ls-f104.1e100.net: icmp_seq=20 ttl=61 time=24.8 ms
^C
.... www.google.com ping statistics ....
20 packets transmitted, 20 received, 0% packet loss, time 1983ms
rtt min/avg/max/mdev = 22.834/28.555/117.926/28.521 ms
stratoclientA:~$
```

SCRIPT:

```
#!/bin/sh
```

```
IPT=/sbin/iptables
```

```
# NAT interface
```

```
NIF=enp0s9
```

```
# NAT IP address
```

```
NIP='10.0.98.100'
```

```
# Host-only interface
```

```
HIF=enp0s3
```

```
# Host-only IP address
```

```
HIP='192.168.60.100'
```

```
# DNS nameserver
```

```
NS='10.0.98.3'
```

```
#####33NS='8.8.8.8'
```

```
## Reset the firewall to an empty, but friendly state
```

```
# Flush all chains in FILTER
```

```
table $IPT -t filter -F
```

```
#####$sudo iptables -A INPUT -p tcp --dport 80 -j REJECT
```

```
##$sudo $IPT -t filter -A INPUT -p tcp --dport 80 -j REJECT
```

```
# Delete any user-defined chains in FILTER table
```

```
#$IPT -t filter -X
```

```
# Flush all chains in NAT
table $IPT -t nat -F

# Delete any user-defined chains in NAT
table $IPT -t nat -X

# Flush all chains in MANGLE table
$IPT -t mangle -F

# Delete any user-defined chains in MANGLE table
$IPT -t mangle -X

# Flush all chains in RAW table
$IPT -t raw -F

# Delete any user-defined chains in RAW
table $IPT -t mangle -X


# Default policy is to send to a dropping chain


####$IPT -t filter -P INPUT ACCEPT
#$IPT -t filter -P INPUT ACCEPT

####TASK14##$IPT -t filter -A OUTPUT -s 127.0.0.0/24 -p tcp --dport
80 -j REJECT

#$IPT -t filter -P OUTPUT ACCEPT
#$IPT -t filter -P FORWARD ACCEPT

##TASK15##


####$IPT -t filter -P INPUT DROP ##1

####$IPT -t filter -I INPUT -p icmp -j ACCEPT ##2

####$IPT -t filter -I OUTPUT -p icmp -j ACCEPT ##2
```

```
#####$IPT -t filter -P OUTPUT DROP ##1
```

```
#####$IPT -t filter -P FORWARD DROP ##1
```

```
##TASKS17-20#####
```

```
$IPT -t filter -P INPUT DROP
```

```
$IPT -t filter -I INPUT -p icmp -j ACCEPT ##1
```

```
$IPT -t filter -A INPUT -s 127.0.0.0/24 -p tcp -j ACCEPT ##2
```

```
##$IPT -t filter -A INPUT -s 10.0.98.3 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A INPUT -s 10.0.98.2 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A INPUT -s 10.0.98.100 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A INPUT -s 10.0.98.3 -p tcp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A INPUT -s 10.0.98.2 -p tcp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A INPUT -s 10.0.98.100 -p tcp --dport 53 -j
```

```
ACCEPT $IPT -t filter -A INPUT -s 10.0.98.100 -j ACCEPT ##3
```

```
$IPT -t filter -A INPUT -s 10.0.98.3 -j ACCEPT ##3
```

```
#$IPT -t filter -A INPUT -p tcp -j ACCEPT ##4##20
```

```
$IPT -t filter -A INPUT -p tcp -m conntrack --ctstate  
ESTABLISHED,RELATED -j ACCEPT ##20
```

```
###IPADDRHOST=192.168.0.1
```

```
###UNPRIVPORTS="1025:65535"
```

```
$IPT -t filter -A INPUT -p tcp --dport 22 -j ACCEPT ##21
```

```
#####$IPT -A INPUT -i $HIP -p tcp !--syn --sport 443 -d $IPADDRHOST --  
match multiport --dports $UNPRIVPORTS -j ACCEPT ##21
```

```
$IPT -t filter -A INPUT -p tcp --dport 443 -j ACCEPT ##21
```



```
$IPT -t filter -A INPUT -p icmp -s 192.168.60.111 -j ACCEPT
$IPT -t filter -A OUTPUT -p icmp -s 192.168.60.111 -j ACCEPT##22
```

```
sudo iptables -I INPUT -s 192.168.60.111 -p tcp -m tcp --dport 10022 -j
ACCEPT##23
```

##5

###

```
$IPT -t filter -P OUTPUT DROP
```

```
$IPT -t filter -A OUTPUT -s 127.0.0.0/24 -p tcp -j ACCEPT ##2
```

```
$IPT -t filter -A OUTPUT -p icmp -j ACCEPT ##1
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.3 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.2 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.100 -p udp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.3 -p tcp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.2 -p tcp --dport 53 -j ACCEPT
```

```
#$IPT -t filter -A OUTPUT -s 10.0.98.100 -p tcp --dport 53 -j ACCEPT
```

```
##$IPT -A OUTPUT -p icmp --icmp-type echo-request -j ACCEPT
```

```
##$IPT -A INPUT -p icmp --icmp-type echo-reply -j ACCEPT
```

```
$IPT -t filter -A OUTPUT -s 10.0.98.100 -j ACCEPT ##3 $IPT -t filter
```

```
-A OUTPUT -s 10.0.98.3 -j ACCEPT ##3
```

```
#$IPT -t filter -I OUTPUT -p tcp -j ACCEPT ##4##20
```

#task 20

```
$IPT -t filter -I OUTPUT -p tcp -m conntrack --ctstate NEW,ESTABLISHED
-j ACCEPT
```

```
###$IPT -t filter -I OUTPUT -p tcp --dport 22 -j ACCEPT ##21
```

```
###$IPT -A OUTPUT -o $HIP -p tcp -s $IPADDRHOST -m multiport --  
sports $UNPRIVPORTS --dport 443 -j ACCEPT ##21
```

```
$IPT -t filter -P FORWARD DROP
```

```
$IPT -t filter -A FORWARD -i $HIF -j ACCEPT  
$IPT -t filter -A FORWARD -i $NIF -m conntrack --ctstate ESTABLISHED,  
RELATED -j ACCEPT ###26
```

```
$IPT -t nat -A POSTROUTING -j SNAT -o $NIF --to $NIP ##27
```

```
#### Create logging chains
```

```
$IPT -t filter -N input_log
```

```
$IPT -t filter -N output_log
```

```
$IPT -t filter -N forward_log
```

```
# Set some logging targets for DROPPED packets
```

```
$IPT -t filter -A input_log -j LOG --log-level notice --log-prefix "input  
drop: "
```

```
$IPT -t filter -A output_log -j LOG --log-level notice --log-prefix "output  
drop: "
```

```
$IPT -t filter -A forward_log -j LOG --log-level notice --log-prefix "forward  
drop: "
```

```
echo "Added logging"
```

```
# Return from the logging chain to the built-in
```

```
chain $IPT -t filter -A input_log -j RETURN
```

```
$IPT -t filter -A output_log -j RETURN
```

```
$IPT -t filter -A forward_log -j RETURN
```

```
# These rules must be inserted at the end of the built-in
```

```
# chain to log packets that will be dropped by the default
```

```
# DROP policy
```

```
$IPT -t filter -A INPUT -j input_log
```

```
$IPT -t filter -A OUTPUT -j output_log
```

```
$IPT -t filter -A FORWARD -j forward_log
```

```
#####$ping 127.0.0.1
```

```
echo $input_log
```

```
echo $output_log
```

```
echo $forward_log
```