Assignment 3

s316620

4018NSA

PARTI: DHCP and DNS in Virtual Lab

1. Start up the network that was created in the previous assignments.

2. Exploring the config files of your Quagga router configuration.

a. First configuring the adapter 1, connect it to Internal network “GW_external” and adapter 2 to “GW_SW1_cable”.

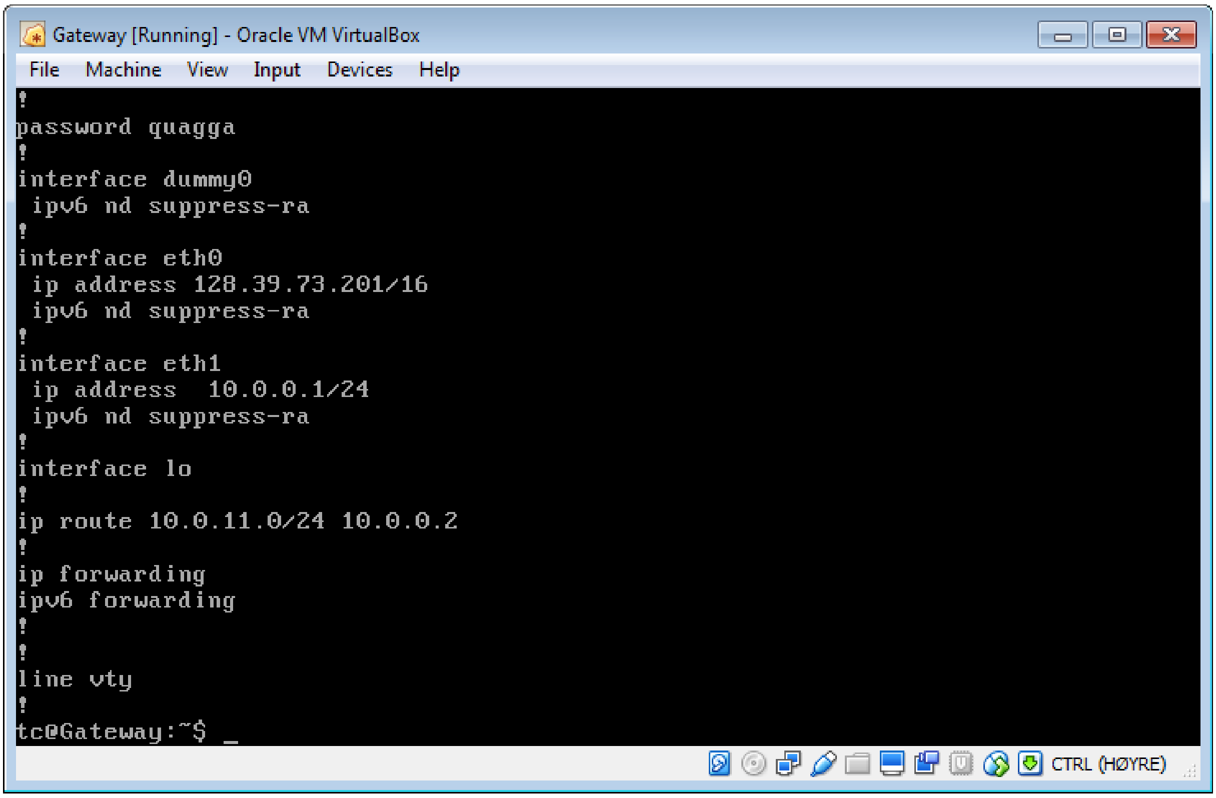
When I set up the quagga interface in Assignment two I removed the ip address configuration line seen below from the “/opt/bootlocal.sh” file.

“ifconfig eth0 10.0.0.1 netmask 255.255.255.0”

b. After changing the internal networks the adapters are connected to, I change the ip address assigned to each interface. This can be done using multiple ways. We can either configure it through the quagga terminal or edit the “*/usr/local/etc/quagga/zebra.conf*” file.

Use one of the following method :

1. Editting the *zebra.conf* file.
   Open the file “*/usr/local/etc/quagga/zebra.conf*” and the edit the IP address to the new IP addresses for each of the interface.


Gateway [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
!
password quagga
!
interface dummy0
 ipv6 nd suppress-ra
!
interface eth0
 ip address 128.39.73.201/16
 ipv6 nd suppress-ra
!
interface eth1
 ip address  10.0.0.1/24
 ipv6 nd suppress-ra
!
interface lo
!
ip route 10.0.11.0/24 10.0.0.2
!
ip forwarding
ipv6 forwarding
!
!
line vty
!
tc@Gateway:~$ _
```

2. Using the quagga terminal :

   Use the commands:

   sudo vtysh   // Open the quagga interface

   config t   // Configure

   interface eth0   // Open interface eth0

   no ip address 10.0.0.1/24   // Remove ip address from the interface

   ip address 128.39.73.201/16 // Add new ip address to the interface

   exit

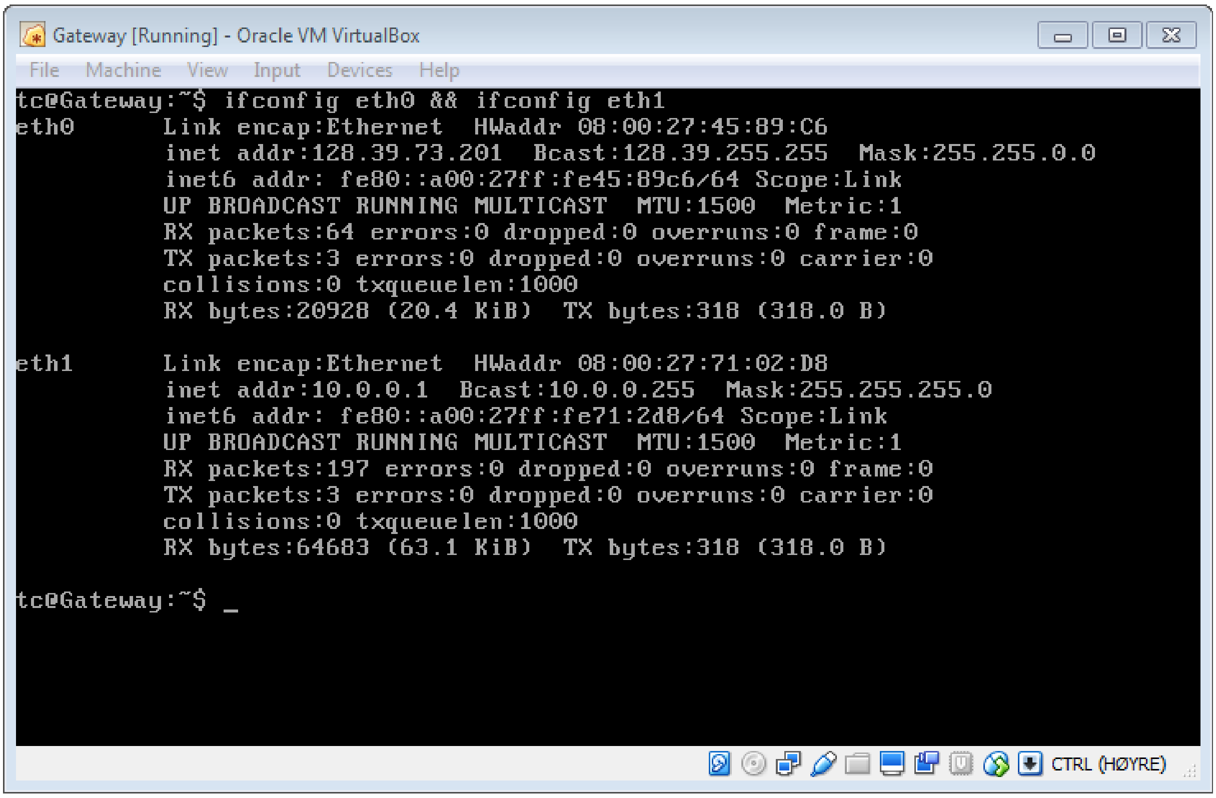
   exit

   write   // Write the changes to /usr/local/etc/quagga/zebra.conf

   exit

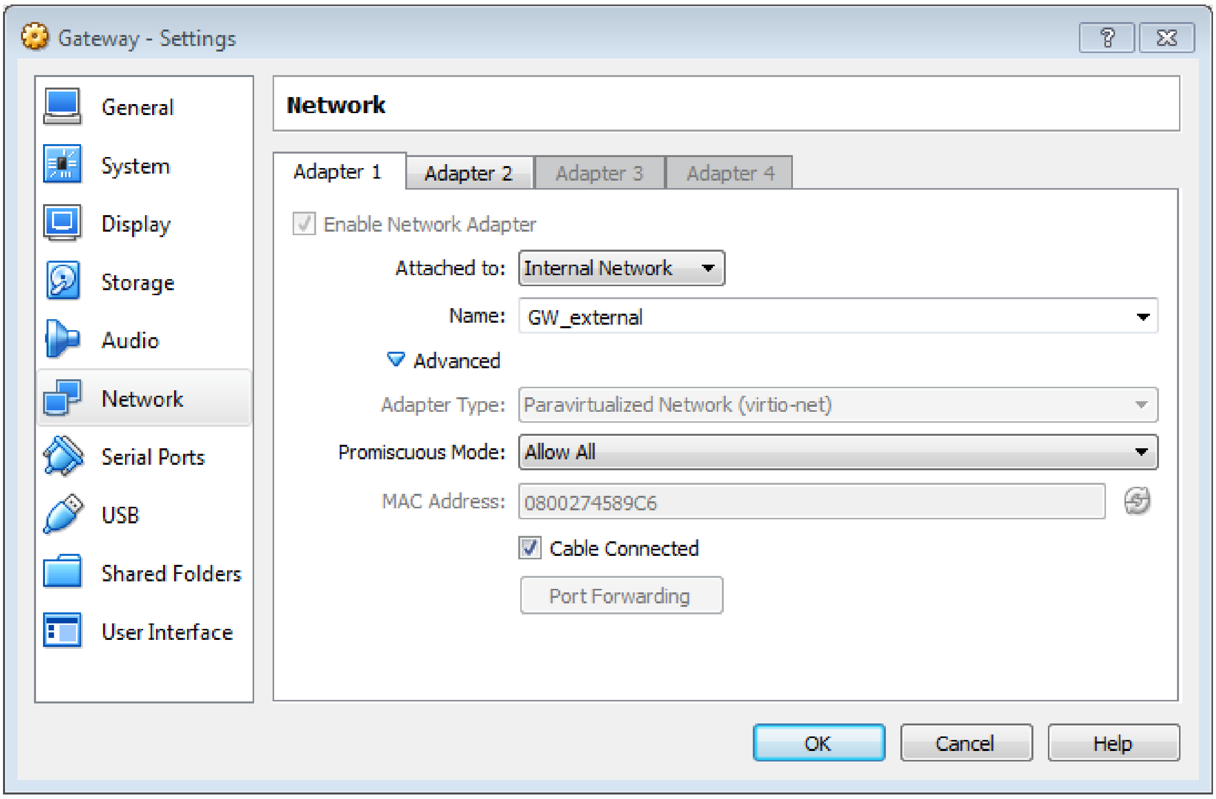
   filetool.sh –b // Save persistently

Do the same for interface eth1 as well, and then reboot. Then we can see that the new IP addresses are assigned to the interfaces.



```
tc@Gateway:~$ ifconfig eth0 && ifconfig eth1
eth0      Link encap:Ethernet  HWaddr 08:00:27:45:89:C6
          inet addr:128.39.73.201  Bcast:128.39.255.255  Mask:255.255.0.0
          inet6 addr: fe80::a00:27ff:fe45:89c6/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:64 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:20928 (20.4 KiB)  TX bytes:318 (318.0 B)

eth1      Link encap:Ethernet  HWaddr 08:00:27:71:02:D8
          inet addr:10.0.0.1  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe71:2d8/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:197 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:64683 (63.1 KiB)  TX bytes:318 (318.0 B)

tc@Gateway:~$ _
```

c. Making sure that Adapter 1 (eth0) is connected to “GW_external” and Adapter 2 (eth1) is connected to “GW_SW1_cable”.



Network

☑ Enable Network Adapter

Attached to: Internal Network

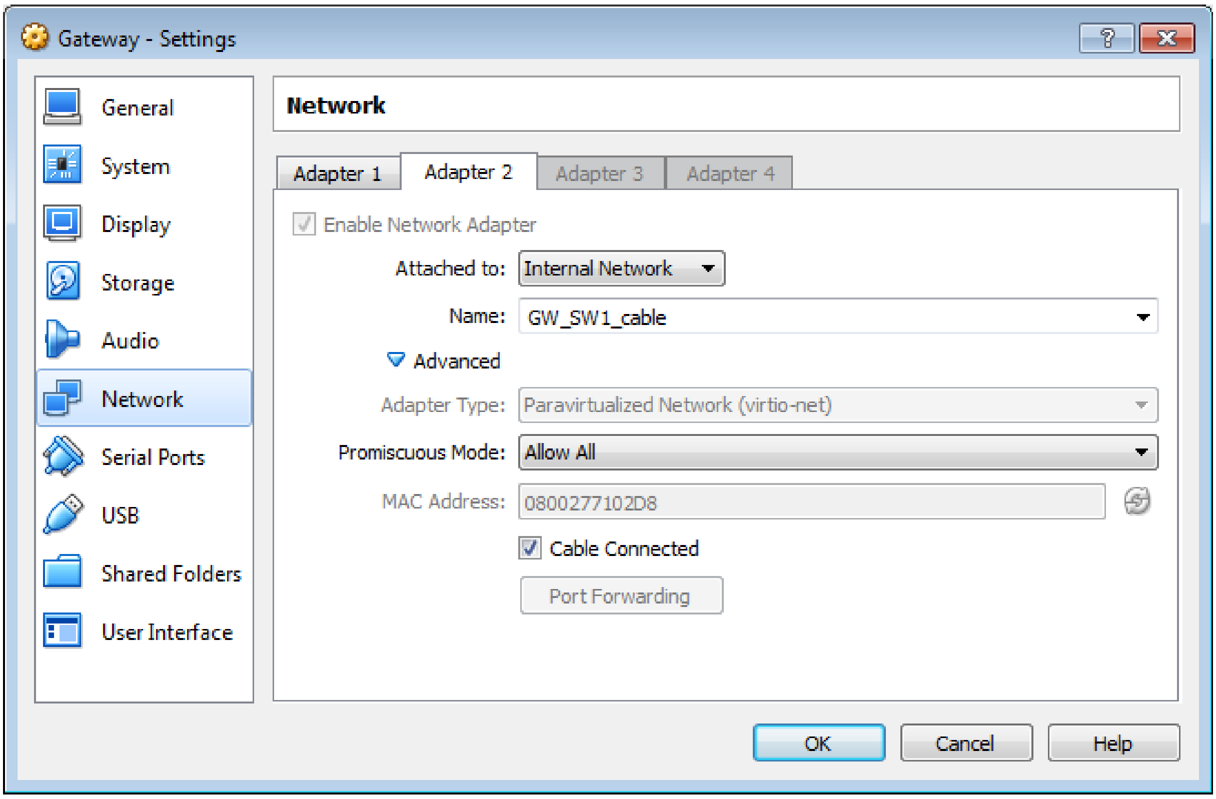
Name: GW_external

Adapter Type: Paravirtualized Network (virtio-net)

Promiscuous Mode: Allow All

MAC Address: 0800274589C6

☑ Cable Connected



Network

☑ Enable Network Adapter

Attached to: Internal Network

Name: GW_SW1_cable

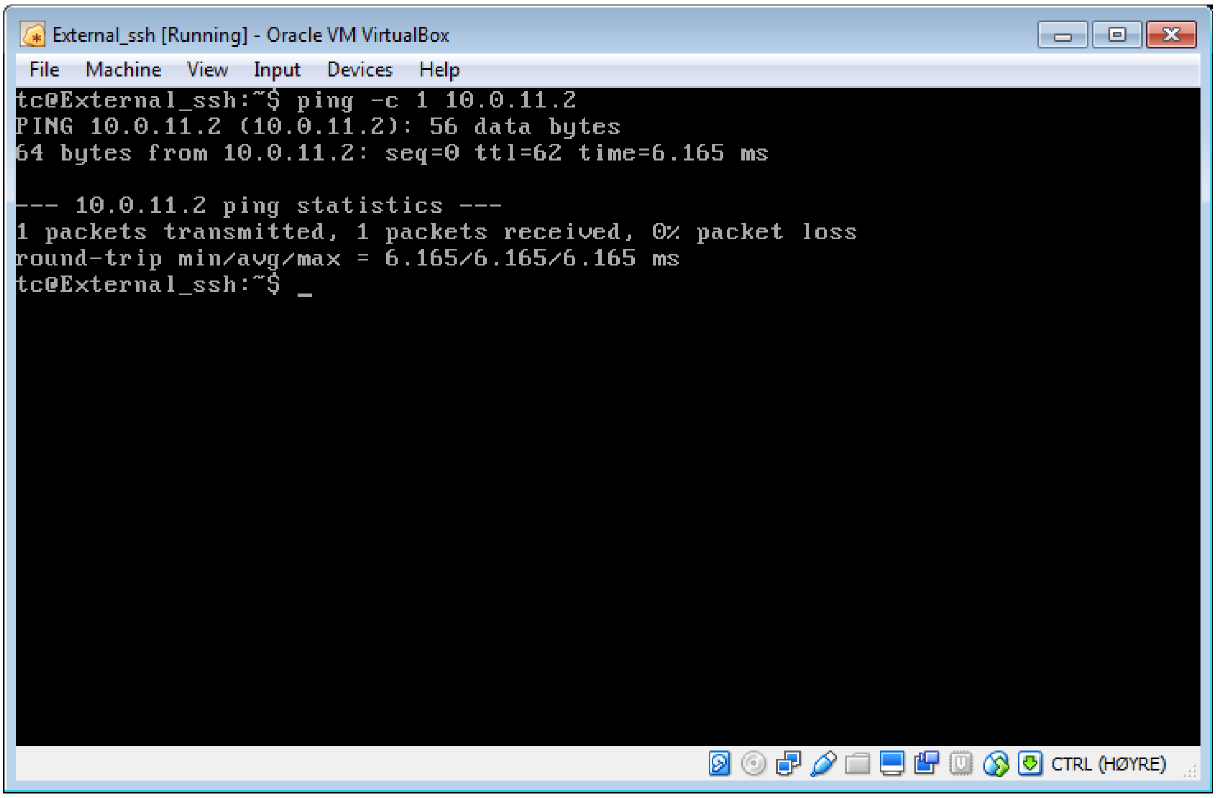
Adapter Type: Paravirtualized Network (virtio-net)

Promiscuous Mode: Allow All

MAC Address: 0800277102D8

☑ Cable Connected

Then checking connectivity with ping from External_ssh to Host 1 and it shows that it is connected.



```
tc@External_ssh:~$ ping -c 1 10.0.11.2
PING 10.0.11.2 (10.0.11.2): 56 data bytes
64 bytes from 10.0.11.2: seq=0 ttl=62 time=6.165 ms

--- 10.0.11.2 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 6.165/6.165/6.165 ms
tc@External_ssh:~$ _
```

3. A minimal DHCP server/daemon (udhcpd)

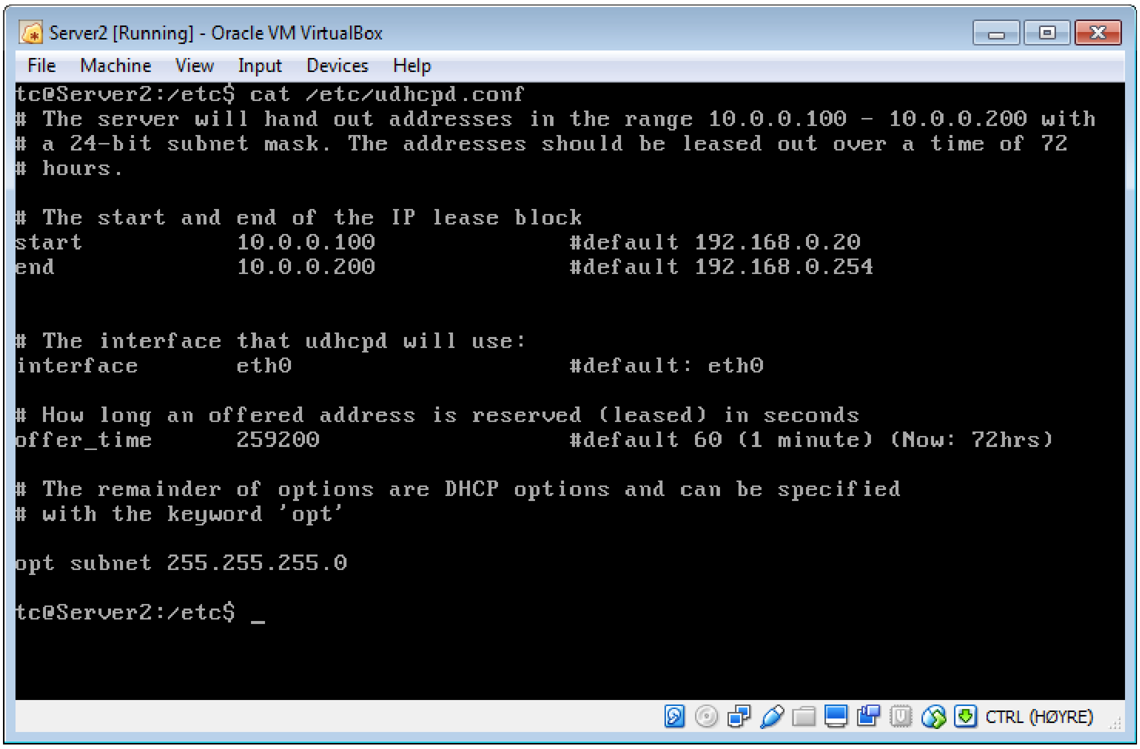
a. Now I am going to install a virtual udhcp-server in the network. However, if there is a udhcp server running in the network, the udhcpc process would quickly obtain a new address from it and would override the manual IP address configuration I set up. So I am turning off the udhcp client processes at the nodes “Gateway”, “Choke” and “Server2” (Where I will set up the udhcp-server).

   I turn off the processes in each of the nodes by adding the command “*pkill udhcp”* in the *bootlocal.sh* file, before the assignment of IP addresses (*ifconfig* lines). [*Note: Remember to save persistently (filetool.sh –b) all the changes that is made.*]

b. Creating a persistent file *“/etc/udhcpd.conf”* to Server2, for the configuration of DCHP-server.

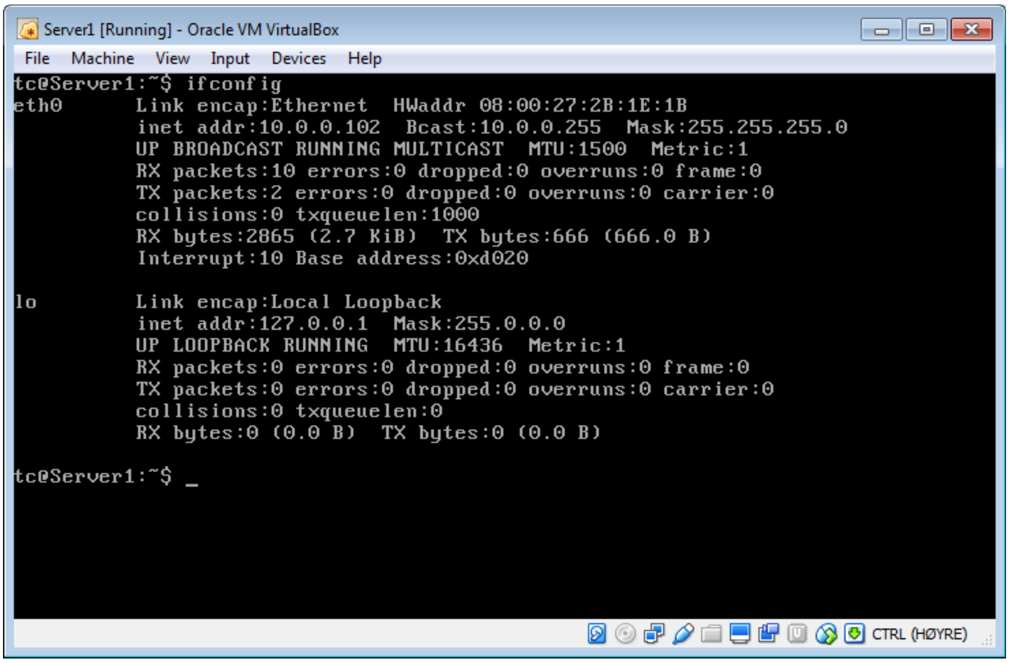
We get a default "*udhcpd.conf*" file from the internet (https://udhcp.busybox.net/udhcpd.conf). And use the format to configure the *udhcpd.conf* file in */etc/* .

My *udhcpd.conf* file looks like :


Server2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Server2:/etc$ cat /etc/udhcpd.conf
# The server will hand out addresses in the range 10.0.0.100 - 10.0.0.200 with
# a 24-bit subnet mask. The addresses should be leased out over a time of 72
# hours.

# The start and end of the IP lease block
start           10.0.0.100              #default 192.168.0.20
end             10.0.0.200              #default 192.168.0.254


# The interface that udhcpd will use:
interface       eth0                    #default: eth0

# How long an offered address is reserved (leased) in seconds
offer_time      259200                  #default 60 (1 minute) (Now: 72hrs)

# The remainder of options are DHCP options and can be specified
# with the keyword 'opt'

opt subnet 255.255.255.0

tc@Server2:/etc$ _
```

c. Now starting the udhcpd-daemon at Server2.
I use the command " *sudo udhcpd /etc/udhcpd.conf &"*
Then restart "Gateway", "Choke" and "Server1".
Gateway has the IP address 10.0.0.1 on eth0 and 128.39.73.201 on eth1.
Choke has the IP address 10.0.0.2 on eth0 and 10.0.11.1 on eth1.
Whereas "Server1" has the IP address 10.0.0.102 on eth0.

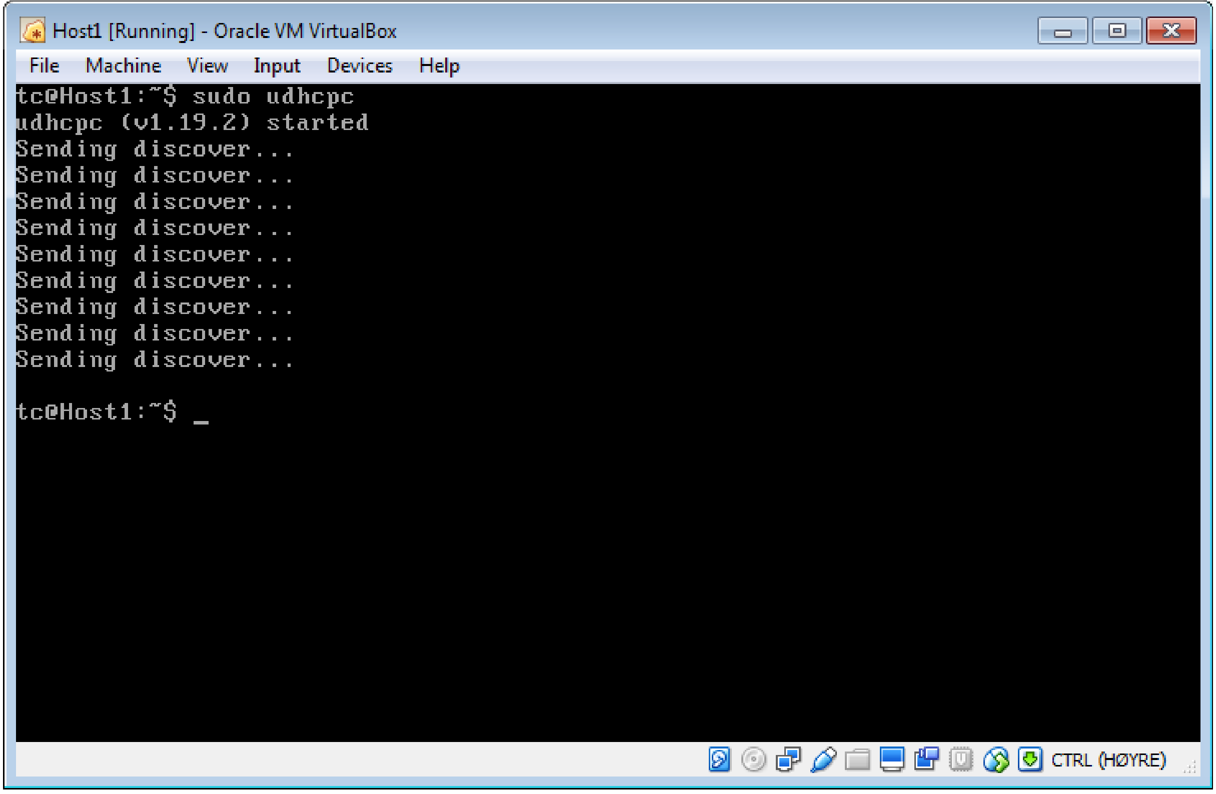


Server1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Server1:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2B:1E:1B
          inet addr:10.0.0.102  Bcast:10.0.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:10 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:2865 (2.7 KiB)  TX bytes:666 (666.0 B)
          Interrupt:10 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tc@Server1:~$ _
```

The reason “Server1” gets an IP address in the range 10.0.0.100 – 10.0.0.200 as configured in “Server2” (DHCP-server) is because when “Server1” reboots the udhcp process runs in Server1 and gets the IP address from “Server2”.

d. Running “*sudo udhcpc”* at Host1, to check what is happening.



Host1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Host1:~$ sudo udhcpc
udhcpc (v1.19.2) started
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending discover...
Sending discover...

tc@Host1:~$ _
```

Host1 does not discover any IP address, here Host1 is the dhcp-client and “Server2” is the dhcp-server. There is no DHCP-relay agent between the server and the client so the client does not get configured.

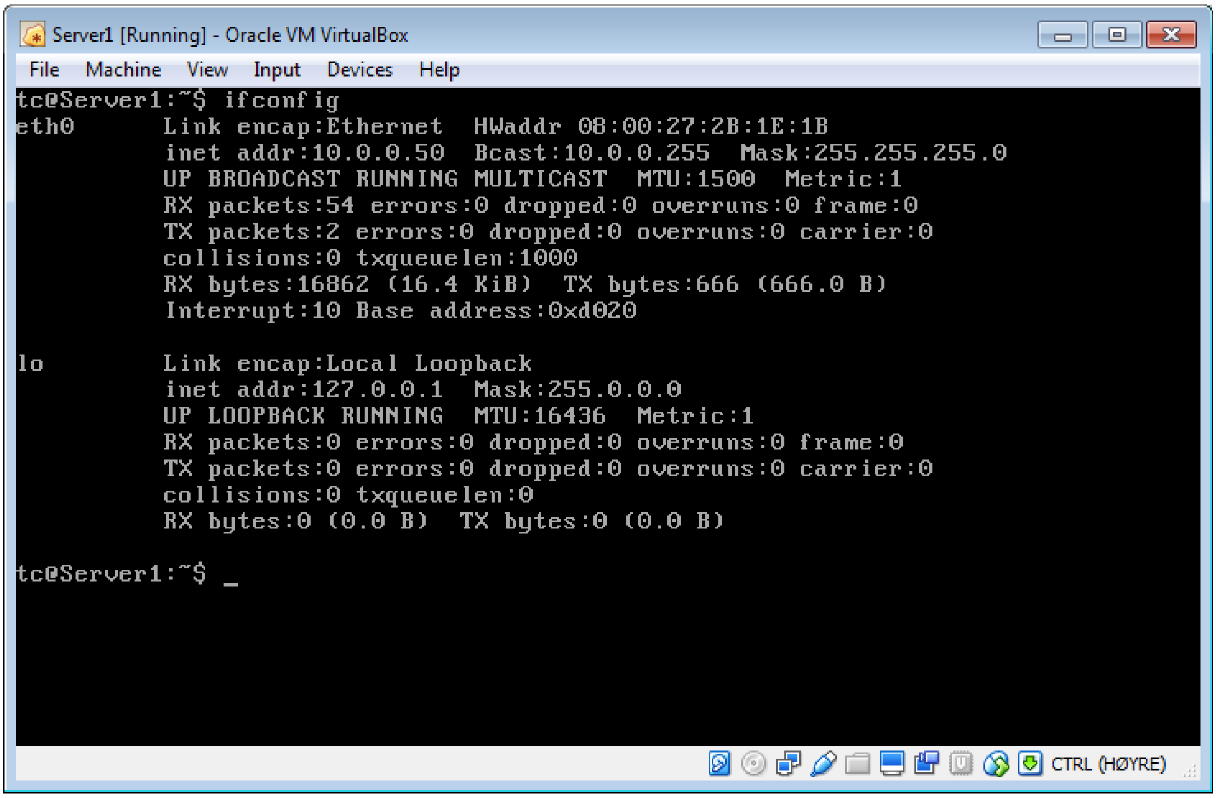
e. DHCP-Relay
The purpose of the DHCP- relay is to pass packets containing information, from DHCP-clients to the DHCP-server and vice-versa. The client requests are forwarded to the server and then the server can configure the IP-address and lease it to the clients. The DHCP relay agent is relied upon when the DHCP-server and DHCP-client are not on the same subnet. In the assignment above, “Host1” (DHCP-client) is on a different subnet than “Server1” (DHCP-server). Without a DHCP-relay agent, which would be “Choke” in this case, the “Host1” is not assigned an IP address. In “Choke” the *udhcp* process has been terminated so this is the region that “Host1” does not have an IP.

f. To assign a static IP address to “Server1” we will run the *ifconfig eth0* command on “Server1”, to get the HW address of “Server1”. Then add the line to the *udhcpd.conf* file in “Server2” :

static_lease <Server1 MAC address>

Where <Server1 MAC address> is the actual MAC address of “Server1”. The MAC address of the interface *eth0* is referred to as “HWaddr”. After adding the ‘*static_lease’* to the udhcpd.conf file save persistently.

Now to check run the *udhcpd* process with the new *udhcpd.conf* file and then restart “Server1”.



Server1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Server1:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:2B:1E:1B
          inet addr:10.0.0.50  Bcast:10.0.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:54 errors:0 dropped:0 overruns:0 frame:0
          TX packets:2 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:16862 (16.4 KiB)  TX bytes:666 (666.0 B)
          Interrupt:10 Base address:0xd020

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tc@Server1:~$ _
```

[NOTE: The “sudo udhcpd /etc/udhcpd.conf &” command is not added to bootlocal.sh as instructed in the assignment. Here we are just checking if what we did worked.]

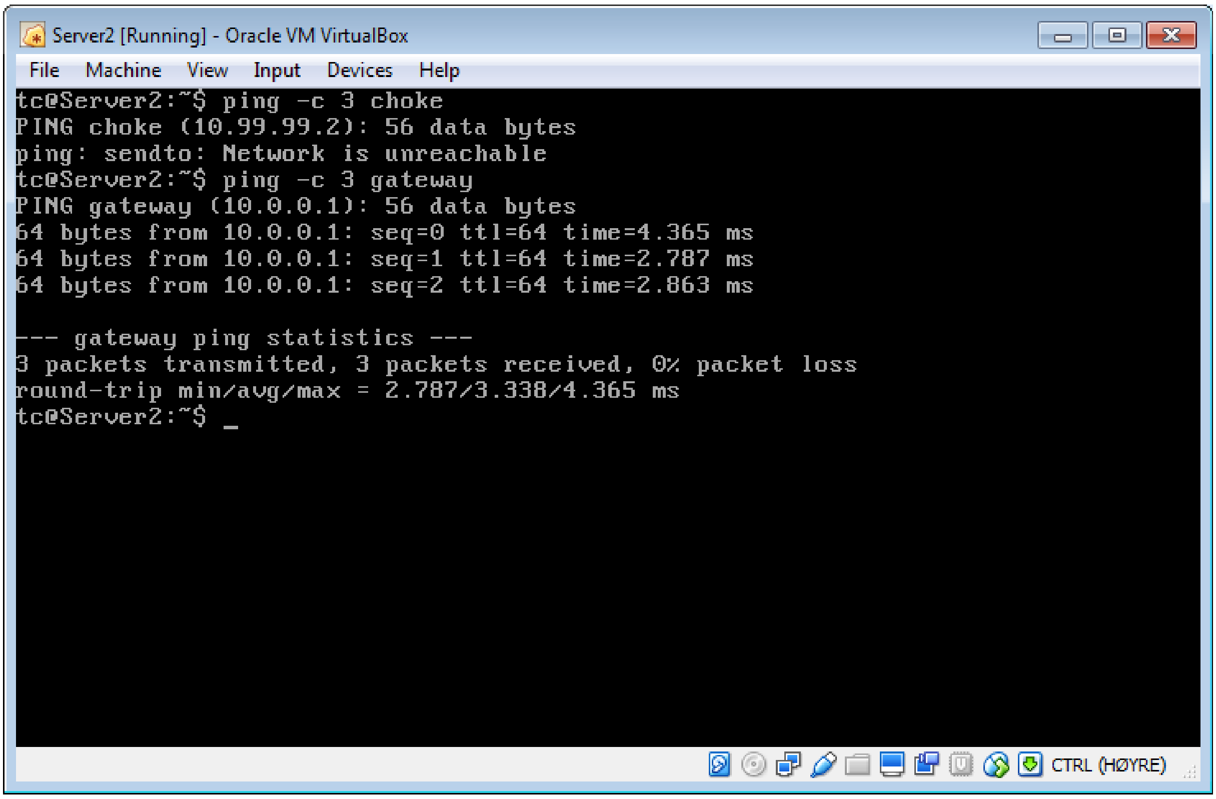
4. The /etc/hosts file.

   a. At Server2, issuing the following command "*ping gateway*". We get the error message :
      ping : bad address 'gateway'

   b. Updating the */etc/hosts* – file at Server2, by adding the two lines :
      i. *10.0.0.1 gateway*
      ii. *10.99.99.2 choke* (We intentionally have set an incorrect address for testing purpose)

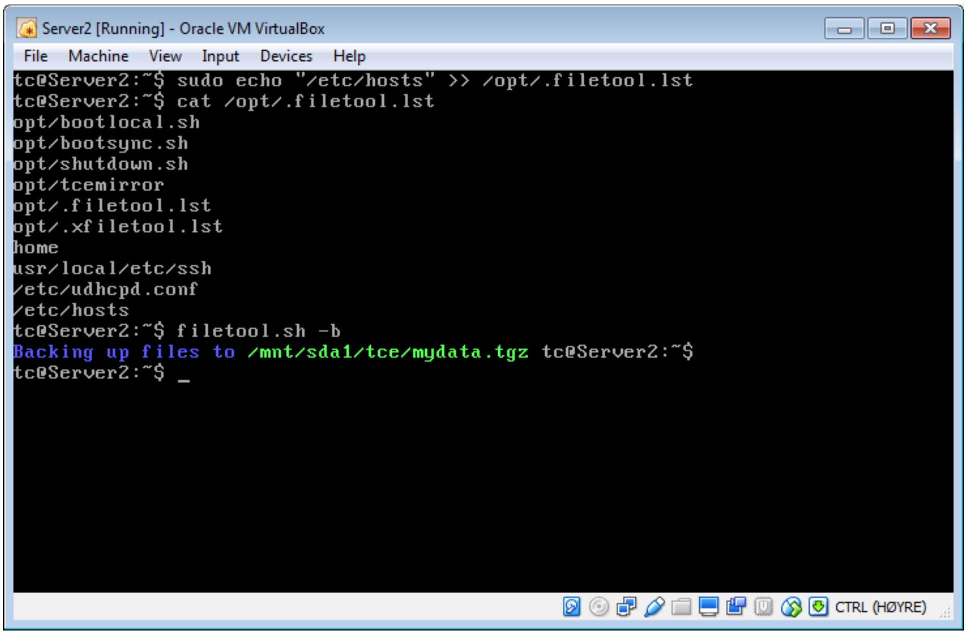
      Now we issue the commands "*ping choke*" and "*ping gateway*" on Server2.

      It can be seen that Server2 cannot "*ping*" Choke, an error message is returned as seen in the image. But now it is able to ping gateway by issuing the command "*ping gateway*". This is because we assigned the correct IP address of Gateway but not of Choke in the *"/etc/hosts"*-file.



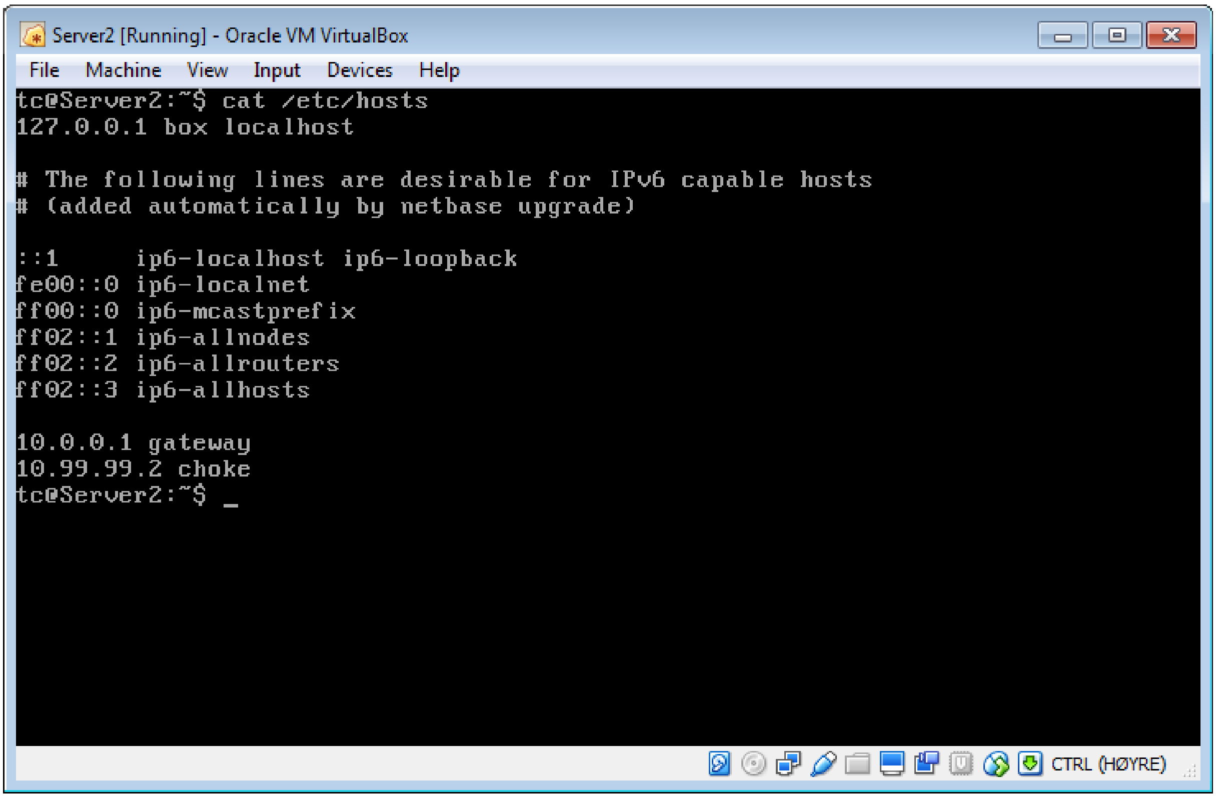
```
tc@Server2:~$ ping -c 3 choke
PING choke (10.99.99.2): 56 data bytes
ping: sendto: Network is unreachable
tc@Server2:~$ ping -c 3 gateway
PING gateway (10.0.0.1): 56 data bytes
64 bytes from 10.0.0.1: seq=0 ttl=64 time=4.365 ms
64 bytes from 10.0.0.1: seq=1 ttl=64 time=2.787 ms
64 bytes from 10.0.0.1: seq=2 ttl=64 time=2.863 ms

--- gateway ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 2.787/3.338/4.365 ms
tc@Server2:~$ _
```

   c. The change in the *"/etc/hosts"* file is made persistent by adding the file name to the "*/opt/.filetool.lst*". Then running the command *"filetool.sh –b"*, which saves the changes in the *hosts* file persistently.


Server2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Server2:~$ sudo echo "/etc/hosts" >> /opt/.filetool.lst
tc@Server2:~$ cat /opt/.filetool.lst
opt/bootlocal.sh
opt/bootsync.sh
opt/shutdown.sh
opt/tcemirror
opt/.filetool.lst
opt/.xfiletool.lst
home
usr/local/etc/ssh
/etc/udhcpd.conf
/etc/hosts
tc@Server2:~$ filetool.sh -b
Backing up files to /mnt/sda1/tce/mydata.tgz tc@Server2:~$
tc@Server2:~$ _
```

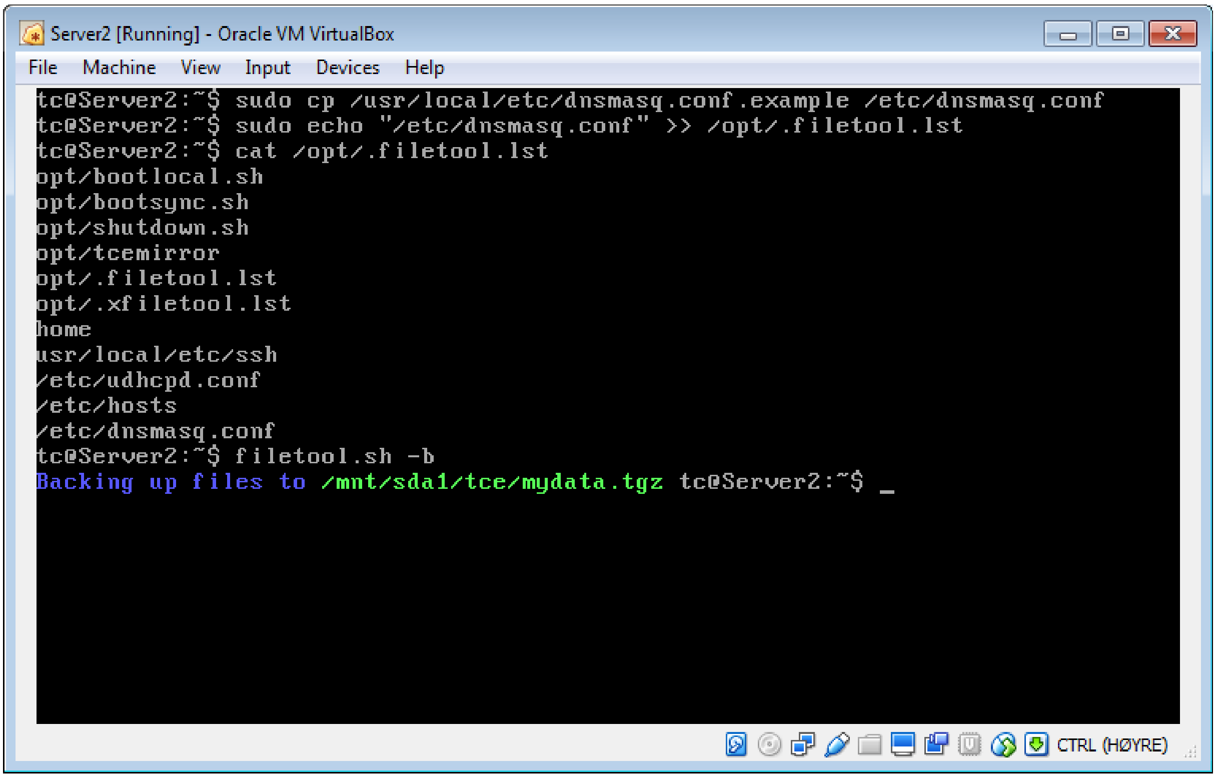
d. The line we added in the above assignment is overwritten by a process after reboot, but before *bootlocal.sh* is run. To fix this problem, I add the line “*echo 10.0.0.1 gateway >> /etc/hosts*”, “ *echo 10.99.99.2 choke >> /etc/hosts*” in the *bootlocal.sh* – file on Server2.
Then save the changes persistently and reboot the host. Check that the Gateway can be pinged from Server2 with the command “*ping gateway*” after reboot, and that the */etc/hosts* – file is OK.


Server2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
CTRL (HØYRE)

```
tc@Server2:~$ cat /etc/hosts
127.0.0.1 box localhost

# The following lines are desirable for IPv6 capable hosts
# (added automatically by netbase upgrade)

::1      ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts

10.0.0.1 gateway
10.99.99.2 choke
tc@Server2:~$ _
```

5. Using dnsmasq for local DNS

a. Installing dnsmasq (Using the command *"ab dnsmasq.tcz"*) at Server 2. To accomplish this, Server2 has to be connected to the internet. The adapter of Server2 should be connected to NAT, and the hosts should be started. We have the "*pkill udhcp"* command running in the *bootlocal.sh* so we have to start the *udhcpc* process first. This can be done by issuing the command "*sudo udhcpc"*. Then when an internet connection is established download and install "dnsmasq.tcz" packet using *ab* command.
Then copy the configuration file example from "*/usr/local/etc/dnsmasq.conf.example"* to *"etc/dnsmasq.conf"*, then save it persistently.



```
tc@Server2:~$ sudo cp /usr/local/etc/dnsmasq.conf.example /etc/dnsmasq.conf
tc@Server2:~$ sudo echo "/etc/dnsmasq.conf" >> /opt/.filetool.lst
tc@Server2:~$ cat /opt/.filetool.lst
opt/bootlocal.sh
opt/bootsync.sh
opt/shutdown.sh
opt/tcemirror
opt/.filetool.lst
opt/.xfiletool.lst
home
usr/local/etc/ssh
/etc/udhcpd.conf
/etc/hosts
/etc/dnsmasq.conf
tc@Server2:~$ filetool.sh -b
Backing up files to /mnt/sda1/tce/mydata.tgz tc@Server2:~$ _
```

b. Now after the installation of the dnsmasq and then having a default configuration file at "*/etc/dnsmasq.conf"*. We configure the following things in *dnsmasq.conf.*
   i. Enable "domain-needed" and "bogus-priv" (probably line 14 – 16)
   ii. Set "dhcp-range =10.0.0.3, 10.0.0.100, 12h" (line 136)
   iii. Set "dhcp-leasefile=/tmp/dnsmasq.lease" (line 413)
   iv. Set "dhcp-option=option:router 10.0.0.1" (line 250)
   v. Configure static routers(gateway and choke) by adding two line in the conf-script (line 195)
      - "dhcp-host=<Gateway HWaddr (eth1)>,ignore"
      - "dhcp-host=<Choke HWaddr (eth0)>,ignore"

Then start the dnsmasq in default mode by running the command :

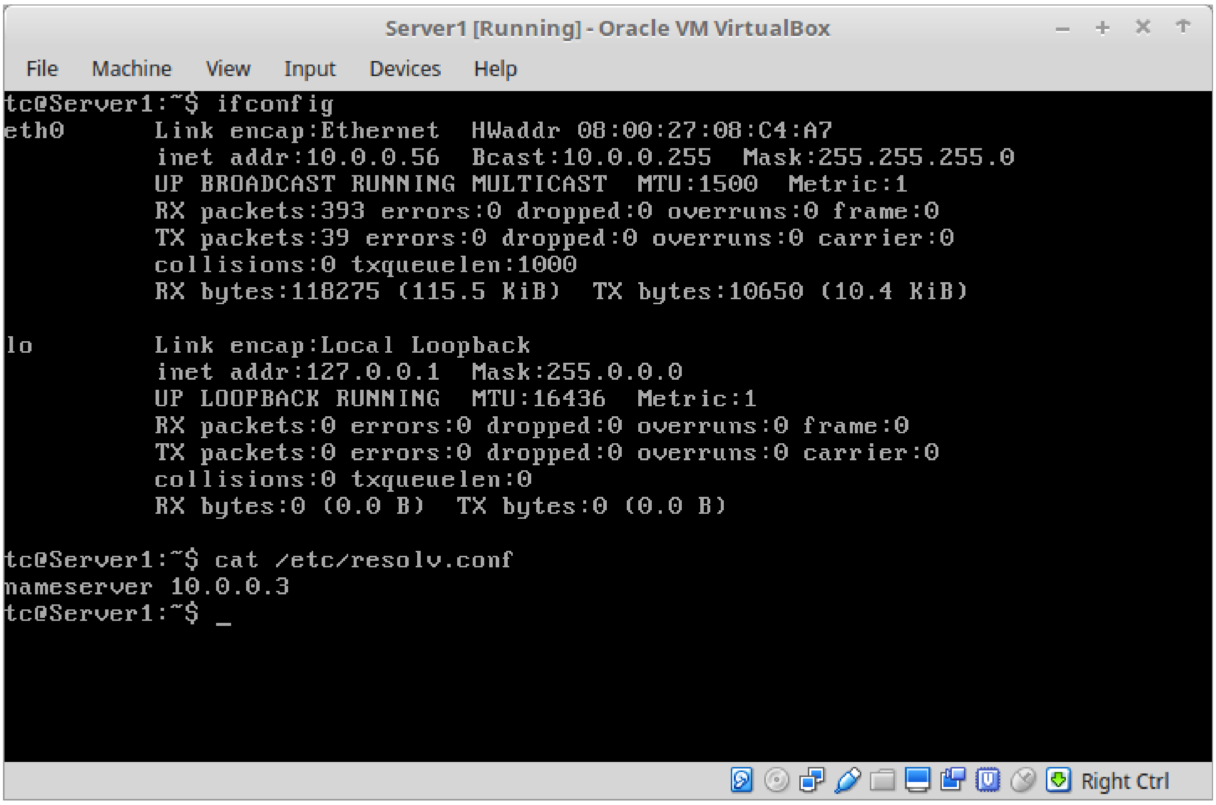
Sudo dnsmasq –d

Then to save the command persistently we added the command to *bootlocal.sh. Using the command :*

sudo echo “dnsmasq” >> /opt/bootlocal.sh

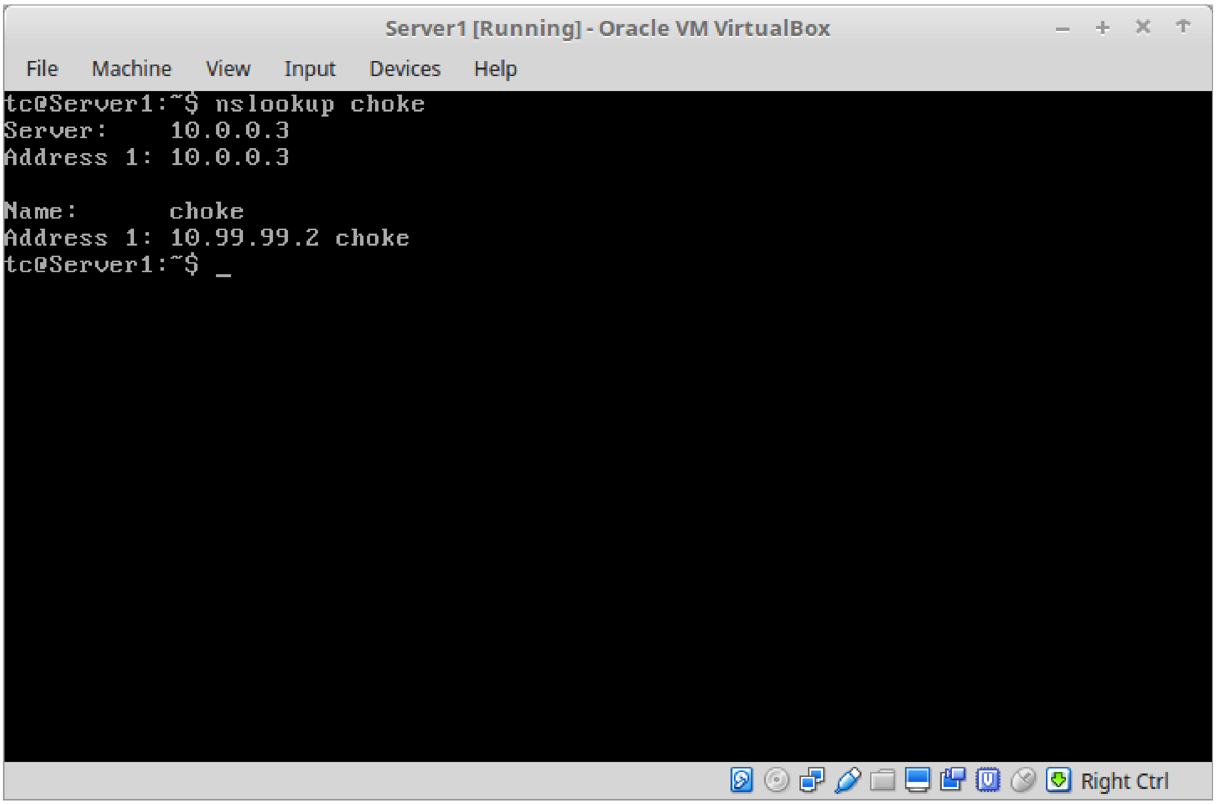
and then save persistently using *filetool.sh*. This runs the dnsmasq after every reboot of the host.

c. Running *udhcpc* at Server1 we get the IP address 10.0.0.56 .The local dns-server has the IP address 10.0.0.3



```
tc@Server1:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:08:C4:A7
          inet addr:10.0.0.56  Bcast:10.0.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:393 errors:0 dropped:0 overruns:0 frame:0
          TX packets:39 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:118275 (115.5 KiB)  TX bytes:10650 (10.4 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

tc@Server1:~$ cat /etc/resolv.conf
nameserver 10.0.0.3
tc@Server1:~$ _
```

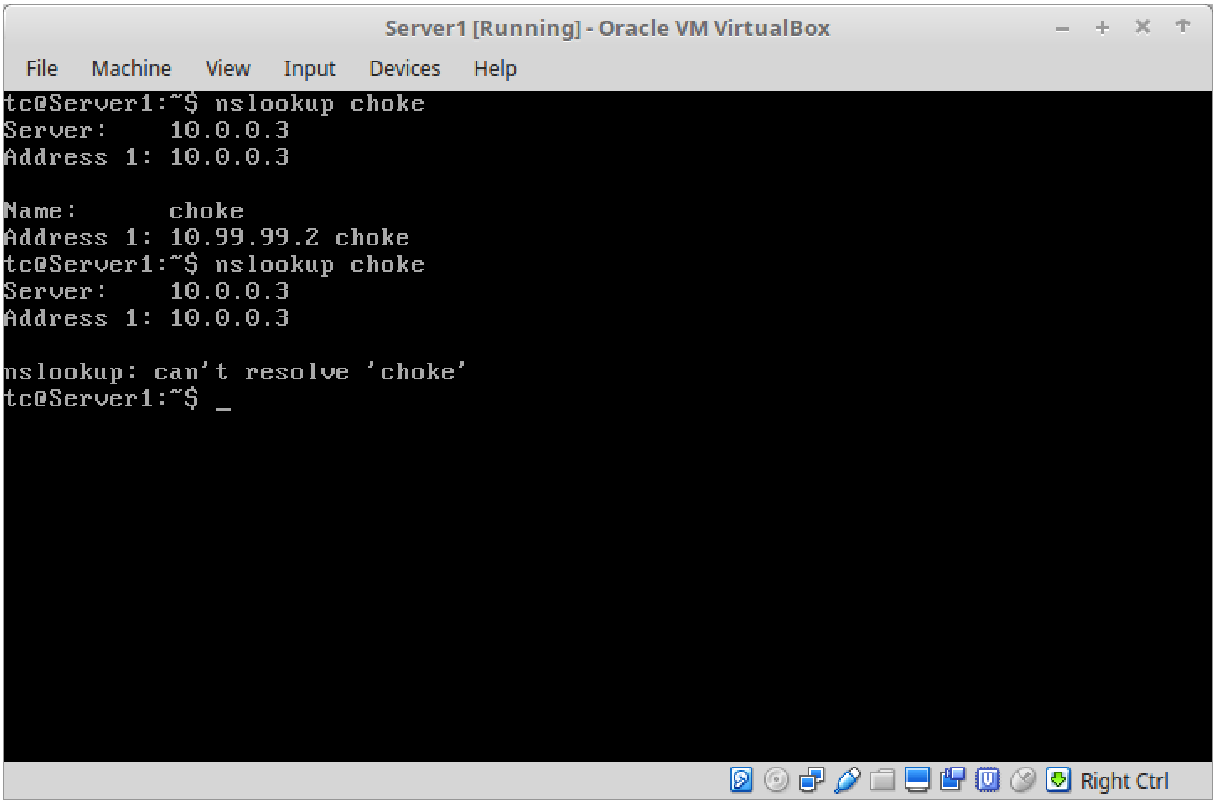
d. Now when running the command *nslookup choke* from Server1. It can be observed that the IP address 10.99.99.2 ip associated with Choke.



```
tc@Server1:~$ nslookup choke
Server:     10.0.0.3
Address 1: 10.0.0.3

Name:       choke
Address 1: 10.99.99.2 choke
tc@Server1:~$ _
```

The nameserver (Server2) has the IP address 10.99.99.2 associated with the name choke in the *"/etc/hosts"* file. As Server2 (10.0.0.3) is the nameserver it looks up its hosts names and assigns the name to the network.

e. If I turn off the dnsmasq at Server2 and then run nslookup we get the message “can’t resolve ‘choke’”. This is because Server1 has Server2 as the nameserver



```
tc@Server1:~$ nslookup choke
Server:    10.0.0.3
Address 1: 10.0.0.3

Name:      choke
Address 1: 10.99.99.2 choke
tc@Server1:~$ nslookup choke
Server:    10.0.0.3
Address 1: 10.0.0.3

nslookup: can't resolve 'choke'
tc@Server1:~$ _
```

where it gets its IP address from, but as the dnsmasq is down Server2 does not provide any IP addresses. Thus we get the above error message.

f. Manually configuring the Gateway and Host to use Server2 as a local dns server, by adding the line :

“echo nameserver 10.0.0.3 >> /etc/resolv.con”

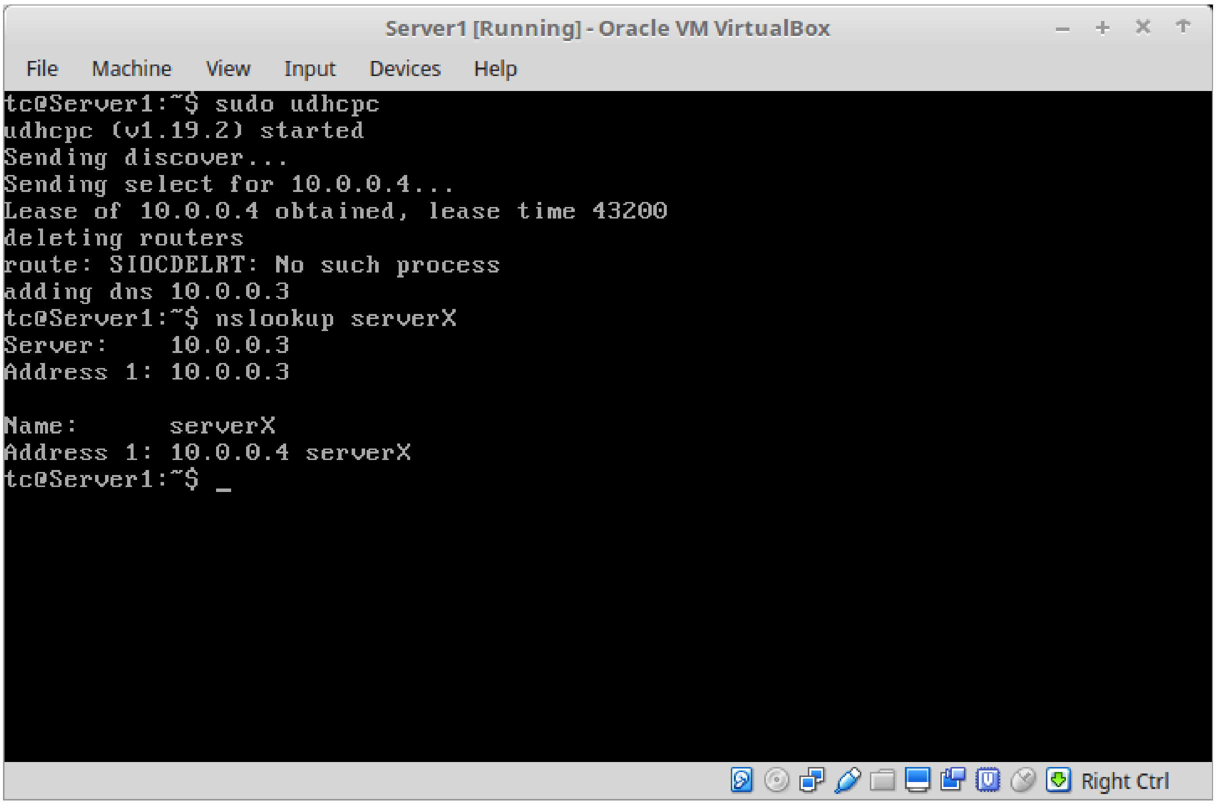
in the *bootlocal.sh* file at Gateway, Choke and Host1.

g. Now, configuring Server1 so that it gets a fixed address by adding the line:

“dhcp-host=<Server1 MAC address>,serverX,10.0.0.4,12h” (line 164)

where the <Server1 MAC address> is the actual Hwaddr of the adapter (eth0) at Server1. This line is added in the *“/etc/dnsmasq.conf”* -file at Server2.

Then running *udhcpc* at server1.



```
tc@Server1:~$ sudo udhcpc
udhcpc (v1.19.2) started
Sending discover...
Sending select for 10.0.0.4...
Lease of 10.0.0.4 obtained, lease time 43200
deleting routers
route: SIOCDELRT: No such process
adding dns 10.0.0.3
tc@Server1:~$ nslookup serverX
Server:    10.0.0.3
Address 1: 10.0.0.3

Name:      serverX
Address 1: 10.0.0.4 serverX
tc@Server1:~$ _
```

Running the udhcpc command and nslookup from Server1 (show above) and Server2 works.

h. Now configuring dnsmasq for all the nodes on the subnet 10.0.0.0/24.

The *dnsmasq.conf* file is edited and Server1 is assigned 10.0.0.4 by changing the line to :

“dhcp-host=<Server1 MAC address>,server1,10.0.0.4,12h” (line 164)

The correct IP address is used for choke in */etc/hosts* at Server2 and Server1, Server2, External_ssh and Host1 is also added to the file.

6. Obtaining Internet access from the virtual network

a. Setting up eth0 of Gateway to connect to the internet. This is done by :

I. Powering off gateway

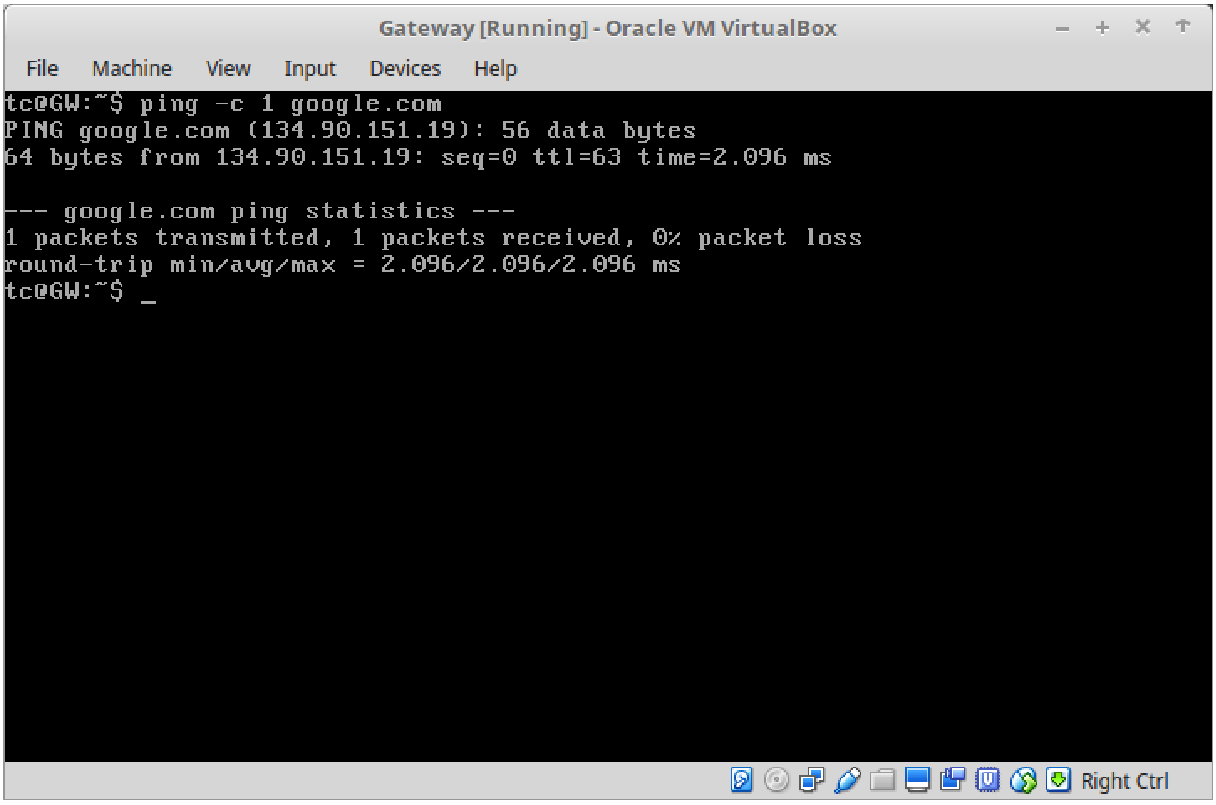
ii. Assign an adapter (Adapter 3) configured to a NAT interface of VirtualBox.

Iii. Disable the Adapter1. Now the VirtualBox assigns Adapter3 as eth0.

The line :

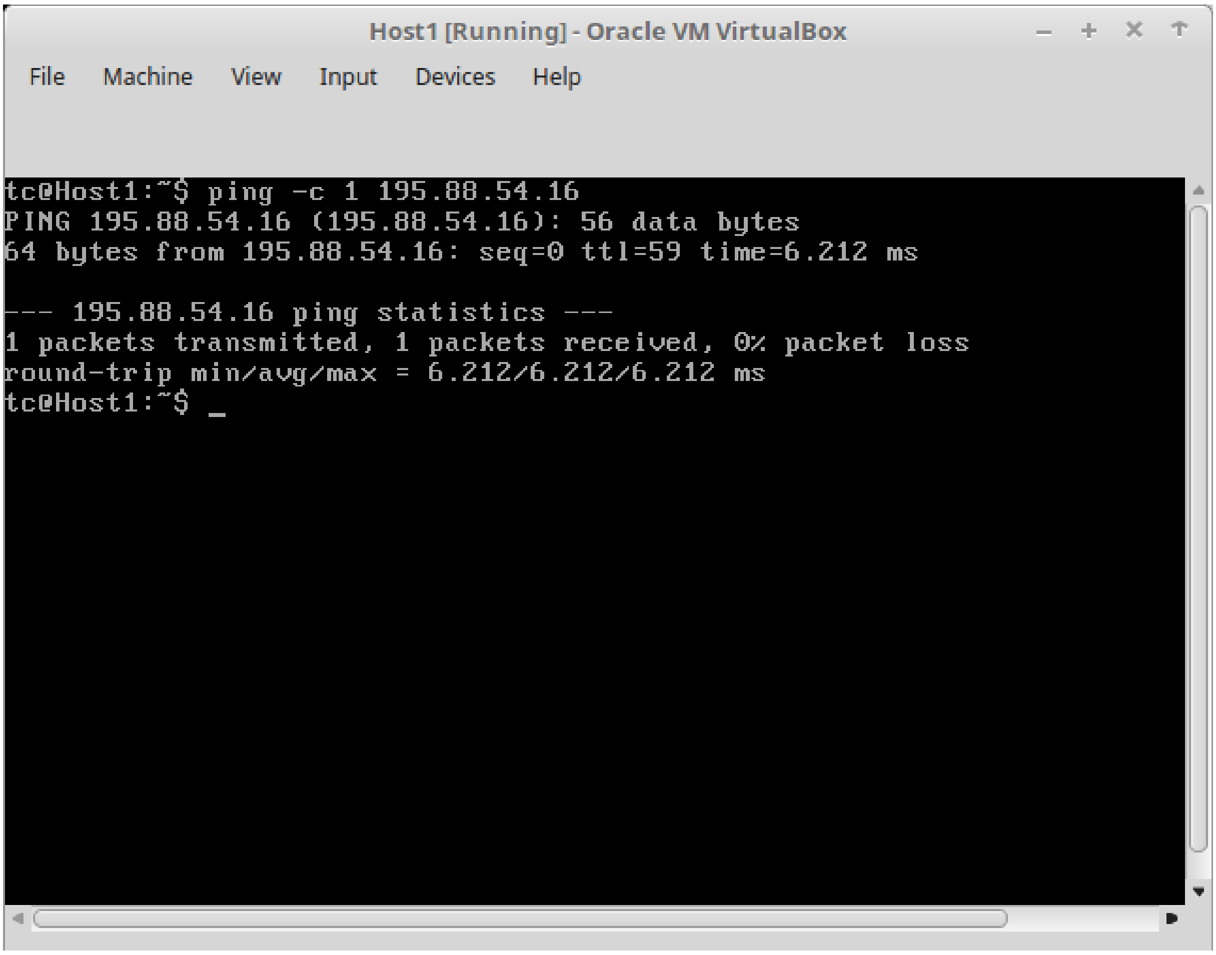
sysctl -w net.ipv4.conf.eth0.proxy_arp=1

is added to the *bootlocal.sh* file. Then we *ping* google.com from Gateway and check that we have internet connection. As seen below, it was sucessful.



```
tc@GW:~$ ping -c 1 google.com
PING google.com (134.90.151.19): 56 data bytes
64 bytes from 134.90.151.19: seq=0 ttl=63 time=2.096 ms

--- google.com ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.096/2.096/2.096 ms
tc@GW:~$ _
```

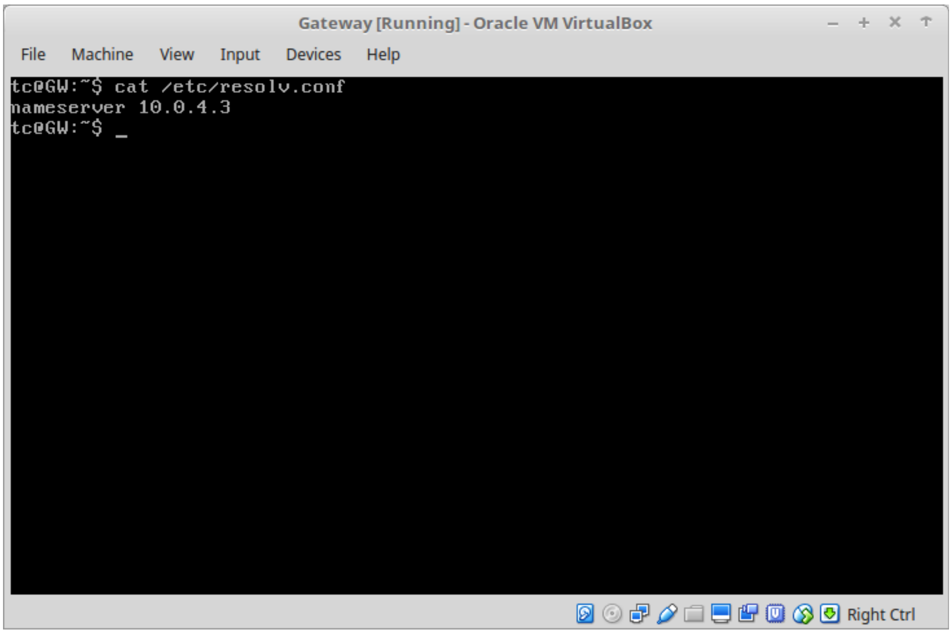
b. Connecting to the internet from Host1. I ping the IP address of *vg.no* (195.88.54.16). Host1 can ping the IP address but not the domain name as the dns server is Server2(10.0.0.3) which does not have the internet nameserver.



```
tc@Host1:~$ ping -c 1 195.88.54.16
PING 195.88.54.16 (195.88.54.16): 56 data bytes
64 bytes from 195.88.54.16: seq=0 ttl=59 time=6.212 ms

--- 195.88.54.16 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 6.212/6.212/6.212 ms
tc@Host1:~$ _
```

c. Now looking for the nameserver at *Gateway* in the *"/etc/resolv.conf"* file.

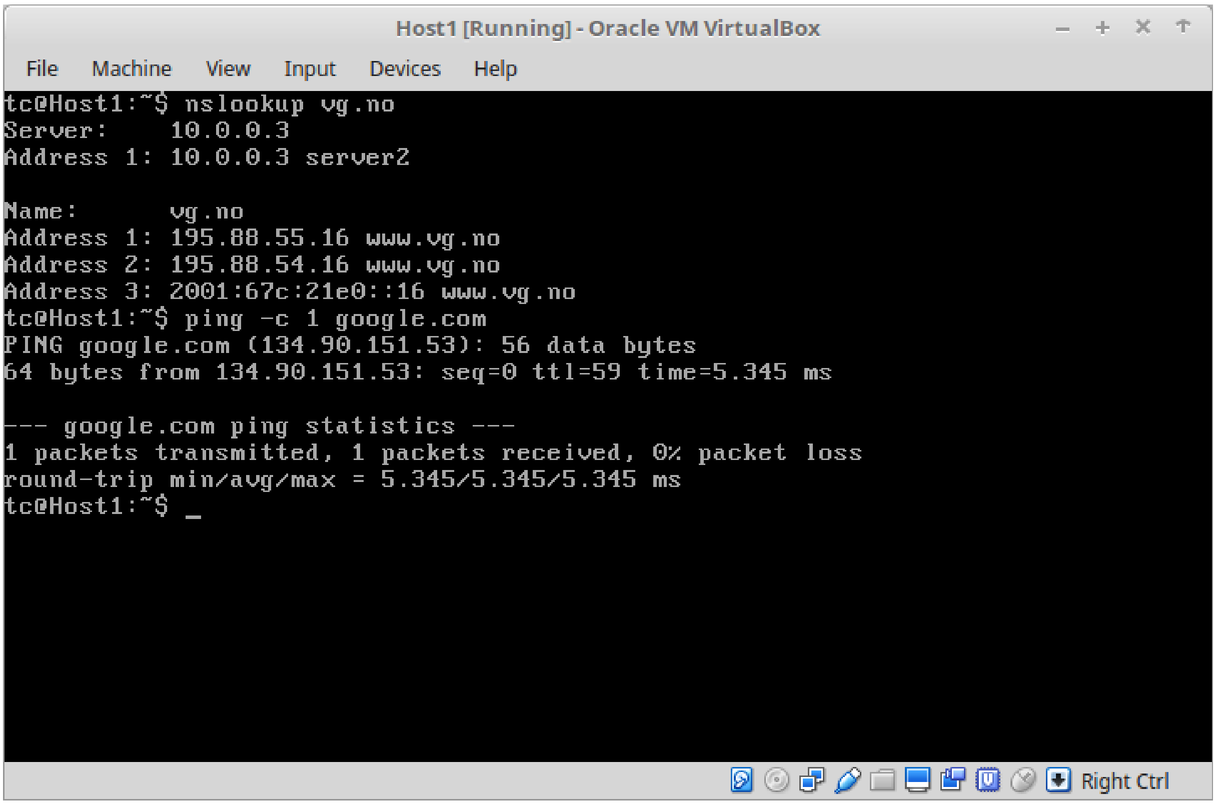
The address was 10.0.4.3 .


Gateway [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

```
tc@GW:~$ cat /etc/resolv.conf
nameserver 10.0.4.3
tc@GW:~$ _
```

Now adding the line : *echo* «nameserver 10.0.4.3» >> /etc/resolv.conf

in the *bootlocal.sh* file at Server2. This provides Server2 with an external nameserver that dnsmasq can read. We use *nslookup* to check for *vg.no* ,Then pinging from Host1 to google.com we can see that it has internet connection as well.


Host1 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Right Ctrl

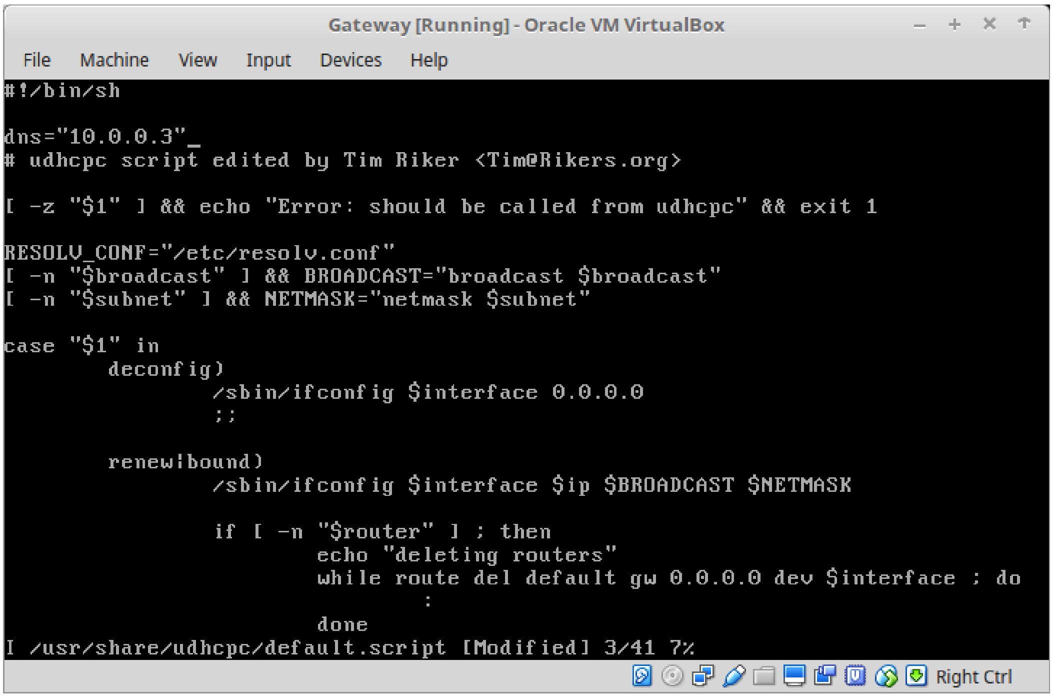
```
tc@Host1:~$ nslookup vg.no
Server:    10.0.0.3
Address 1: 10.0.0.3 server2

Name:      vg.no
Address 1: 195.88.55.16 www.vg.no
Address 2: 195.88.54.16 www.vg.no
Address 3: 2001:67c:21e0::16 www.vg.no
tc@Host1:~$ ping -c 1 google.com
PING google.com (134.90.151.53): 56 data bytes
64 bytes from 134.90.151.53: seq=0 ttl=59 time=5.345 ms

--- google.com ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 5.345/5.345/5.345 ms
tc@Host1:~$ _
```

d.

According to content of */etc/resolv.conf* of Gateway we can not look for local names (like Host1). To configure this I added the line:

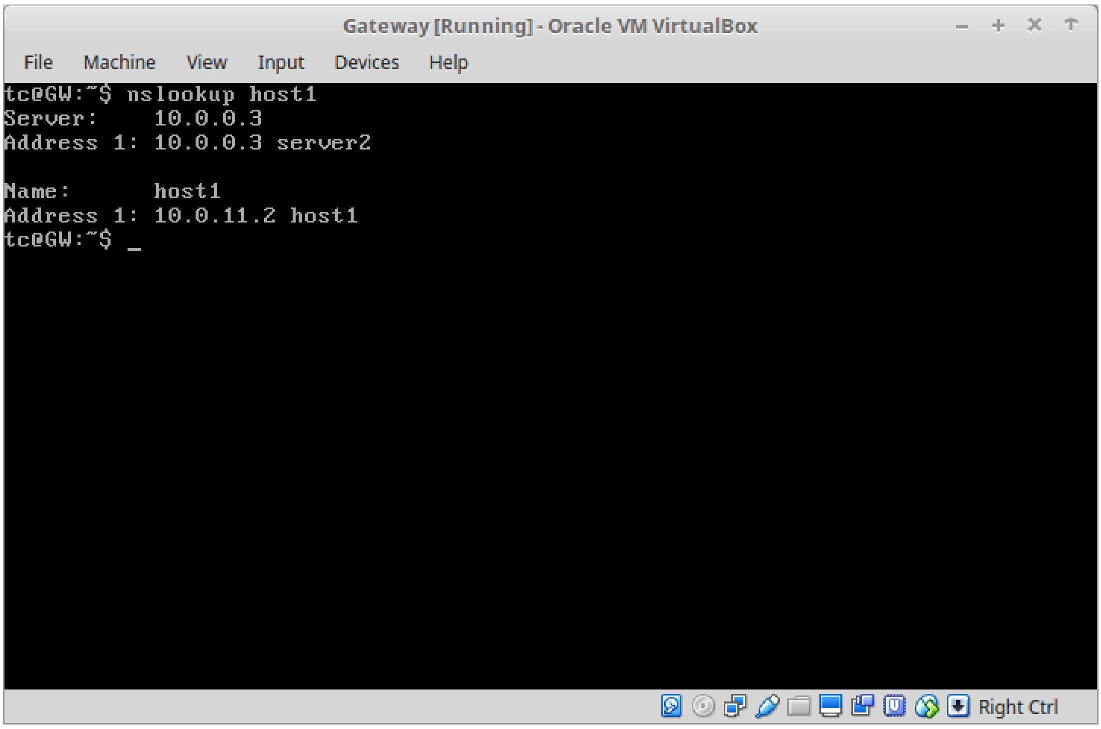
echo nameserver 10.0.0.3 >> /etc/resolv.conf

to the *bootlocal.sh* file at the Gateway. This worked on my personal computer but did not work in the computers at the university. So I configured the «*/usr/share/udhcpc/default.script*» file. I added the line «*dns=10.0.0.3*» to the start of the file as seen below.



```
#!/bin/sh

dns="10.0.0.3"_
# udhcpc script edited by Tim Riker <Tim@Rikers.org>

[ -z "$1" ] && echo "Error: should be called from udhcpc" && exit 1

RESOLV_CONF="/etc/resolv.conf"
[ -n "$broadcast" ] && BROADCAST="broadcast $broadcast"
[ -n "$subnet" ] && NETMASK="netmask $subnet"

case "$1" in
        deconfig)
                /sbin/ifconfig $interface 0.0.0.0
                ;;

        renew|bound)
                /sbin/ifconfig $interface $ip $BROADCAST $NETMASK

                if [ -n "$router" ] ; then
                        echo "deleting routers"
                        while route del default gw 0.0.0.0 dev $interface ; do
                                :
                        done
[ /usr/share/udhcpc/default.script [Modified] 3/41 7%
```

Then after saving persistently and then rebooting, I had connection to the nameserver at Server2(10.0.0.3).



```
tc@GW:~$ nslookup host1
Server:    10.0.0.3
Address 1: 10.0.0.3 server2

Name:      host1
Address 1: 10.0.11.2 host1
tc@GW:~$ _
```

7. Documenting the final dnsmasq configuration.

a. For proper the dnsmasq configuration the settings we have enabled are:

i. domain-needed

Tells the dnsmasq never to give plain names (without dots or domain parts) to upstream nameservers. A «not found» is returned if the name in not known from */etc/hosts.*

ii. bogus-priv

It reverse lookups the private Ips which are not found in /etc/hosts, «no such domain» is answered to the DHCP lease file if not found.

iii. Dhcp-range

This is the range in which the IP addresses will be handed out. The format is <start-address>,<end-address>,<lease time>. The lease time is the amount of time the IP will be linked to the host.

iv. dhcp-host

This leases an IP address to the host with the given Hwaddress for the specified amount of time.

v. dhcp-option with router

This tells the hosts which gateway to use when a hosts requests an IP address.

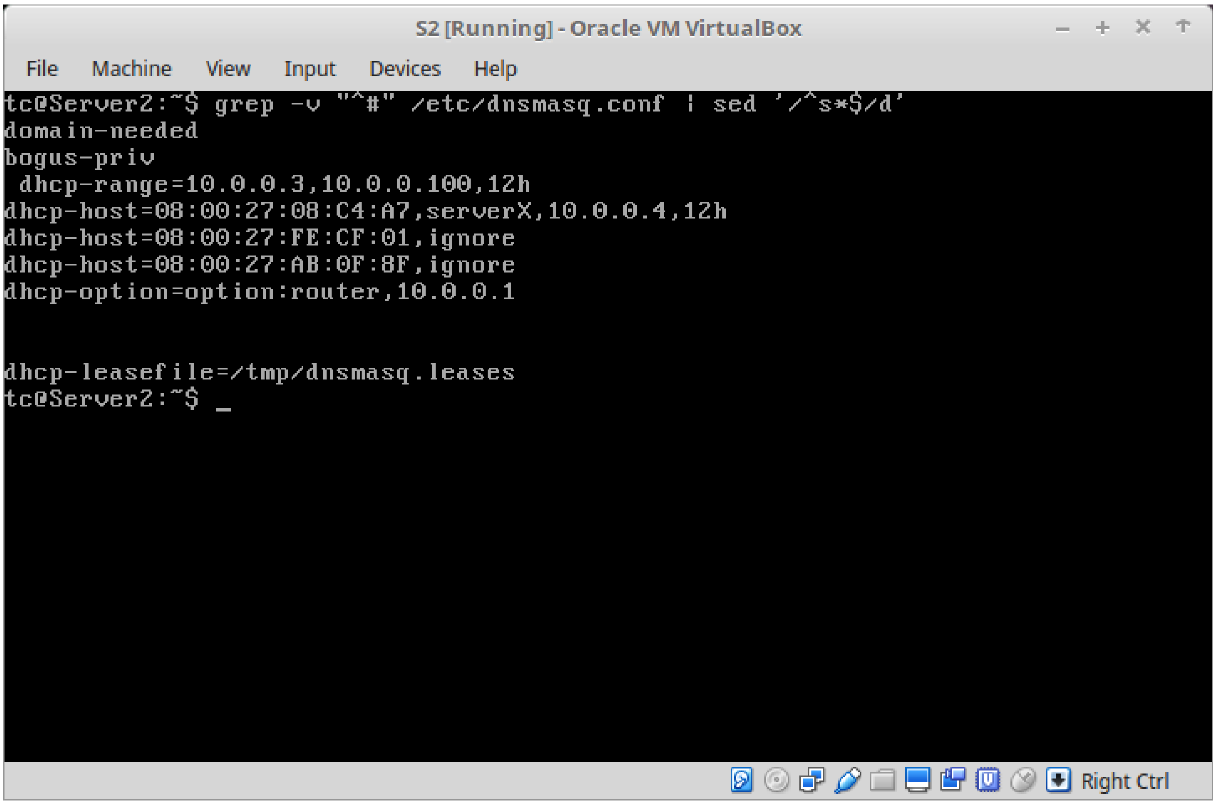
vi. dhcp-leasefile

This is where the DHCP lease information is stored.

This information about the dnsmasq was gathered from the dnsmasq-man page[1].

To see the *dnsmasq.conf* file I use the command:

grep -v “^#” /etc/dnsmasq.conf | sed ‘/^s$/d’*



```
tc@Server2:~$ grep -v "^#" /etc/dnsmasq.conf | sed '/^s*$/d'
domain-needed
bogus-priv
 dhcp-range=10.0.0.3,10.0.0.100,12h
dhcp-host=08:00:27:08:C4:A7,serverX,10.0.0.4,12h
dhcp-host=08:00:27:FE:CF:01,ignore
dhcp-host=08:00:27:AB:0F:8F,ignore
dhcp-option=option:router,10.0.0.1


dhcp-leasefile=/tmp/dnsmasq.leases
tc@Server2:~$ _
```

Here we can see the *dnsmasq.conf* file with our configurations.

Part II : Using TCPDump to capture TCP connections

a. Checking that we have connectivity between Host1 and Server2. First we ping Server2 from Host1, and make sure there is connection (which there was). Then the *openssh* server is started on both Server2 and Host1, if not already running. This can be done by issuing the following command at both the hosts;

“sudo /usr/local/etc/init.d/openssh start”

 I have already set up the passwords for the hosts, if this is not done, it can be done with the command : *“sudo passwd tc”* where *tc* is the username.

[1] http://www.thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html

Then I have also installed *tcpdump* at choke from before using the "*ab tcpdump"* command.

b. Now we run the TCPdump to capture traffic in interface which is connected to the Host1, that is eth1 at Choke. The command that is used is :

"sudo tcpdump -i eth1 -w ssh.dump not port 67"

This command is run at Choke. This captures all the traffic at the adapter *eth1* at Choke and stores it in the file *ssh.dump*. The *not port 67* tells the tcpdump not to capture traffic on port 67.

c. Running the command "*ssh 10.0.0.3"* from Host1, connects Host1 to Server2 (10.0.0.3). The TCPDump at choke captures the SSH TCP connection.

d. Now it is possible to simulate the packet loss in this TCP connection by shutting down the adapter at Server2. This is done with the command "*sudo ifconfig eth0 down"* at Server2.

e. Now we can see the tcpdump file where we saved all the packets that were captured. When the *ssh* connection between Host1 and Server2 is established, TCP uses a 3 way handshake:

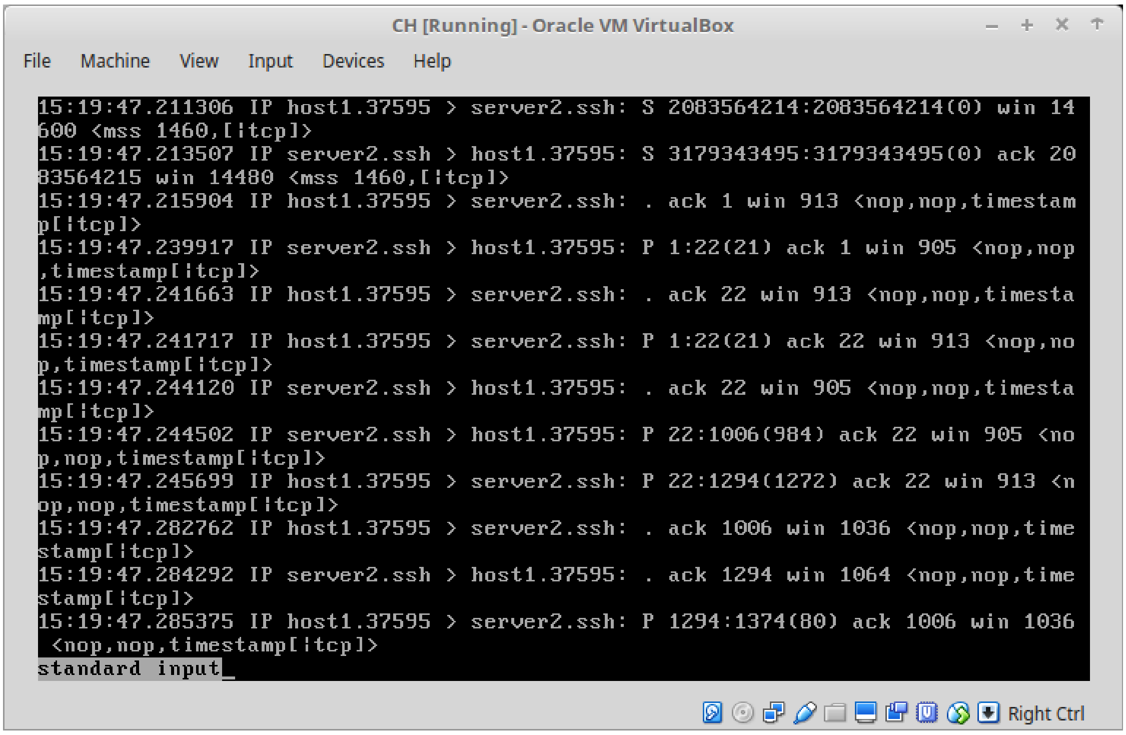
I. Host1 sends SYN

ii. Server2 responds with SYN,ACK

iii. Host1 sends ACK

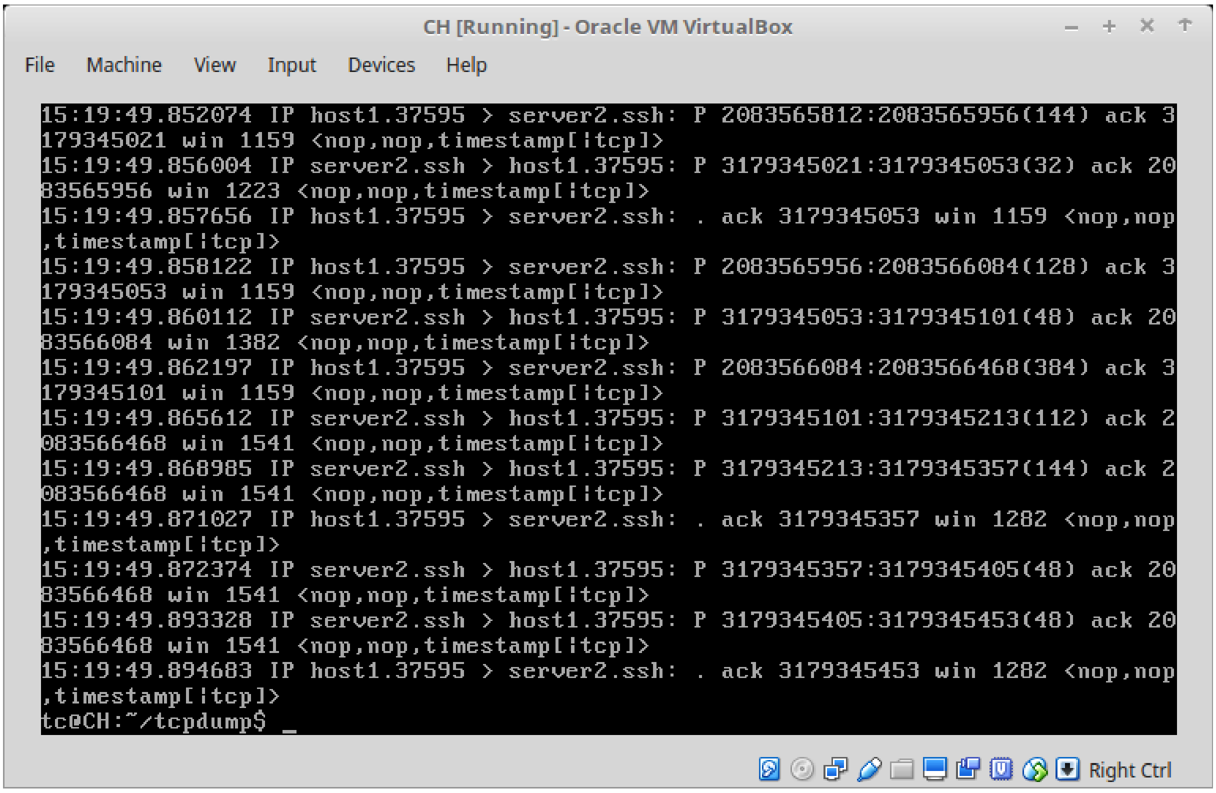
We can see this in the following image as well. I have used the following line to filter the packages scene:

tcpdump -r ssh.dump "tcp[tcpflags] & (tcp-syn|tcp-ack) != 0



```
15:19:47.211306 IP host1.37595 > server2.ssh: S 2083564214:2083564214(0) win 14
600 <mss 1460,[|tcp]>
15:19:47.213507 IP server2.ssh > host1.37595: S 3179343495:3179343495(0) ack 20
83564215 win 14480 <mss 1460,[|tcp]>
15:19:47.215904 IP host1.37595 > server2.ssh: . ack 1 win 913 <nop,nop,timestam
p[|tcp]>
15:19:47.239917 IP server2.ssh > host1.37595: P 1:22(21) ack 1 win 905 <nop,nop
,timestamp[|tcp]>
15:19:47.241663 IP host1.37595 > server2.ssh: . ack 22 win 913 <nop,nop,timesta
mp[|tcp]>
15:19:47.241717 IP host1.37595 > server2.ssh: P 1:22(21) ack 22 win 913 <nop,no
p,timestamp[|tcp]>
15:19:47.244120 IP server2.ssh > host1.37595: . ack 22 win 905 <nop,nop,timesta
mp[|tcp]>
15:19:47.244502 IP server2.ssh > host1.37595: P 22:1006(984) ack 22 win 905 <no
p,nop,timestamp[|tcp]>
15:19:47.245699 IP host1.37595 > server2.ssh: P 22:1294(1272) ack 22 win 913 <n
op,nop,timestamp[|tcp]>
15:19:47.282762 IP host1.37595 > server2.ssh: . ack 1006 win 1036 <nop,nop,time
stamp[|tcp]>
15:19:47.284292 IP server2.ssh > host1.37595: . ack 1294 win 1064 <nop,nop,time
stamp[|tcp]>
15:19:47.285375 IP host1.37595 > server2.ssh: P 1294:1374(80) ack 1006 win 1036
 <nop,nop,timestamp[|tcp]>
standard input_
```

When the adapter (eth0) at Server2 is down (*ifconfig eth0 down)* the Host1 freezes, the tcpdump captures a packet where Host1 sends a packet but Server2 does not respond.



```
15:19:49.852074 IP host1.37595 > server2.ssh: P 2083565812:2083565956(144) ack 3
179345021 win 1159 <nop,nop,timestamp[|tcp]>
15:19:49.856004 IP server2.ssh > host1.37595: P 3179345021:3179345053(32) ack 20
83565956 win 1223 <nop,nop,timestamp[|tcp]>
15:19:49.857656 IP host1.37595 > server2.ssh: . ack 3179345053 win 1159 <nop,nop
,timestamp[|tcp]>
15:19:49.858122 IP host1.37595 > server2.ssh: P 2083565956:2083566084(128) ack 3
179345053 win 1159 <nop,nop,timestamp[|tcp]>
15:19:49.860112 IP server2.ssh > host1.37595: P 3179345053:3179345101(48) ack 20
83566084 win 1382 <nop,nop,timestamp[|tcp]>
15:19:49.862197 IP host1.37595 > server2.ssh: P 2083566084:2083566468(384) ack 3
179345101 win 1159 <nop,nop,timestamp[|tcp]>
15:19:49.865612 IP server2.ssh > host1.37595: P 3179345101:3179345213(112) ack 2
083566468 win 1541 <nop,nop,timestamp[|tcp]>
15:19:49.868985 IP server2.ssh > host1.37595: P 3179345213:3179345357(144) ack 2
083566468 win 1541 <nop,nop,timestamp[|tcp]>
15:19:49.871027 IP host1.37595 > server2.ssh: . ack 3179345357 win 1282 <nop,nop
,timestamp[|tcp]>
15:19:49.872374 IP server2.ssh > host1.37595: P 3179345357:3179345405(48) ack 20
83566468 win 1541 <nop,nop,timestamp[|tcp]>
15:19:49.893328 IP server2.ssh > host1.37595: P 3179345405:3179345453(48) ack 20
83566468 win 1541 <nop,nop,timestamp[|tcp]>
15:19:49.894683 IP host1.37595 > server2.ssh: . ack 3179345453 win 1282 <nop,nop
,timestamp[|tcp]>
tc@CH:~/tcpdump$ _
```

Reference

I. http://www.thekelleys.org.uk/dnsmasq/docs/dnsmasq-man.html; September 7, 2016

ii. http://www.tcpdump.org/tcpdump_man.html; September 7, 2016

iii.