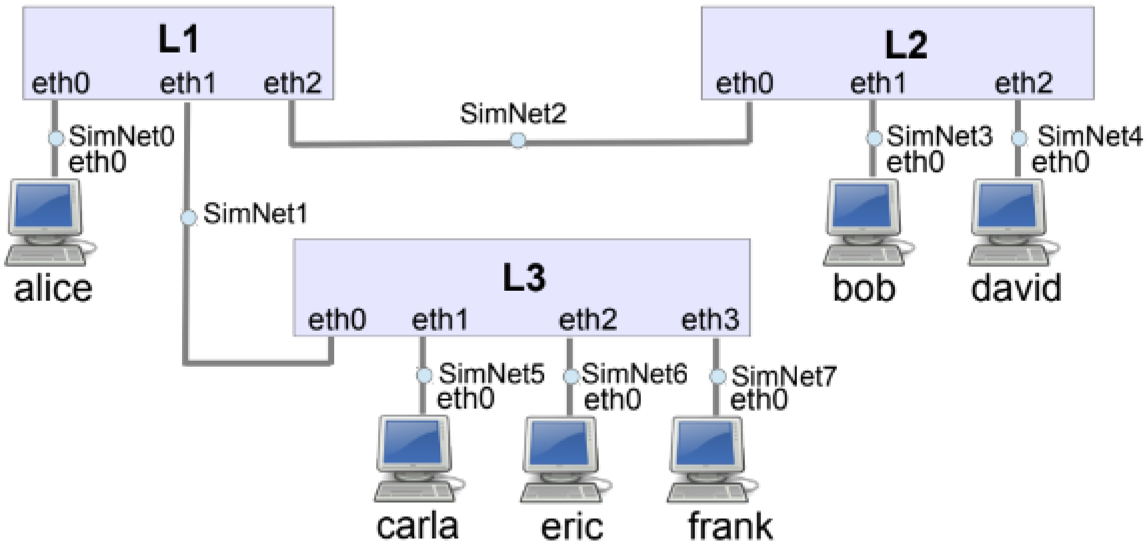
P2: IP routing



Exercise1– In this exercise we will examine how Ethernet switching and IP routing work together. We will use the simulation scenario of Figure 1.

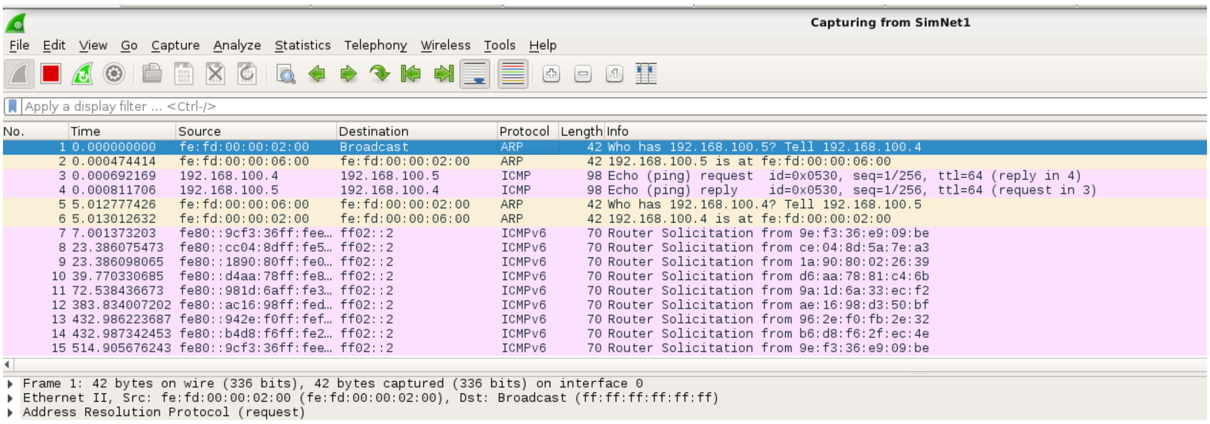
alice eth0 192.168.100.3
bob eth0 192.168.101.4
frank eth0 192.168.102.5

1. Using the IP address block 192.168.100.0/24 assign an IP address to bob and frank. Example:

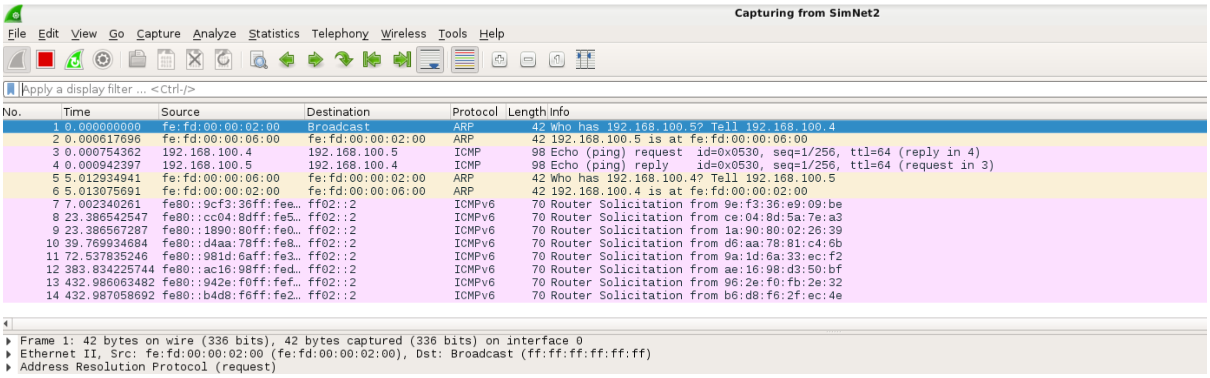
root@frank:~# ifconfig eth0 192.168.100.5/24
root@bob:~# ifconfig eth0 192.168.100.4/24
root@bob:~# ping -c 1 192.168.100.5

Capture on SimNet1, SimNet2 and SimNet3 and try a ping from bob to frank of one message (use the IPthat you have assigned to frank):

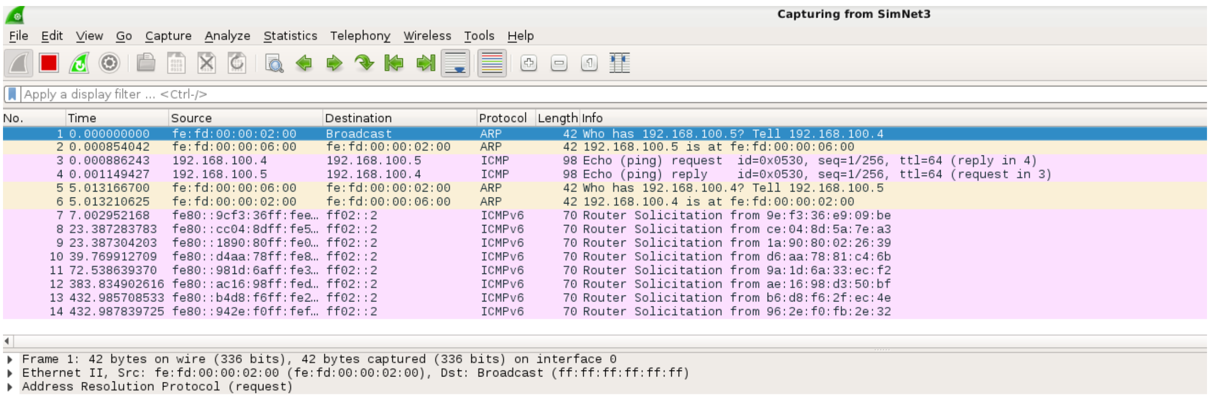
```
root@bob:~# ping -c 192.168.100.5
Usage: ping [-LRUbdfnqrvVaAD] [-c count] [-i interval] [-w deadline]
            [-p pattern] [-s packetsize] [-t ttl] [-I interface]
            [-M pmtudisc-hint] [-m mark] [-S sndbuf]
            [-T tstamp-options] [-Q tos] [hop1 ...] destination
root@bob:~# ping -c 1 192.168.100.5
PING 192.168.100.5 (192.168.100.5) 56(84) bytes of data.
64 bytes from 192.168.100.5: icmp_req=1 ttl=64 time=1.34 ms

--- 192.168.100.5 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.347/1.347/1.347/0.000 ms
```



Simnet1



Simnet2



Simnet3

In all of them we can see an ICMP message request sent from 192.168.100.5(frank) and an ICMP message response from 192.168.100.4(bob). We also have a reminder ARP that checks that 192.168.100.5 has still the same MAC because the ARP caches are reset in a certain lapse of time to avoid unused or nonvalid IP addresses.

2. Now we will convert L1, which is a Linux system into a router.

Capture on SimNet1, SimNet2 and SimNet3 and try a ping of one message from alice to bob and another ping of one message from bob to frank.

//from L1 to make it a router
ifconfig br1 down
brctl delbr br1
echo 1 > /proc/sys/net/ipv4/ip_forward

ifconfig eth0 192.168.100.1/24
ifconfig eth1 192.168.101.1/24
ifconfig eth2 192.168.102.1/24

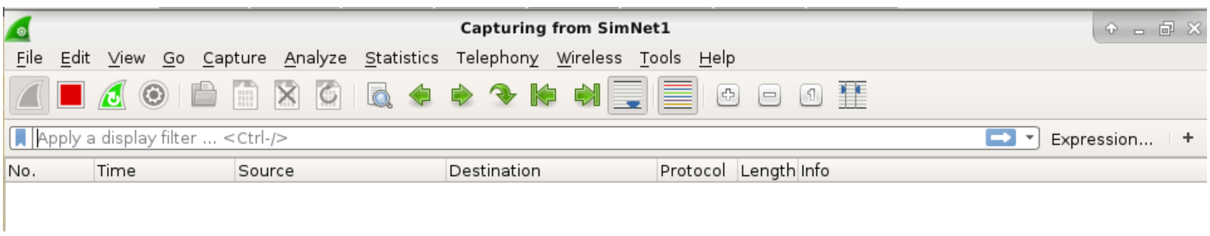
//from L1 we add the routing rules
route add -net 192.168.100.0/24 gw 192.168.100.1
route add -net 192.168.101.0/24 gw 192.168.101.1
route add -net 192.168.102.0/24 gw 192.168.102.1

//from alice
ifconfig eth0 192.168.100.2/24
route add default gw 192.168.100.1

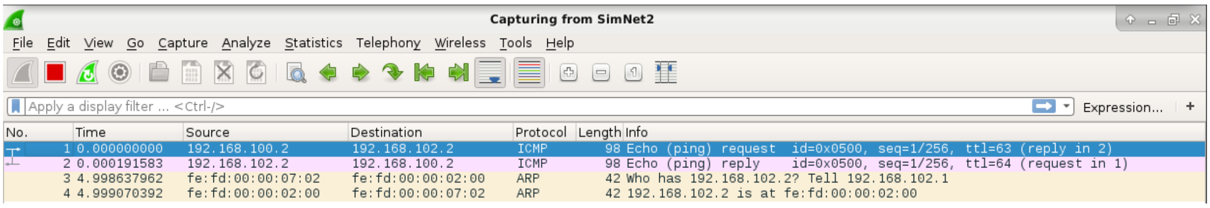
//from frank
ifconfig eth0 192.168.101.4/24
route add default gw 192.168.101.1

//from bob
ifconfig eth0 192.168.102.2/24
route add default gw 192.168.102.1

//ping alice to bob
ping -c 1 192.168.102.2

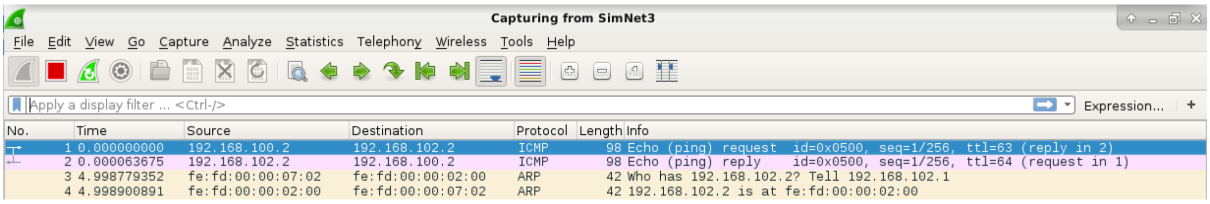


Simnet1



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.2	192.168.102.2	ICMP	98	Echo (ping) request id=0x0500, seq=1/256, ttl=63 (reply in 2)
2	0.000191583	192.168.102.2	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0500, seq=1/256, ttl=64 (request in 1)
3	4.998637962	fe:fd:00:00:07:02	fe:fd:00:00:02:00	ARP	42	Who has 192.168.102.2? Tell 192.168.102.1
4	4.999070392	fe:fd:00:00:02:00	fe:fd:00:00:07:02	ARP	42	192.168.102.2 is at fe:fd:00:00:02:00

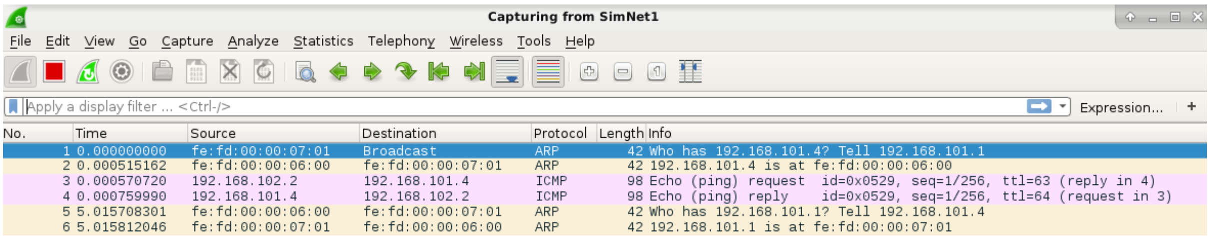
Simnet2



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.100.2	192.168.102.2	ICMP	98	Echo (ping) request id=0x0500, seq=1/256, ttl=63 (reply in 2)
2	0.000063675	192.168.102.2	192.168.100.2	ICMP	98	Echo (ping) reply id=0x0500, seq=1/256, ttl=64 (request in 1)
3	4.998779352	fe:fd:00:00:07:02	fe:fd:00:00:02:00	ARP	42	Who has 192.168.102.2? Tell 192.168.102.1
4	4.998900891	fe:fd:00:00:02:00	fe:fd:00:00:07:02	ARP	42	192.168.102.2 is at fe:fd:00:00:02:00

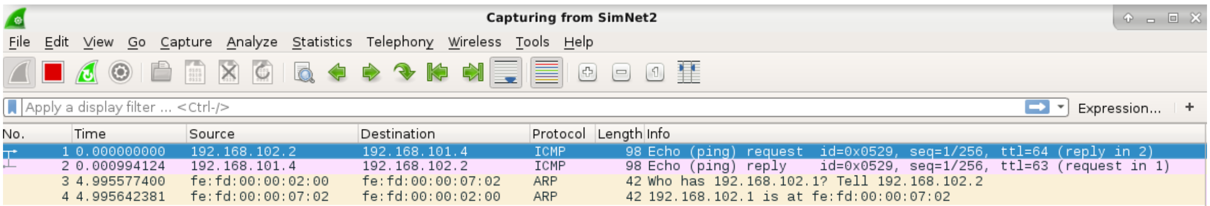
Simnet3

//ping bob to frank
ping -c 1 192.168.101.4



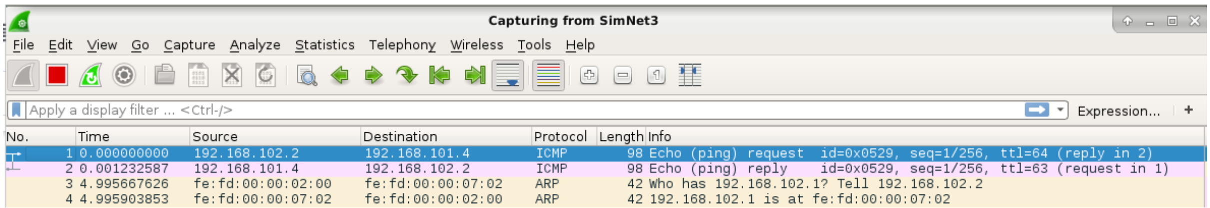
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:07:01	Broadcast	ARP	42	Who has 192.168.101.4? Tell 192.168.101.1
2	0.000515162	fe:fd:00:00:06:00	fe:fd:00:00:07:01	ARP	42	192.168.101.4 is at fe:fd:00:00:06:00
3	0.000570720	192.168.102.2	192.168.101.4	ICMP	98	Echo (ping) request id=0x0529, seq=1/256, ttl=63 (reply in 4)
4	0.000759990	192.168.101.4	192.168.102.2	ICMP	98	Echo (ping) reply id=0x0529, seq=1/256, ttl=64 (request in 3)
5	5.015708301	fe:fd:00:00:06:00	fe:fd:00:00:07:01	ARP	42	Who has 192.168.101.1? Tell 192.168.101.4
6	5.015812046	fe:fd:00:00:07:01	fe:fd:00:00:06:00	ARP	42	192.168.101.1 is at fe:fd:00:00:07:01

Simnet1



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.102.2	192.168.101.4	ICMP	98	Echo (ping) request id=0x0529, seq=1/256, ttl=64 (reply in 2)
2	0.000994124	192.168.101.4	192.168.102.2	ICMP	98	Echo (ping) reply id=0x0529, seq=1/256, ttl=63 (request in 1)
3	4.995577400	fe:fd:00:00:02:00	fe:fd:00:00:07:02	ARP	42	Who has 192.168.102.1? Tell 192.168.102.2
4	4.995642381	fe:fd:00:00:07:02	fe:fd:00:00:02:00	ARP	42	192.168.102.1 is at fe:fd:00:00:07:02

Simnet2

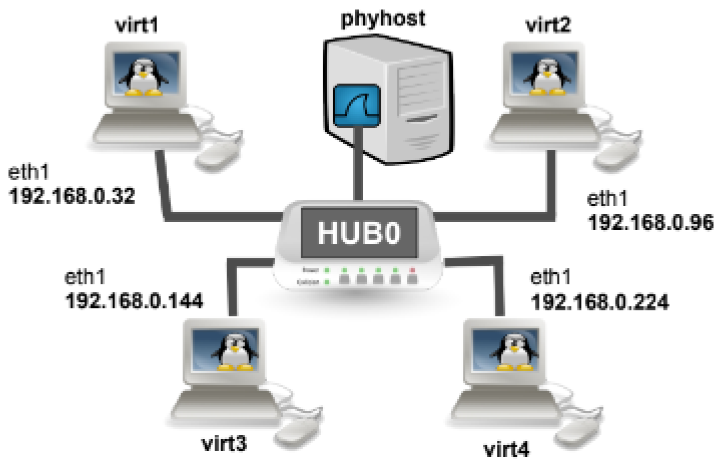


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.102.2	192.168.101.4	ICMP	98	Echo (ping) request id=0x0529, seq=1/256, ttl=64 (reply in 2)
2	0.001232587	192.168.101.4	192.168.102.2	ICMP	98	Echo (ping) reply id=0x0529, seq=1/256, ttl=63 (request in 1)
3	4.995667626	fe:fd:00:00:02:00	fe:fd:00:00:07:02	ARP	42	Who has 192.168.102.1? Tell 192.168.102.2
4	4.995903853	fe:fd:00:00:07:02	fe:fd:00:00:02:00	ARP	42	192.168.102.1 is at fe:fd:00:00:07:02

Simnet3

Describe in detail the configuration that you propose and all the processes involved that make the pings work.
Include in your explanation the MACs learned by the switches, the ARP caches and the MAC (and IP addresses)
in each frame captured. Explain also the differences with the previous case, when L1 was a switch.
Bob already knows the route to L1 so there is not a broadcast in the beginning.

Exercise2– In this first exercise, we will examine how the direct forwarding of IP datagrams works. We will use the virtual network topology shown in Figure 2, which has a hub and four virtual machines: virt1, virt2, virt3 and virt4.
Type on your physical host the following command to start the scenario:



//from terminal
simctl ip-subnetting sh
start
get virt1

1. Analyzing the IP addresses assigned to virtX machines in the network (see Figure 2), find which is the larger mask (biggest quantity of ones) that makes all the machines on the topology belong to the same IP network. Ineach virtual machine, use ifconfig to configure the IP address and the mask found.

32 -> 0010 0000
96 -> 0110 0000
144 -> 1001 0000
224 -> 1110 0000
The largest mask that includes all is /24 (255.255.255.0)

//from virt1
ifconfig eth1 192.168.0.32/24
//from virt2
ifconfig eth1 192.168.0.96/24
//from virt3
ifconfig eth1 192.168.0.144/24
//from virt4
ifconfig eth1 192.168.0.224/24

2. Capture on the phyhost the SimNet0 with wireshark. Check that the ARP cache is empty in virt1:
//from virt1

ping -c 1 192.168.0.96
arp -n

Capturing from SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
2	0.000335157	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	192.168.0.96 is at fe:fd:00:00:02:01
3	0.000550163	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0x8c04, seq=1/256, ttl=64 (reply in 4)
4	0.000865617	192.168.0.96	192.168.0.32	ICMP	98	Echo (ping) reply id=0x8c04, seq=1/256, ttl=64 (request in 3)
5	5.001842870	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	Who has 192.168.0.32? Tell 192.168.0.96
6	5.002174616	fe:fd:00:00:01:01	fe:fd:00:00:02:01	ARP	42	192.168.0.32 is at fe:fd:00:00:01:01

```
virt1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.96             ether   FE:FD:00:00:02:01   C                     eth1
```

```
virt2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.32             ether   FE:FD:00:00:01:01   C                     eth1
```

```
virt3:~# arp -n
virt3:~#
```

```
virt4:~# arp -n
virt4:~#
```

//second ping
ping -c 1 192.168.0.96

Capturing from SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
2	0.000279618	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	192.168.0.96 is at fe:fd:00:00:02:01
3	0.000490414	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0x9004, seq=1/256, ttl=64 (reply in 4)
4	0.000700666	192.168.0.96	192.168.0.32	ICMP	98	Echo (ping) reply id=0x9004, seq=1/256, ttl=64 (request in 3)
5	5.007427275	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	Who has 192.168.0.32? Tell 192.168.0.96
6	5.007590142	fe:fd:00:00:01:01	fe:fd:00:00:02:01	ARP	42	192.168.0.32 is at fe:fd:00:00:01:01
7	25.666335102	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0x9104, seq=1/256, ttl=64 (reply in 8)
8	25.668775419	192.168.0.96	192.168.0.32	ICMP	98	Echo (ping) reply id=0x9104, seq=1/256, ttl=64 (request in 7)

We can see in the second ping, there is not arp frame because the route is already established.

Note. Observe that Linux generates a gratuitous ARP some time after the end of each transmission. These gratuitous ARPs are unicast and they are intended for refreshing the ARP cache.

3. Now, let's delete the ARP entry for 192.168.0.96 in virt1:
//from virt1
arp -d 192.168.0.96

```
virt1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.96                     (incomplete)                              eth1
```

ping -c 1 192.168.0.96

Capturing from SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
2	0.000281948	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	192.168.0.96 is at fe:fd:00:00:02:01
3	0.000455009	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0x9f04, seq=1/256, ttl=64 (reply in 4)
4	0.000650889	192.168.0.96	192.168.0.32	ICMP	98	Echo (ping) reply id=0x9f04, seq=1/256, ttl=64 (request in 3)
5	4.999711436	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	Who has 192.168.0.32? Tell 192.168.0.96
6	4.999909579	fe:fd:00:00:01:01	fe:fd:00:00:02:01	ARP	42	192.168.0.32 is at fe:fd:00:00:01:01

It needs to do a broadcast (send arp frame) again because there is no arp entry that knows the destiny MAC.

4. Now, let’s create an erroneous mapping for 192.168.0.96 in virt1:

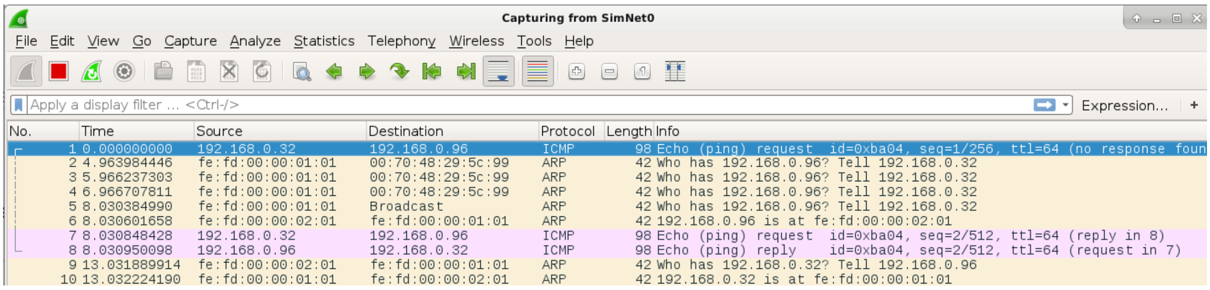
//from virt1

arp -s 192.168.0.96 00:70:48:29:5c:99 temp

```
virt1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.96             ether   00:70:48:29:5C:99   C                     eth1
```

ping -c 2 -i 8 192.168.0.96

```
virt1:~# ping -c 2 -i 8 192.168.0.96
PING 192.168.0.96 (192.168.0.96) 56(84) bytes of data.
64 bytes from 192.168.0.96: icmp_seq=2 ttl=64 time=21.1 ms

--- 192.168.0.96 ping statistics ---
2 packets transmitted, 1 received, 50% packet loss, time 8010ms
rtt min/avg/max/mdev = 21.197/21.197/21.197/0.000 ms
```



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0xba04, seq=1/256, ttl=64 (no response foun
2	4.963984446	fe:fd:00:00:01:01	00:70:48:29:5c:99	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
3	5.966237303	fe:fd:00:00:01:01	00:70:48:29:5c:99	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
4	6.966707811	fe:fd:00:00:01:01	00:70:48:29:5c:99	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
5	8.030384990	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.0.96? Tell 192.168.0.32
6	8.030601658	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	192.168.0.96 is at fe:fd:00:00:02:01
7	8.030848428	192.168.0.32	192.168.0.96	ICMP	98	Echo (ping) request id=0xba04, seq=2/512, ttl=64 (reply in 8)
8	8.030950098	192.168.0.96	192.168.0.32	ICMP	98	Echo (ping) reply id=0xba04, seq=2/512, ttl=64 (request in 7)
9	13.031889914	fe:fd:00:00:02:01	fe:fd:00:00:01:01	ARP	42	Who has 192.168.0.32? Tell 192.168.0.96
10	13.032224190	fe:fd:00:00:01:01	fe:fd:00:00:02:01	ARP	42	192.168.0.32 is at fe:fd:00:00:01:01

El primer ping se pierde porque la MAC era errónea, entonces virt1 intenta llegar hasta virt2 y envía arp unicast para virt2 con MAC errónea, como no recibe respuesta, descarta la primera trama, y al enviar el segundo ping envía un broadcast y recibe respuesta, por eso el segundo ping se envia exitosamente.

5. Now, we need to “clean” the ARP cache of virt1:

//from virt1

ip neigh flush all

```
virt1:~# arp -n
virt1:~#
```

Next, you have to find out which is the mask needed to divide the network into two subnets so that virt1 and virt2 belong to one subnet and virt3 and virt4 belong to another subnet. Configure the IP/mask on each virtual machine and explain how you check the configuration.

6. Which would be the smallest mask (minimum number of ones) that makes not possible the IP communication between the machines on the topology?

virt1: 32 ->   0010 0000
virt2: 96 ->   0110 0000
virt3: 144 -> 1001 0000
virt4: 224 -> 1110 0000

- **we can use the mask /25 to divide virt1 and virt2 from virt3 and virt4.**
- **we can use the mask /26 to put each host in different net making not possible the IP communication between the machines.**

//from virt1
ifconfig eth1 192.168.0.32/25
//from virt2
ifconfig eth1 192.168.0.96/25
//from virt3
ifconfig eth1 192.168.0.144/25
//from virt4
ifconfig eth1 192.168.0.224/25

if we do ping from virt4, its successful to virt3 but not to virt2

```
virt4:~# ping -c 1 192.168.0.144 #virt3
PING 192.168.0.144 (192.168.0.144) 56(84) bytes of data.
64 bytes from 192.168.0.144: icmp_seq=1 ttl=64 time=0.191 ms

--- 192.168.0.144 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.191/0.191/0.191/0.000 ms
virt4:~# ping -c 1 192.168.0.96 #virt2
connect: Network is unreachable
virt4:~# S
```

7. Finally, let's test what happens when we have masks of different values on different interfaces. Configure the mask /24 in virt1 and virt3 and /25 in virt2 and virt4. Discuss in detail what happens when you ping from virt1 to the other machines and when you ping from virt2 to the other machines.

//from virt1
ifconfig eth1 192.168.0.32/24
//from virt2
ifconfig eth1 192.168.0.96/25
//from virt3
ifconfig eth1 192.168.0.144/24
//from virt4
ifconfig eth1 192.168.0.224/25

//from virt1
ping -c 1 192.168.0.96
ping -c 1 192.168.0.144

ping -c 1 192.168.0.224

```
virt1:~# ping -c 1 192.168.0.96
PING 192.168.0.96 (192.168.0.96) 56(84) bytes of data.
64 bytes from 192.168.0.96: icmp_seq=1 ttl=64 time=21.7 ms

--- 192.168.0.96 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 21.764/21.764/21.764/0.000 ms
virt1:~# ping -c 1 192.168.0.144
PING 192.168.0.144 (192.168.0.144) 56(84) bytes of data.
64 bytes from 192.168.0.144: icmp_seq=1 ttl=64 time=26.8 ms

--- 192.168.0.144 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 26.898/26.898/26.898/0.000 ms
virt1:~# ping -c 1 192.168.0.224
PING 192.168.0.224 (192.168.0.224) 56(84) bytes of data.

--- 192.168.0.224 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

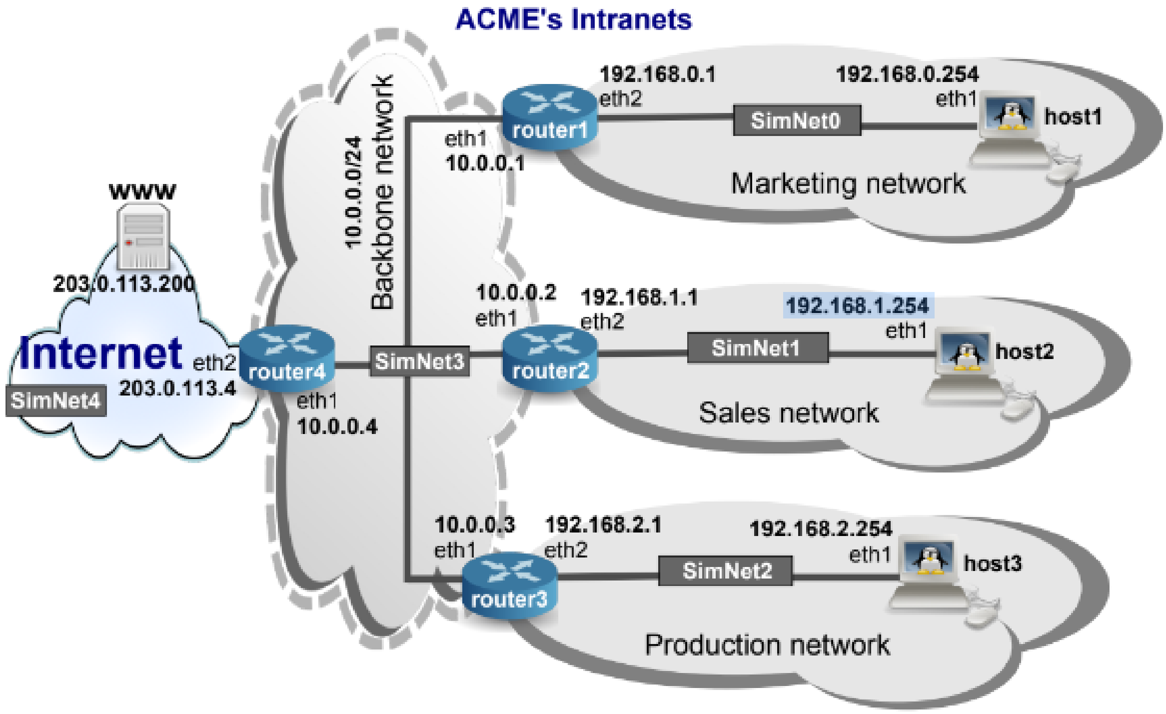
Only the ping to virt2 and virt3 works because from the point of view of virt1(using mask 24) virt1, virt2,virt3 and virt4 are in the same network (the same for virt3 which is using mask 24). Therefore, virt2 and virt3 can receive and answer the ping, but virt4 can only receive the ping and won't answer it.

//from virt2
ping -c 1 192.168.0.32
ping -c 1 192.168.0.144
ping -c 1 192.168.0.224

```
virt2:~# ping -c 1 192.168.0.32
PING 192.168.0.32 (192.168.0.32) 56(84) bytes of data.
64 bytes from 192.168.0.32: icmp_seq=1 ttl=64 time=0.299 ms

--- 192.168.0.32 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.299/0.299/0.299/0.000 ms
virt2:~# ping -c 1 192.168.0.144
connect: Network is unreachable
virt2:~# ping -c 1 192.168.0.224
connect: Network is unreachable
```

virt2 can only send an ICMP message to virt1 because they both share the same prefix under the mask 25, being both of them in the same network. But virt2 can't reach virt3 and virt4 because they are part of another network (using the mask 25). That's the reason of the error message: Network unreachable.

Exercise3– In this exercise we will configure a network for a small fictitious company called ACME. Figure 3 shows the network topology. ACME has three departments: marketing, sales and production. Each department is represented by a host and a router. Finally, the IP network 10.0.0.0/24 interconnects the routers (backbone network). Type on you physical host the following command to start the scenario:



//from terminal
simctl ip-routing sh
start

1. Analyzing the IP addresses assigned in the network, select an appropriate netmask for each network interface of host1, router1, host2 and router2. Verify the direct communications with pings. What would happen if you configure a mask /23 in host1 (only in this host) and try a ping to host2?

//from router1
ifconfig eth1 10.0.0.1/24
ifconfig eth2 192.168.0.1/24

//from host1
ifconfig eth1 192.168.0.254/24

//from router2
ifconfig eth1 10.0.0.2/24
ifconfig eth2 192.168.1.1/24

//from host2
ifconfig eth1 192.168.1.254/24

//from router1 ping router2 and host1
ping -c 1 10.0.0.2

ping -c 1 192.168.0.254
//from router2 ping host2
ping -c 1 192.168.1.254

```
router1:~# ping -c 1 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=21.6 ms

--- 10.0.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 21.609/21.609/21.609/0.000 ms
router1:~# ping -c 1 192.168.0.254
PING 192.168.0.254 (192.168.0.254) 56(84) bytes of data.
64 bytes from 192.168.0.254: icmp_seq=1 ttl=64 time=21.8 ms

--- 192.168.0.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 21.834/21.834/21.834/0.000 ms
```

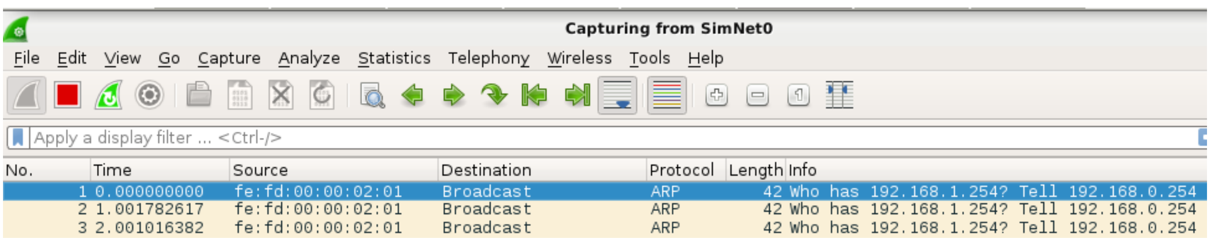
router1 ping router2 and host1

```
router2:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=64 time=20.7 ms

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 20.722/20.722/20.722/0.000 ms
```

router2 ping host2

//from host1
ifconfig eth1 192.168.0.254/23
ping -c 1 192.168.1.254

```
host1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
From 192.168.0.254 icmp_seq=1 Destination Host Unreachable

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```



It sends a broadcast message using arp to request the MAC address related to that IP address. Even if we named them in the same network (with mask 23) host1 cannot reach host2 because there is no direct connection between them. If we want a successful ping we should configure the routers.

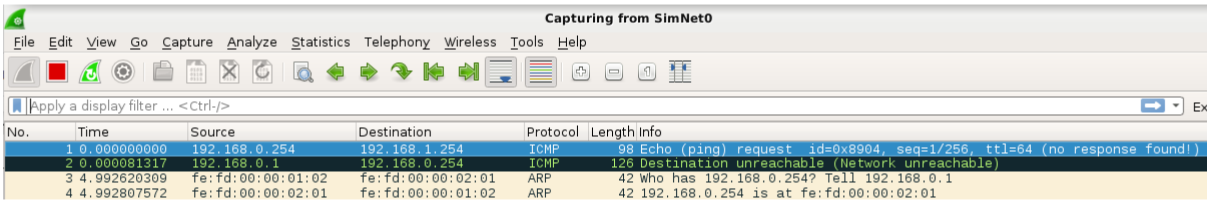
2. Configure a route in host1 to the 192.168.1.0/24 network. Send a ping of one message from host1 to 192.168.1.254. Obviously, the ping does not succeed. Explain the traffic captured in each interface (frame fields, IP packet fields and ICMP message fields, MAC addresses, IP addresses, etc.).

//from host1
route add -net default gw 192.168.0.1

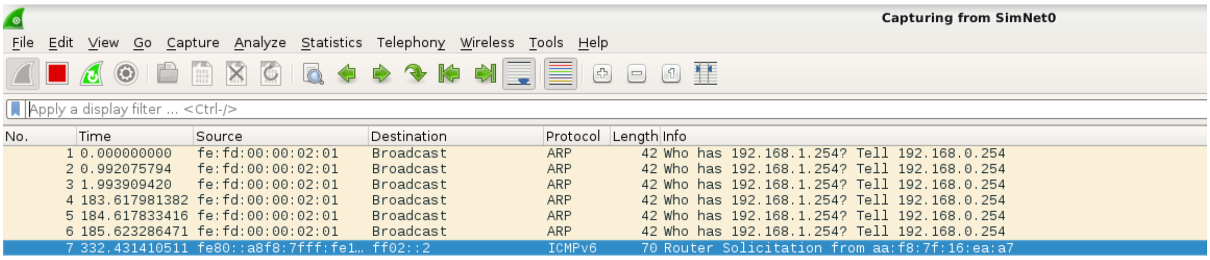
```
host1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth1
0.0.0.0         192.168.0.1     0.0.0.0         UG    0      0        0 eth1
```

//from host1
ping -c 1 192.168.1.254

```
host1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
From 192.168.0.1 icmp_seq=1 Destination Net Unreachable

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

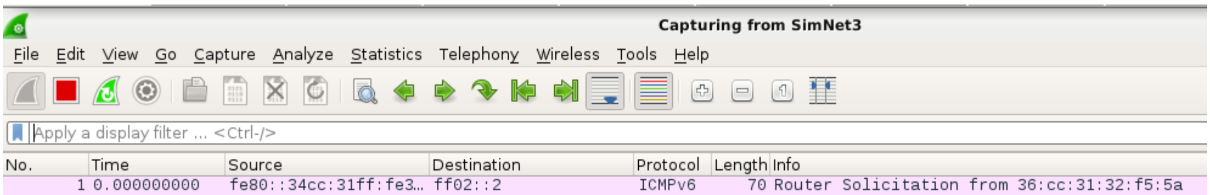


No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.0.254	192.168.1.254	ICMP	98	Echo (ping) request id=0x8904, seq=1/256, ttl=64 (no response found!)
2	0.000081317	192.168.0.1	192.168.0.254	ICMP	126	Destination unreachable (Network unreachable)
3	4.992620309	fe:fd:00:00:01:02	fe:fd:00:00:02:01	ARP	42	Who has 192.168.0.254? Tell 192.168.0.1
4	4.992807572	fe:fd:00:00:02:01	fe:fd:00:00:01:02	ARP	42	192.168.0.254 is at fe:fd:00:00:02:01



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
2	0.992075794	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
3	1.993909420	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
4	183.617981382	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
5	184.617833416	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
6	185.623286471	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.0.254
7	332.431410511	fe80::a8f8:7fff:fe1…	ff02::2	ICMPv6	70	Router Solicitation from aa:f8:7f:16:ea:a7

Simnet0 (router1-host1)



Simnet3 (router1-router2)

Host1 sends an icmp message to destination 192.168.1.254(host2) which is redirected to the router1 by the default rule.
Router1 doesn't have configured that ip, so we lost the ping and we got the message of Destination net unreachable

address therefore returns an icmp message of destination unreachable to host1.
//Router1 sends an ARP frame to the MAC of host1 asking for who has 192.168.0.254 to confirm that host1 has this IP address. Host1 responds with it's IP address and his MAC.//

3. Check that router1 has enabled IP forwarding. Then, add the necessary entry in the routing table of router1 to reach the sales network. Send again the ping from host1 to 192.168.1.254. Obviously, the ping still does not succeed. Explain the traffic that you observe in each interface and the contents of the ARP cache of the different
machines.

//from router1
cat /proc/sys/net/ipv4/ip_forward
route add -net 192.168.1.0/24 gw 10.0.0.2

```
router1:~# cat /proc/sys/net/ipv4/ip_forward
1
```

```
router1:~# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
10.0.0.0        0.0.0.0         255.255.255.0   U     0      0        0 eth1
192.168.1.0     10.0.0.2        255.255.255.0   UG    0      0        0 eth1
192.168.0.0     0.0.0.0         255.255.255.0   U     0      0        0 eth2
```

//from host1
ping -c 1 192.168.1.254

```
host1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
From 192.168.0.254 icmp_seq=1 Destination Host Unreachable

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms

host1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.254                    (incomplete)                              eth1
host1:~#
```

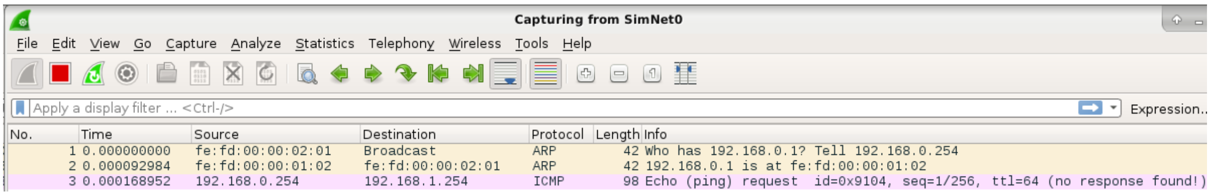
```
host1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms
```

```
host1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.1              ether   FE:FD:00:00:01:02   C                     eth1
```

```
router1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.254            ether   FE:FD:00:00:02:01   C                     eth2
10.0.0.2                 ether   FE:FD:00:00:03:01   C                     eth1
```

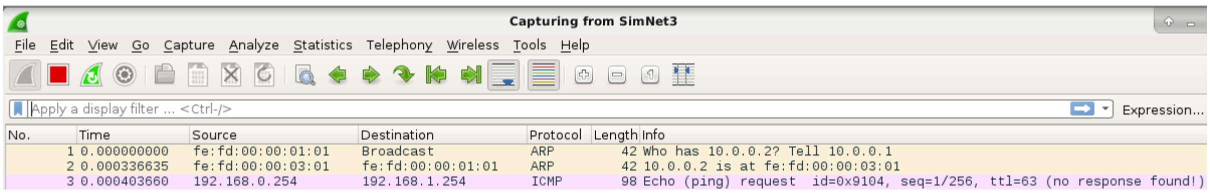
```
router2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.254            ether   FE:FD:00:00:04:01   C                     eth2
10.0.0.1                 ether   FE:FD:00:00:01:01   C                     eth1
```

```
host2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.1              ether   FE:FD:00:00:03:02   C                     eth1
```



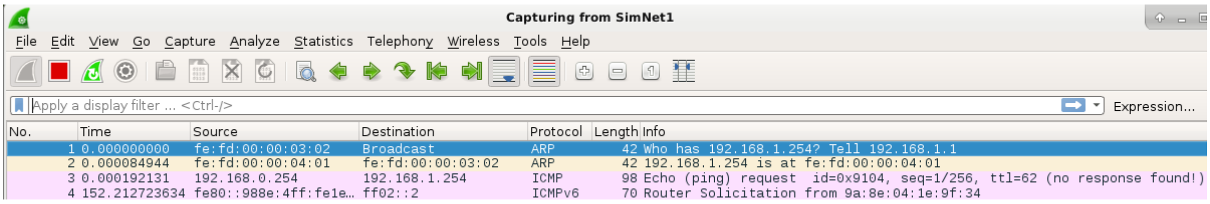
Simnet0(host1-router1)

Host1 sends an ARP frame to broadcast asking for the router's IP which is answered by the same router. Then the mac address of the router and its ip is saved in the arp table of host1 and the mac and ip of host1 is saved in the arp table of router1.
Host1 sends an icmp message to 192.168.1.254(host2) and is redirected to router1 and the message is not answered.



Simnet3(router1-router2)

With the new rule, the router1 has to redirect the icmp message to router2. In order to do so it has to know the MAC address, so it sends an ARP request which is broadcasted. Router2 replies with an ARP response with its MAC address. Router1 redirects the ICMP message sent from host1 which is not answered.



Simnet1(router2-host2)

Router2 sends an ARP requesting for the MAC address of the IP 192.168.1.254(host2). Host2 answers with its MAC and IP addresses. Then router2 sends the ICMP message which is not requested.

4. Finish the configuration adding entries for the network 192.168.0.0/24 in router2 and host2. Check that the ping works correctly. Explain the IP addresses observed in each packet, the MAC addresses and the contents of the ARP cache of host1, router1, host2 and router2.

//from host2
route add -net default gw 192.168.1.1
//from router2
cat /proc/sys/net/ipv4/ip_forward
route add -net 192.168.0.0/24 gw 10.0.0.1

//from host1
ping -c 1 192.168.1.254

```
host1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=62 time=63.1 ms

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 63.100/63.100/63.100/0.000 ms
```

```
host1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.1              ether   FE:FD:00:00:01:02   C                     eth1
```

```
router1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.254            ether   FE:FD:00:00:02:01   C                     eth2
10.0.0.2                 ether   FE:FD:00:00:03:01   C                     eth1
```

```
router2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.254            ether   FE:FD:00:00:04:01   C                     eth2
10.0.0.1                 ether   FE:FD:00:00:01:01   C                     eth1
```

```
host2:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.1.1              ether   FE:FD:00:00:03:02   C                     eth1
```

Capturing from SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:02:01	Broadcast	ARP	42	Who has 192.168.0.1? Tell 192.168.0.254
2	0.000098669	fe:fd:00:00:01:02	fe:fd:00:00:02:01	ARP	42	192.168.0.1 is at fe:fd:00:00:01:02
3	0.000257509	192.168.0.254	192.168.1.254	ICMP	98	Echo (ping) request id=0x9404, seq=1/256, ttl=64 (reply in 4)
4	0.042055469	192.168.1.254	192.168.0.254	ICMP	98	Echo (ping) reply id=0x9404, seq=1/256, ttl=62 (request in 3)
5	5.042454725	fe:fd:00:00:01:02	fe:fd:00:00:02:01	ARP	42	Who has 192.168.0.254? Tell 192.168.0.1
6	5.042597484	fe:fd:00:00:02:01	fe:fd:00:00:01:02	ARP	42	192.168.0.254 is at fe:fd:00:00:02:01

Simnet0(host1-router1)

Capturing from SimNet3

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 10.0.0.2? Tell 10.0.0.1
2	0.000175432	fe:fd:00:00:03:01	fe:fd:00:00:01:01	ARP	42	10.0.0.2 is at fe:fd:00:00:03:01
3	0.000289182	192.168.0.254	192.168.1.254	ICMP	98	Echo (ping) request id=0x9404, seq=1/256, ttl=63 (reply in 4)
4	0.021007696	192.168.1.254	192.168.0.254	ICMP	98	Echo (ping) reply id=0x9404, seq=1/256, ttl=63 (request in 3)
5	5.032249554	fe:fd:00:00:03:01	fe:fd:00:00:01:01	ARP	42	Who has 10.0.0.1? Tell 10.0.0.2
6	5.032630945	fe:fd:00:00:01:01	fe:fd:00:00:03:01	ARP	42	10.0.0.1 is at fe:fd:00:00:01:01

Simnet3(router1-router2)

Capturing from SimNet1

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:03:02	Broadcast	ARP	42	Who has 192.168.1.254? Tell 192.168.1.1
2	0.000086224	fe:fd:00:00:04:01	fe:fd:00:00:03:02	ARP	42	192.168.1.254 is at fe:fd:00:00:04:01
3	0.000146511	192.168.0.254	192.168.1.254	ICMP	98	Echo (ping) request id=0x9404, seq=1/256, ttl=62 (reply in 4)
4	0.000204952	192.168.1.254	192.168.0.254	ICMP	98	Echo (ping) reply id=0x9404, seq=1/256, ttl=64 (request in 3)
5	5.011638737	fe:fd:00:00:04:01	fe:fd:00:00:03:02	ARP	42	Who has 192.168.1.1? Tell 192.168.1.254
6	5.011780373	fe:fd:00:00:03:02	fe:fd:00:00:04:01	ARP	42	192.168.1.1 is at fe:fd:00:00:03:02

Simnet1(router2-host2)

The same process as before but host2 has the default route configured and router2 has the route of router1. Therefore, the icmp message can come back.

5. Now, send a ping from router1 to 192.168.1.254. Find out why it does not work and propose two configurations of the routing table of host2 to fix the problem.

//from router1 ping to host2
ping -c 1 192.168.1.254

```
router1:~# ping -c 1 192.168.1.254
PING 192.168.1.254 (192.168.1.254) 56(84) bytes of data.
64 bytes from 192.168.1.254: icmp_seq=1 ttl=63 time=32.1 ms

--- 192.168.1.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 32.121/32.121/32.121/0.000 ms
```

The ping it does work because the default route for host2 is router2.

6. This exercise deals with the IPv4 Source Routing option. Capture on SimNet0, SimNet3 and SimNet2, and execute the following commands:

//from host1
ping -c 1 192.168.2.254

```
host1:~# ping -c 1 192.168.2.254
PING 192.168.2.254 (192.168.2.254) 56(84) bytes of data.
From 192.168.0.1 icmp_seq=1 Destination Net Unreachable

--- 192.168.2.254 ping statistics ---
1 packets transmitted, 0 received, +1 errors, 100% packet loss, time 0ms
```

//from host1
ping -c1 -r 192.168.0.1 10.0.0.3 192.168.2.254
ping -c1 10.0.0.3 192.168.2.254
ping -c1 -r 10.0.0.3 192.168.2.254

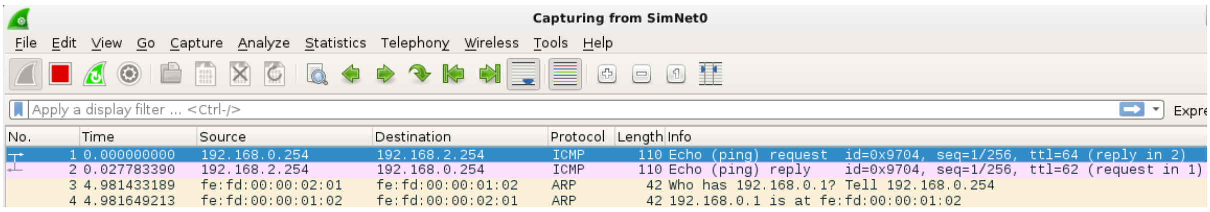
```
host1:~# ping -c1 -r 192.168.0.1 10.0.0.3 192.168.2.254
PING 192.168.2.254 (192.168.2.254) 56(124) bytes of data.
64 bytes from 192.168.2.254: icmp_seq=1 ttl=62 time=27.8 ms
SSRR:   10.0.0.3
        192.168.0.1


--- 192.168.2.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 27.895/27.895/27.895/0.000 ms
```

```
host1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.0.1              ether   FE:FD:00:00:01:02   C                     eth1
```

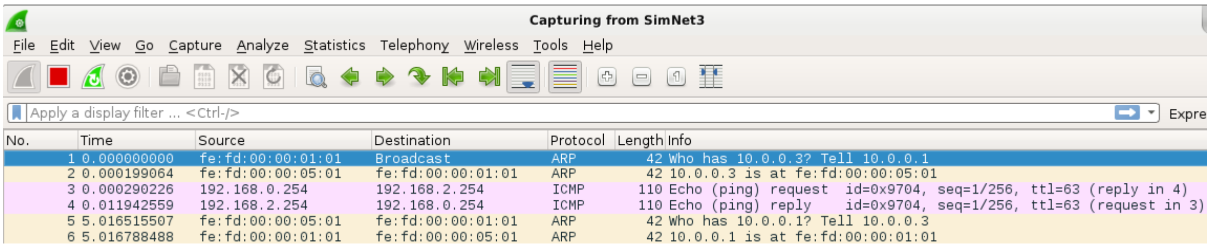
```
router1:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
10.0.0.3                 ether   FE:FD:00:00:05:01   C                     eth1
192.168.0.254            ether   FE:FD:00:00:02:01   C                     eth2
```

```
router3:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.2.254            ether   FE:FD:00:00:06:01   C                     eth2
10.0.0.1                 ether   FE:FD:00:00:01:01   C                     eth1
```

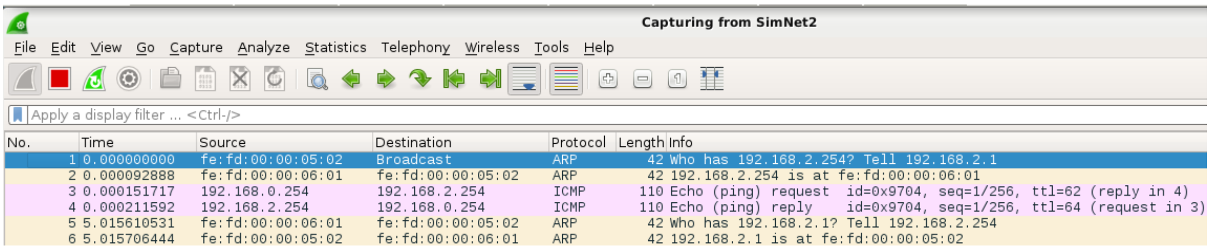
```
host3:~# arp -n
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.2.1              ether   FE:FD:00:00:05:02   C                     eth1
```



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.0.254	192.168.2.254	ICMP	110	Echo (ping) request id=0x9704, seq=1/256, ttl=64 (reply in 2)
2	0.027783390	192.168.2.254	192.168.0.254	ICMP	110	Echo (ping) reply id=0x9704, seq=1/256, ttl=62 (request in 1)
3	4.981433189	fe:fd:00:00:02:01	fe:fd:00:00:01:02	ARP	42	Who has 192.168.0.1? Tell 192.168.0.254
4	4.981649213	fe:fd:00:00:01:02	fe:fd:00:00:02:01	ARP	42	192.168.0.1 is at fe:fd:00:00:01:02



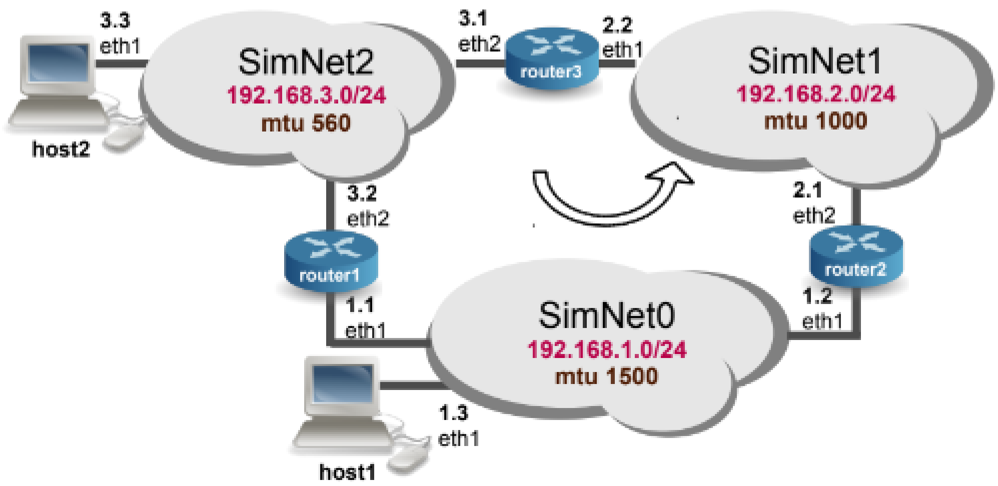
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 10.0.0.3? Tell 10.0.0.1
2	0.000199064	fe:fd:00:00:05:01	fe:fd:00:00:01:01	ARP	42	10.0.0.3 is at fe:fd:00:00:05:01
3	0.000290226	192.168.0.254	192.168.2.254	ICMP	110	Echo (ping) request id=0x9704, seq=1/256, ttl=63 (reply in 4)
4	0.011942559	192.168.2.254	192.168.0.254	ICMP	110	Echo (ping) reply id=0x9704, seq=1/256, ttl=63 (request in 3)
5	5.016515507	fe:fd:00:00:05:01	fe:fd:00:00:01:01	ARP	42	Who has 10.0.0.1? Tell 10.0.0.3
6	5.016788488	fe:fd:00:00:01:01	fe:fd:00:00:05:01	ARP	42	10.0.0.1 is at fe:fd:00:00:01:01



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:05:02	Broadcast	ARP	42	Who has 192.168.2.254? Tell 192.168.2.1
2	0.000092888	fe:fd:00:00:06:01	fe:fd:00:00:05:02	ARP	42	192.168.2.254 is at fe:fd:00:00:06:01
3	0.000151717	192.168.0.254	192.168.2.254	ICMP	110	Echo (ping) request id=0x9704, seq=1/256, ttl=62 (reply in 4)
4	0.000211592	192.168.2.254	192.168.0.254	ICMP	110	Echo (ping) reply id=0x9704, seq=1/256, ttl=64 (request in 3)
5	5.015610531	fe:fd:00:00:06:01	fe:fd:00:00:05:02	ARP	42	Who has 192.168.2.1? Tell 192.168.2.254
6	5.015706444	fe:fd:00:00:05:02	fe:fd:00:00:06:01	ARP	42	192.168.2.1 is at fe:fd:00:00:05:02

The first command tells each router to which IP redirect the ICMP message.
The first address is 192.168.0.1 (router1). When it is in router1 is sent to 10.0.0.3(router3)
and when it is in router3 is sent to 192.168.2.254 (host3). In each step the origin of the jump is described, therefore the ICMP message can be sent back to host1.

Exercise4– The goal of this exercise is to practice with the fragmentation of IP datagrams and with the operation of various ICMP messages when there are different error conditions. The network used for this exercise is shown in Figure 4. Type on your physical host the following command to start the scenario:
//from terminal
simctl icmp sh
start



//from router1
ifconfig eth1 192.168.1.1/24 mtu 1500
ifconfig eth2 192.168.3.2/24 mtu 560
route add default gw 192.168.1.2

//from router2
ifconfig eth1 192.168.1.2/24 mtu 1500
ifconfig eth2 192.168.2.1/24 mtu 1000
route add default gw 192.168.2.2

//from router3
ifconfig eth1 192.168.2.2/24 mtu 1000
ifconfig eth2 192.168.3.1/24 mtu 560
route add default gw 192.168.3.2

//from host1
ifconfig eth1 192.168.1.3/24 mtu 1500
route add default gw 192.168.1.2

//from host2
ifconfig eth1 192.168.3.3/24 mtu 560
route add default gw 192.168.3.2

```
router1:~# ifconfig
eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:01:01
          inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:101/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:29 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:2736 (2.6 KiB)  TX bytes:338 (338.0 B)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr fe:fd:00:00:01:02
          inet addr:192.168.3.2  Bcast:192.168.3.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:560  Metric:1
          RX packets:28 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:2680 (2.6 KiB)  TX bytes:338 (338.0 B)
          Interrupt:5
```

```
router2:~# ifconfig
eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:02:01
          inet addr:192.168.1.2  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:201/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:14 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:892 (892.0 B)  TX bytes:338 (338.0 B)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr fe:fd:00:00:02:02
          inet addr:192.168.2.1  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1000  Metric:1
          RX packets:10 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:688 (688.0 B)  TX bytes:338 (338.0 B)
          Interrupt:5
```

```
router3:~# ifconfig
eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:03:01
          inet addr:192.168.2.2  Bcast:192.168.2.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1000  Metric:1
          RX packets:7 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:452 (452.0 B)  TX bytes:338 (338.0 B)
          Interrupt:5

eth2      Link encap:Ethernet  HWaddr fe:fd:00:00:03:02
          inet addr:192.168.3.1  Bcast:192.168.3.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:560  Metric:1
          RX packets:12 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:720 (720.0 B)  TX bytes:338 (338.0 B)
          Interrupt:5
```

```
host1:~# ifconfig
eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:04:01
          inet addr:192.168.1.3  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::fcfd:ff:fe00:401/64 Scope:Link
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:1500  Metric:1
          RX packets:6 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:336 (336.0 B)  TX bytes:468 (468.0 B)
          Interrupt:5
```

```
host2:~# ifconfig
eth1      Link encap:Ethernet  HWaddr fe:fd:00:00:05:01
          inet addr:192.168.3.3  Bcast:192.168.3.255  Mask:255.255.255.0
          UP BROADCAST RUNNING PROMISC MULTICAST  MTU:560  Metric:1
          RX packets:5 errors:0 dropped:0 overruns:0 frame:0
          TX packets:6 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:280 (280.0 B)  TX bytes:468 (468.0 B)
          Interrupt:5
```

1. Find out which is the path that a packet will take going from host1 to host2, indicating the networks and routers that it will cross.

The path will be host1→ router2(simnet0)→ router3(simnet1)→ host2(simnet2)

2. Find out the path that a packet will take going from host2 to host1, indicating the networks and routers that it
will cross.

The path will be host2→ router1(simnet2)→ host1(simnet0)

3. Check your previous answers capturing traffic on SimNet0, SimNet1 and SimNet2 and executing the following pings:

//from host1
ping -c 1 192.168.3.3

```
host1:~# ping -c 1 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 56(84) bytes of data.
64 bytes from 192.168.3.3: icmp_seq=1 ttl=63 time=99.0 ms

--- 192.168.3.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 99.026/99.026/99.026/0.000 ms
```

Capturing from SimNet0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:04:01	Broadcast	ARP	42	Who has 192.168.1.2? Tell 192.168.1.3
2	0.000095057	fe:fd:00:00:02:01	fe:fd:00:00:04:01	ARP	42	192.168.1.2 is at fe:fd:00:00:02:01
3	0.000156866	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) request id=0x8704, seq=1/256, ttl=64 (reply in 6)
4	0.078613921	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.1.3? Tell 192.168.1.1
5	0.078725937	fe:fd:00:00:04:01	fe:fd:00:00:01:01	ARP	42	192.168.1.3 is at fe:fd:00:00:04:01
6	0.078794068	192.168.3.3	192.168.1.3	ICMP	98	Echo (ping) reply id=0x8704, seq=1/256, ttl=63 (request in 3)

Capturing from SimNet1

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:02:02	Broadcast	ARP	42	Who has 192.168.2.2? Tell 192.168.2.1
2	0.000062677	fe:fd:00:00:03:01	fe:fd:00:00:02:02	ARP	42	192.168.2.2 is at fe:fd:00:00:03:01
3	0.000104607	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) request id=0x8704, seq=1/256, ttl=63 (no response found!)

Capturing from SimNet2

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

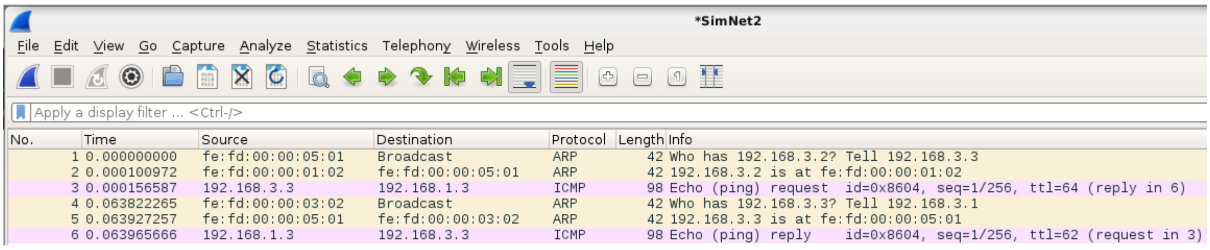
Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:03:02	Broadcast	ARP	42	Who has 192.168.3.3? Tell 192.168.3.1
2	0.000085657	fe:fd:00:00:05:01	fe:fd:00:00:03:02	ARP	42	192.168.3.3 is at fe:fd:00:00:05:01
3	0.000139369	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) request id=0x8704, seq=1/256, ttl=62 (reply in 6)
4	0.020451883	fe:fd:00:00:05:01	Broadcast	ARP	42	Who has 192.168.3.2? Tell 192.168.3.3
5	0.020554986	fe:fd:00:00:01:02	fe:fd:00:00:05:01	ARP	42	192.168.3.2 is at fe:fd:00:00:01:02
6	0.020592170	192.168.3.3	192.168.1.3	ICMP	98	Echo (ping) reply id=0x8704, seq=1/256, ttl=64 (request in 3)

From the wireshark it can be observed that the ICMP request follows the route described before and the ICMP reply too. The ICMP reply does not go through simnet1 and there is no fragmentation because the length didn't exceed 560, no need to fragment.

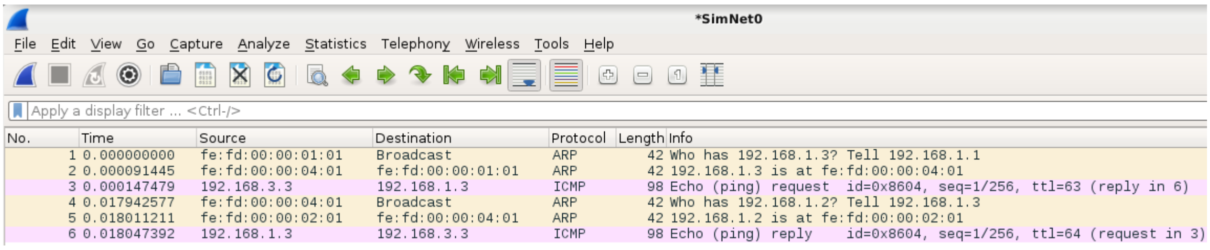
//from host2
ping -c 1 192.168.1.3

```
host2:~# ping -c 1 192.168.1.3
PING 192.168.1.3 (192.168.1.3) 56(84) bytes of data.
64 bytes from 192.168.1.3: icmp_seq=1 ttl=62 time=84.3 ms

--- 192.168.1.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 84.379/84.379/84.379/0.000 ms
```



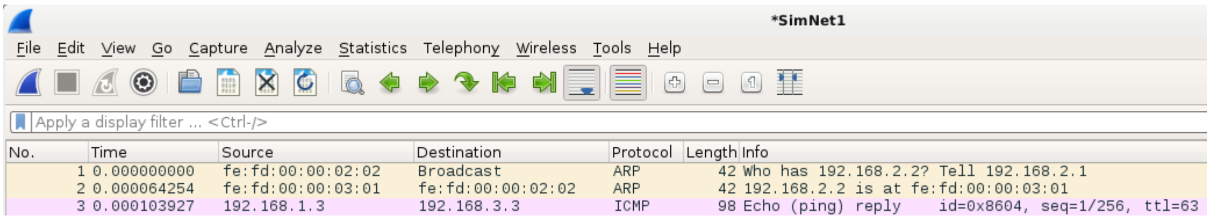
*SimNet2

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:05:01	Broadcast	ARP	42	Who has 192.168.3.2? Tell 192.168.3.3
2	0.000100972	fe:fd:00:00:01:02	fe:fd:00:00:05:01	ARP	42	192.168.3.2 is at fe:fd:00:00:01:02
3	0.000156587	192.168.3.3	192.168.1.3	ICMP	98	Echo (ping) request id=0x8604, seq=1/256, ttl=64 (reply in 6)
4	0.063822265	fe:fd:00:00:03:02	Broadcast	ARP	42	Who has 192.168.3.3? Tell 192.168.3.1
5	0.063927257	fe:fd:00:00:05:01	fe:fd:00:00:03:02	ARP	42	192.168.3.3 is at fe:fd:00:00:05:01
6	0.063965666	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) reply id=0x8604, seq=1/256, ttl=62 (request in 3)



*SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.1.3? Tell 192.168.1.1
2	0.000091445	fe:fd:00:00:04:01	fe:fd:00:00:01:01	ARP	42	192.168.1.3 is at fe:fd:00:00:04:01
3	0.000147479	192.168.3.3	192.168.1.3	ICMP	98	Echo (ping) request id=0x8604, seq=1/256, ttl=63 (reply in 6)
4	0.017942577	fe:fd:00:00:04:01	Broadcast	ARP	42	Who has 192.168.1.2? Tell 192.168.1.3
5	0.018011211	fe:fd:00:00:02:01	fe:fd:00:00:04:01	ARP	42	192.168.1.2 is at fe:fd:00:00:02:01
6	0.018047392	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) reply id=0x8604, seq=1/256, ttl=64 (request in 3)



*SimNet1

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:02:02	Broadcast	ARP	42	Who has 192.168.2.2? Tell 192.168.2.1
2	0.000064254	fe:fd:00:00:03:01	fe:fd:00:00:02:02	ARP	42	192.168.2.2 is at fe:fd:00:00:03:01
3	0.000103927	192.168.1.3	192.168.3.3	ICMP	98	Echo (ping) reply id=0x8604, seq=1/256, ttl=63

We can check that the ICMP request goes only through simnet2 and simnet0 and the ICMP reply through simnet0, simnet1 and simnet2. (the fram follow the route)

4. Determine the size of the IP packets containing the ICMP echo-request and echo-reply messages. Was it neces-sary to fragment any IP packet somewhere in the network?

size echo-request = 98
size echo-reply = 98

5. Comment the value of the DF flag found in the IP headers of captured packets. Which is the purpose of this flag?

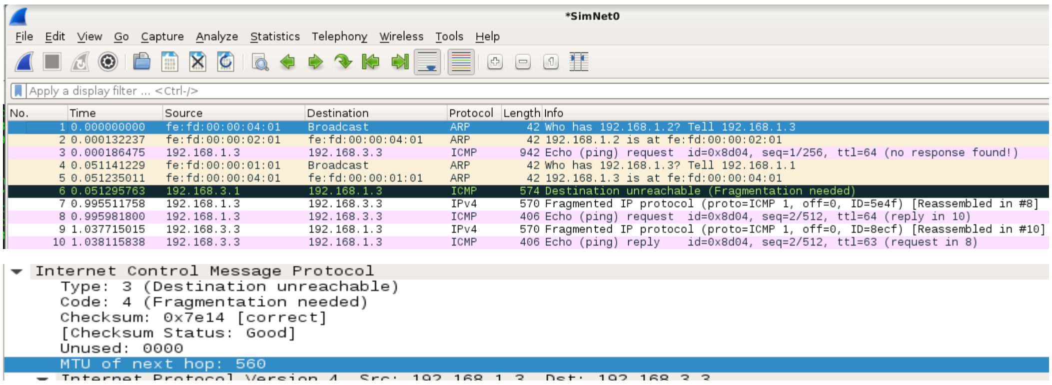
```
▾ Flags: 0x0000
    0... .... .... .... = Reserved bit: Not set
    .0.. .... .... .... = Don't fragment: Not set
    ..0. .... .... .... = More fragments: Not set
    ...0 0000 0000 0000 = Fragment offset: 0
  Time to live: 63
  Protocol: ICMP (1)
  Header checksum: 0x62b6 [validation disabled]
```

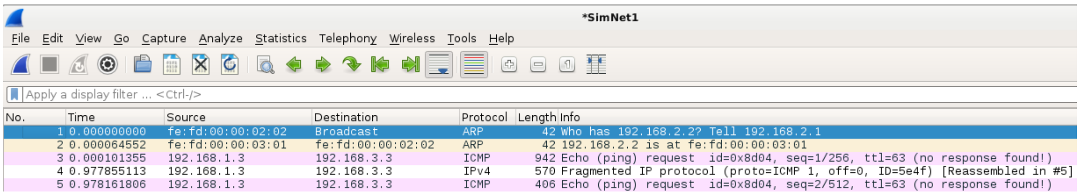
By default the DF flag is not set. DF means don't fragment.

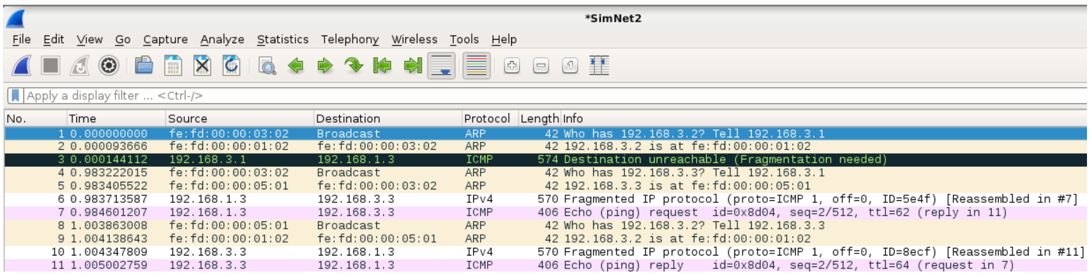
Now, capturing traffic on the three networks send two echo-request messages of 900 bytes of payload from host1 to host2. Note. Always delete the routing cache before sending the ping:

//from host1
ip route flush cache
ping -c 2 -s 900 192.168.3.3

```
host1:~# ping -c 1 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 56(84) bytes of data.
64 bytes from 192.168.3.3: icmp_seq=1 ttl=63 time=99.0 ms

--- 192.168.3.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 99.026/99.026/99.026/0.000 ms
host1:~# ip route flush cache
host1:~# ping -c 2 -s 900 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 900(928) bytes of data.
From 192.168.3.1 icmp_seq=1 Frag needed and DF set (mtu = 560)
908 bytes from 192.168.3.3: icmp_seq=2 ttl=63 time=42.8 ms

--- 192.168.3.3 ping statistics ---
2 packets transmitted, 1 received, +1 errors, 50% packet loss, time 1015ms
rtt min/avg/max/mdev = 42.895/42.895/42.895/0.000 ms
```







6. Which is the size of the first IP packet captured on SimNet0? Find the sizes of the headers of each protocol found in the frame that encapsulates this packet. Identify where the 900 bytes indicated in the ping command.

```
▾ Frame 3: 942 bytes on wire (7536 bits), 942 bytes captured (7536 bits) on interface 0
  ▾ Interface id: 0 (SimNet0)
      Interface name: SimNet0
    Encapsulation type: Ethernet (1)
    Arrival Time: Oct  2, 2022 20:18:00.452181139 CEST
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1664734680.452181139 seconds
    [Time delta from previous captured frame: 0.000054642 seconds]
    [Time delta from previous displayed frame: 0.000054642 seconds]
    [Time since reference or first frame: 0.000165550 seconds]
    Frame Number: 3
    Frame Length: 942 bytes (7536 bits)
    Capture Length: 942 bytes (7536 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:icmp:data]
    [Coloring Rule Name: ICMP]
    [Coloring Rule String: icmp || icmpv6]
▾ Ethernet II, Src: fe:fd:00:00:04:01 (fe:fd:00:00:04:01), Dst: fe:fd:00:00:02:01 (fe:fd:00:00:02:01)
  ▸ Destination: fe:fd:00:00:02:01 (fe:fd:00:00:02:01)
  ▸ Source: fe:fd:00:00:04:01 (fe:fd:00:00:04:01)
    Type: IPv4 (0x0800)
```

The first packet sent through simnet0 has a length of 942.

data.length = 892

ip.len == 20

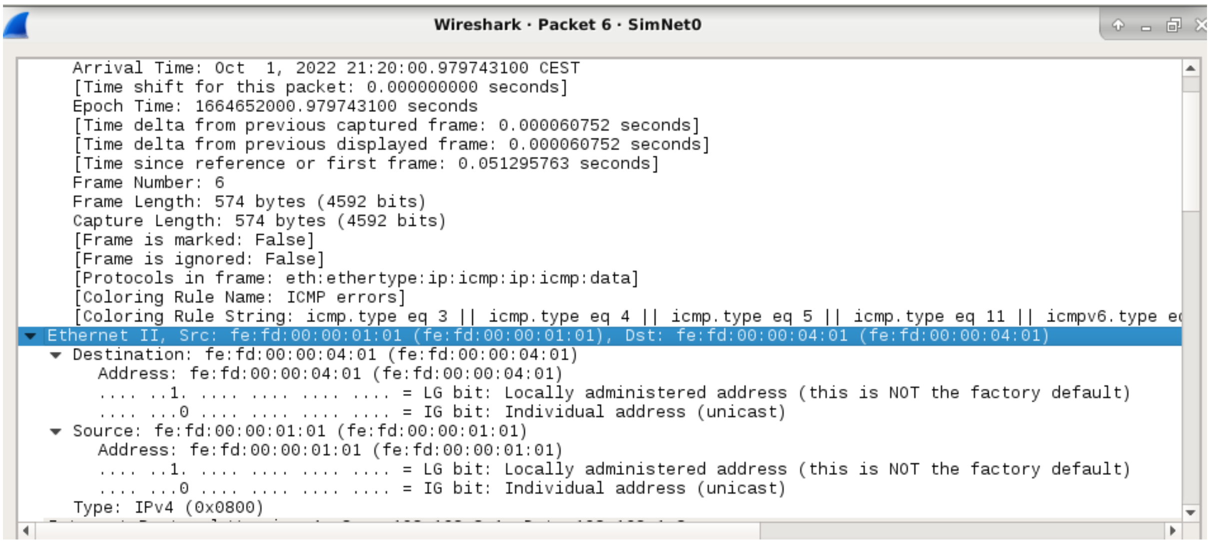
```
    Source: 192.168.1.3
    Destination: 192.168.3.3
▾ Internet Control Message Protocol
    Type: 8 (Echo (ping) request)
    Code: 0
    Checksum: 0x5d65 [correct]
    [Checksum Status: Good]
    Identifier (BE): 36100 (0x8d04)
    Identifier (LE): 1165 (0x048d)
    Sequence number (BE): 1 (0x0001)
    Sequence number (LE): 256 (0x0100)
  ▸ [No response seen]
    Timestamp from icmp data: Oct  1, 2022 21:20:00.907999000 CEST
    [Timestamp from icmp data (relative): 0.020634812 seconds]
  ▾ Data (892 bytes)
      Data: 08090a0b0c0d0e0f101112131415161718191a1b1c1d1e1f...
      [Length: 892]

0000  fe fd 00 00 02 01 fe fd  00 00 04 01 08 00 45 00   ········ ······E·
0010  03 a0 00 00 40 00 40 01  b2 06 c0 a8 01 03 c0 a8   ····@·@· ········
0020  03 03 08 00 5d 65 8d 04  00 01 e0 92 38 63 df da   ····]e·· ····8c··
0030  0d 00 08 09 0a 0b 0c 0d  0e 0f 10 11 12 13 14 15   ········ ········
0040  16 17 18 19 1a 1b 1c 1d  1e 1f 20 21 22 23 24 25   ········ ·· !"#$%
0050  26 27 28 29 2a 2b 2c 2d  2e 2f 30 31 32 33 34 35   &'()*+,- ./012345
0060  36 37 38 39 3a 3b 3c 3d  3e 3f 40 41 42 43 44 45   6789:;<= >?@ABCDE
0070  46 47 48 49 4a 4b 4c 4d  4e 4f 50 51 52 53 54 55   FGHIJKLM NOPQRSTU
0080  56 57 58 59 5a 5b 5c 5d  5e 5f 60 61 62 63 64 65   VWXYZ[\] ^_`abcde
0090  66 67 68 69 6a 6b 6c 6d  6e 6f 70 71 72 73 74 75   fghijklm nopqrstu
00a0  76 77 78 79 7a 7b 7c 7d  7e 7f 80 81 82 83 84 85   vwxyz{|} ~·······
00b0  86 87 88 89 8a 8b 8c 8d  8e 8f 90 91 92 93 94 95   ········ ········
00c0  96 97 98 99 9a 9b 9c 9d  9e 9f a0 a1 a2 a3 a4 a5   ········ ········
00d0  a6 a7 a8 a9 aa ab ac ad  ae af b0 b1 b2 b3 b4 b5   ········ ········

Data (data.data), 892 bytes     Packets: 10 · Displayed: 10 (100.0%) · Dropped: 0 (0.0%)     Profile: Default
```

The 892 bytes are in the data part.

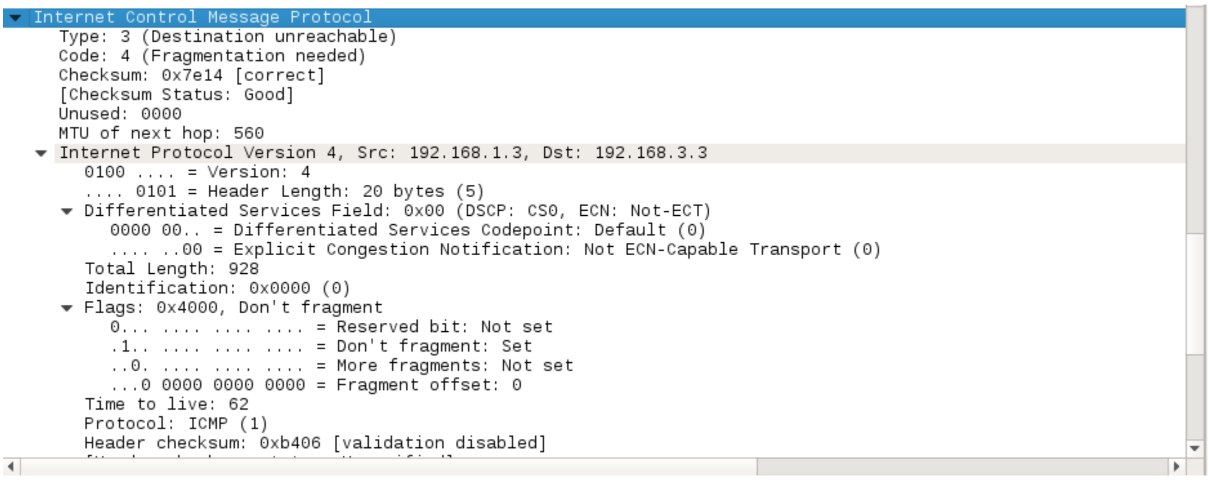
7. Checking the captures on SimNet1 and SimNet2 try to find out which is the path that the previous packet has followed.

We can see that the ICMP message in simnet0 starts with a TTL of 64 and in simnet1 is 63 and in simnet2 is 62. That means that it has gone through simnet0, simnet1 and simnet2 in this order.

8. Analyze the ICMP packet “Destination unreachable”. This ICMP message is telling us that the destination is unreachable, but why? Analyze the ICMP header of this message. Which is the IP packet that caused the error? Who is the sender of this ICMP message? Who is the recipient? Which path has followed this ICMP message from source to destination?



The sender is fe:fe:00:00:01:01 and the receiver is fe:fe:00:00:04:01.



MTU of router1 interface eth2 = 560

The destination is unreachable because the ICMP message has a size of 942 which is bigger than the Maximum Transmission Unit allowed by the interface eth2 of router1 and the ICMP message has the Don’t Fragment flag enabled.

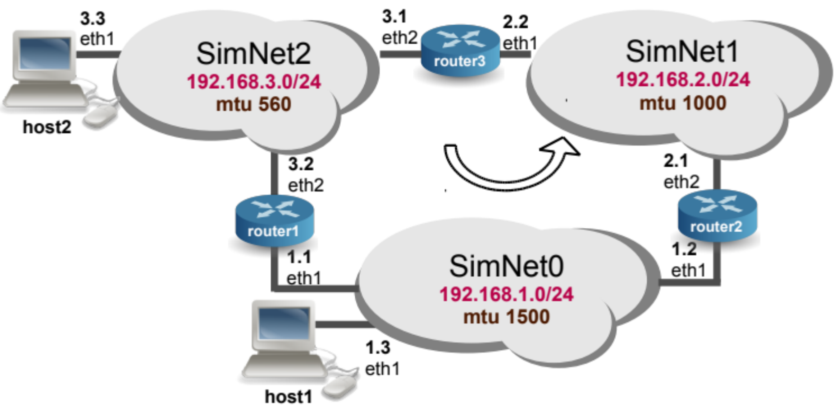
You should have observed that the first echo-request message with 900 bytes of payload has not reached the destination and, therefore, there was not an echo-reply. Now, you have to analyze the captures for the second echo-request.

9. Comment the values of the “Don’t Fragment” (DF), “ More Fragments” (MF) flags, “identification” (ID), “fragment offset” (FO) and the size of each IP packet related to this second ICMP message. Which is the purpose of MF, ID and FO? Try to correlate what you observe with the fact that we send an echo-request with 900 bytes of payload and that there is an IP network with an MTU of 560 bytes.

```
▼ Flags: 0x2000, More fragments
    0... .... .... .... = Reserved bit: Not set
    .0.. .... .... .... = Don't fragment: Not set
    ..1. .... .... .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment offset: 0
  Time to live: 64
```

```
▼ Flags: 0x2000, More fragments
    0... .... .... .... = Reserved bit: Not set
    .0.. .... .... .... = Don't fragment: Not set
    ..1. .... .... .... = More fragments: Set
    ...0 0000 0000 0000 = Fragment offset: 0
  Time to live: 63
  Protocol: ICMP (1)
```

10. Identify the path followed by the fragmented echo-request ICMP message from origin to destination and identify as well the path followed by echo-reply ICMP response message from origin to destination. Which machine made packet fragmentation?



The path will be host1→ router2(simnet0)→ router3(simnet1)→ host2(simnet2)
The path will be host2→ router1(simnet2)→ host1(simnet0)

Router3 makes the fragmentation.

11. Capture traffic on the three SimNet interfaces and send just one echo-request message from host1 to host2 with a payload of 900 bytes but with DF=0 (see with man the -M option of ping). Analyzing the captured traffic, determine where fragmentation is occurring.

```
-M hint
       Select Path MTU Discovery strategy.  hint may be either do (pro-
       hibit  fragmentation,  even local one), want (do PMTU discovery,
       fragment locally when packet size is large), or dont (do not set
       DF flag).
```

options:
do 1
want
dont 0

//fom host1
ip route flush cache
ping -M dont -c 1 -s 900 192.168.3.3

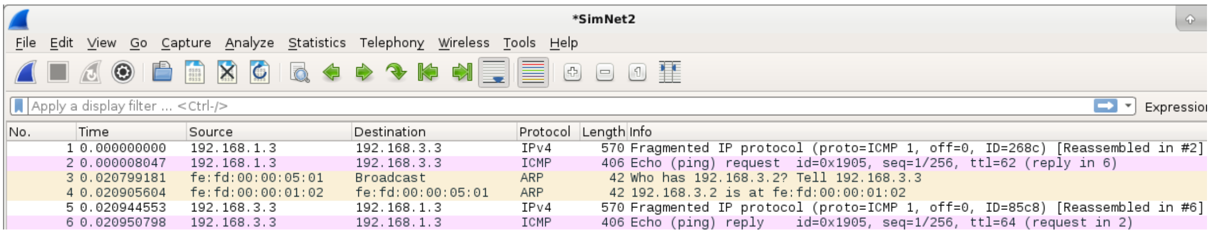
```
host1:~# ping -M dont -c 1 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 56(84) bytes of data.
64 bytes from 192.168.3.3: icmp_seq=1 ttl=63 time=103 ms

--- 192.168.3.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 103.418/103.418/103.418/0.000 ms
host1:~# ip route flush cache
host1:~# ping -M dont -c 1 -s 900 192.168.3.3
PING 192.168.3.3 (192.168.3.3) 900(928) bytes of data.
908 bytes from 192.168.3.3: icmp_seq=1 ttl=63 time=55.5 ms

--- 192.168.3.3 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 55.583/55.583/55.583/0.000 ms
```

*SimNet0

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	192.168.1.3	192.168.3.3	ICMP	942	Echo (ping) request id=0x1905, seq=1/256, ttl=64 (reply in 5)
2	0.055303966	fe:fd:00:00:01:01	Broadcast	ARP	42	Who has 192.168.1.3? Tell 192.168.1.1
3	0.055429979	fe:fd:00:00:04:01	fe:fd:00:00:01:01	ARP	42	192.168.1.3 is at fe:fd:00:00:04:01
4	0.055474213	192.168.3.3	192.168.1.3	IPv4	570	Fragmented IP protocol (proto=ICMP 1, off=0, ID=85c8) [Reassembled in #5]
5	0.055481063	192.168.3.3	192.168.1.3	ICMP	406	Echo (ping) reply id=0x1905, seq=1/256, ttl=63 (request in 1)
6	4.943885288	fe:fd:00:00:04:01	fe:fd:00:00:02:01	ARP	42	Who has 192.168.1.2? Tell 192.168.1.3
7	4.944006394	fe:fd:00:00:02:01	fe:fd:00:00:04:01	ARP	42	192.168.1.2 is at fe:fd:00:00:02:01

*SimNet1

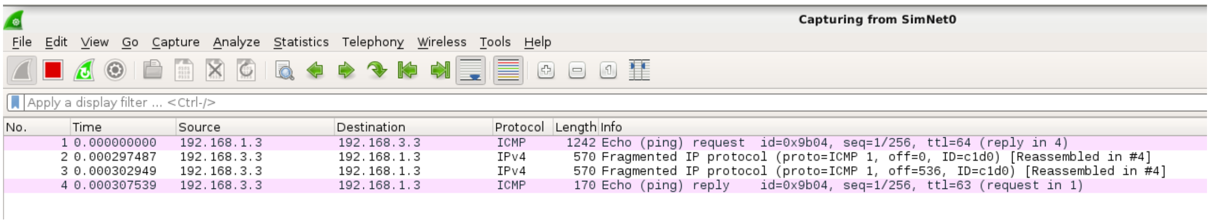
No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:02:02	Broadcast	ARP	42	Who has 192.168.2.2? Tell 192.168.2.1
2	0.000058701	fe:fd:00:00:03:01	fe:fd:00:00:02:02	ARP	42	192.168.2.2 is at fe:fd:00:00:03:01
3	0.000096215	192.168.1.3	192.168.3.3	ICMP	942	Echo (ping) request id=0x1905, seq=1/256, ttl=63 (no response found!)

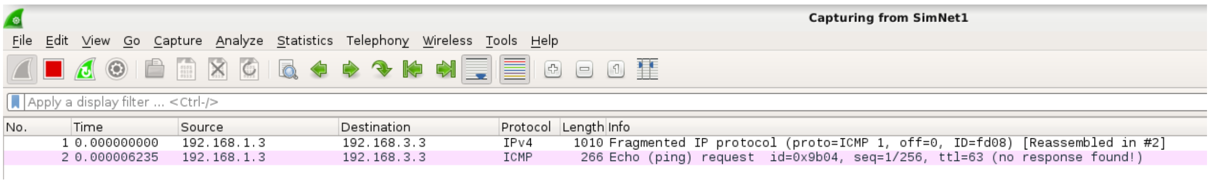


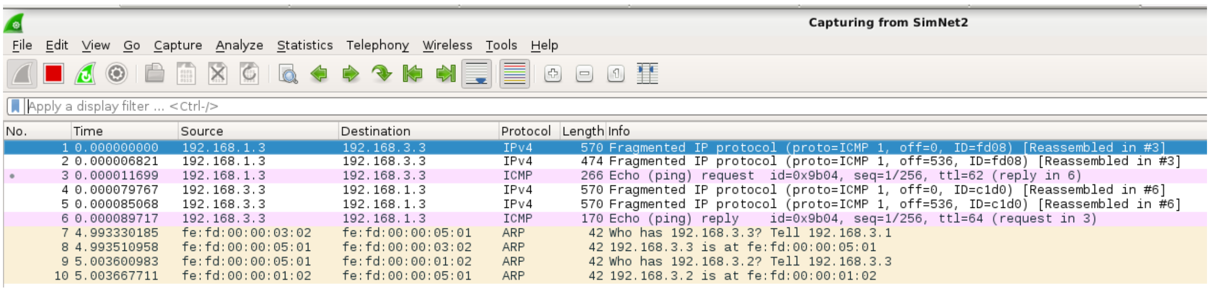
In SimNet 2 and SimNet0 there is a fragmentation but not in Simnet 1.

12. What happens if we send one echo-request message from host1 to host2 with a payload of 1200 bytes with DF=0?

//fom host1
ip route flush cache
ping -M dont -c 1 -s 1200 192.168.3.3







There is a fragmentation in all.

C. Time To Live (TTL) Exceeded

The goal of this test is to generate the error condition that causes the transmission of a Time To Live exceeded ICMP message. Recall that when an IP datagram arrives at a router, before being forwarded to destination, the router must do some processing:

- Decrement the Time To Live (TTL) field by one.
- Recalculate the “checksum” field (given that the TTL has changed).
- If the TTL reaches zero, the router throws away the packet and sends a Time To Live exceeded ICMP message to the sender of the IP datagram that generated the error.

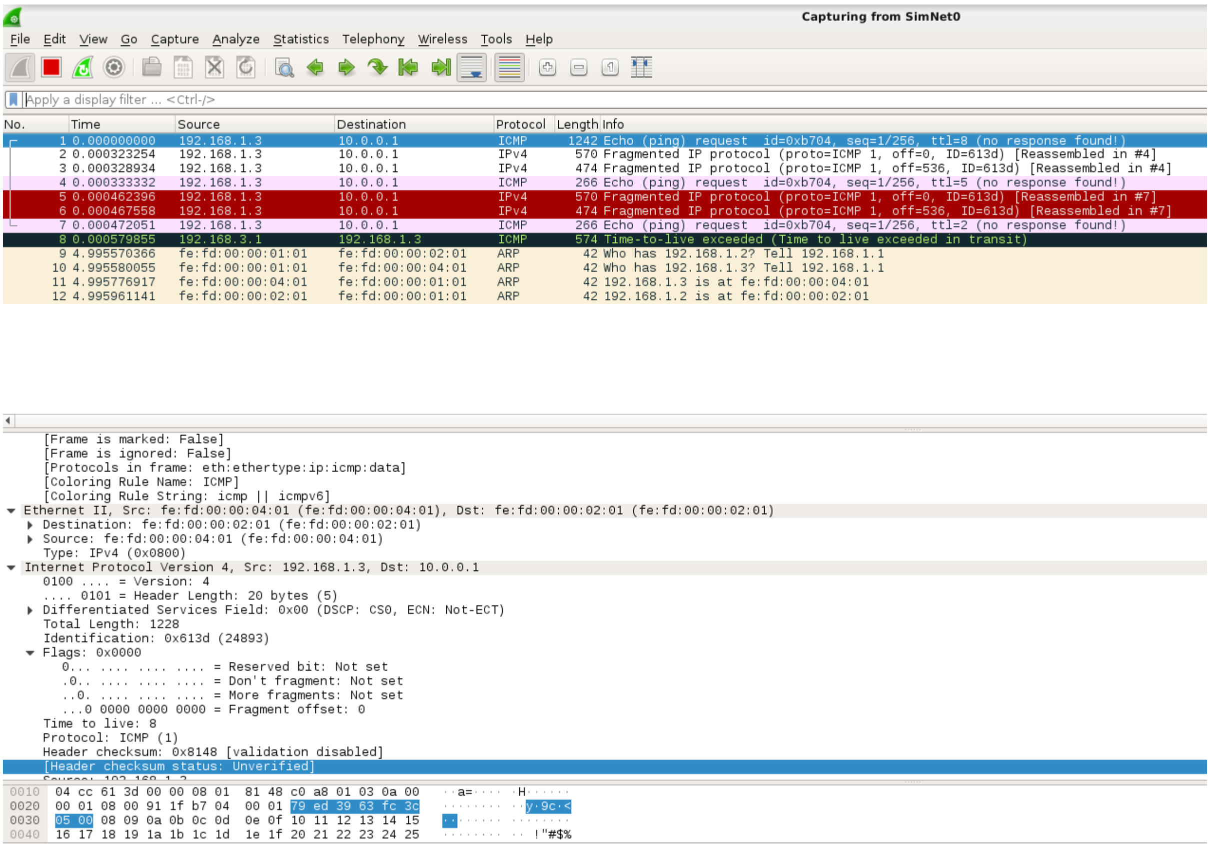
To conveniently check the operation described above, we will send an echo-request with TTL=8 (see -t option in the man page of ping) from host1 to the IP address 10.0.0.1. Before starting the practical test, answer theoretically the following questions:

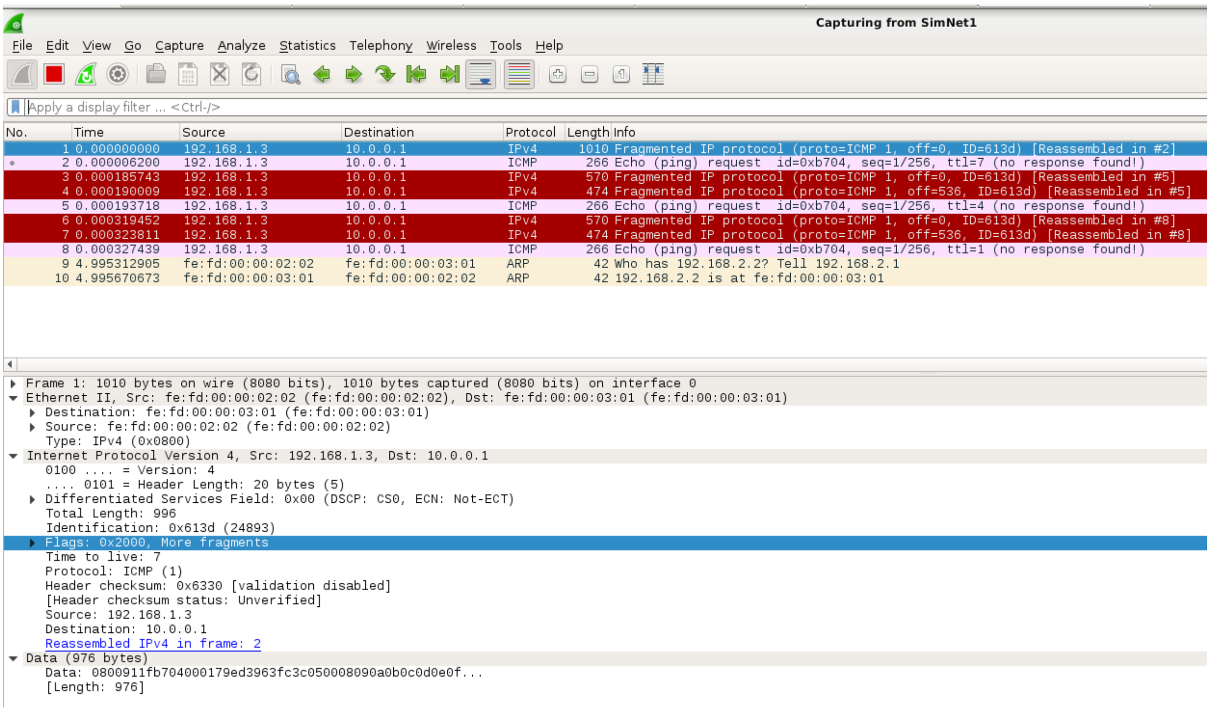
```
-t ttl Set the IP Time to Live.
```

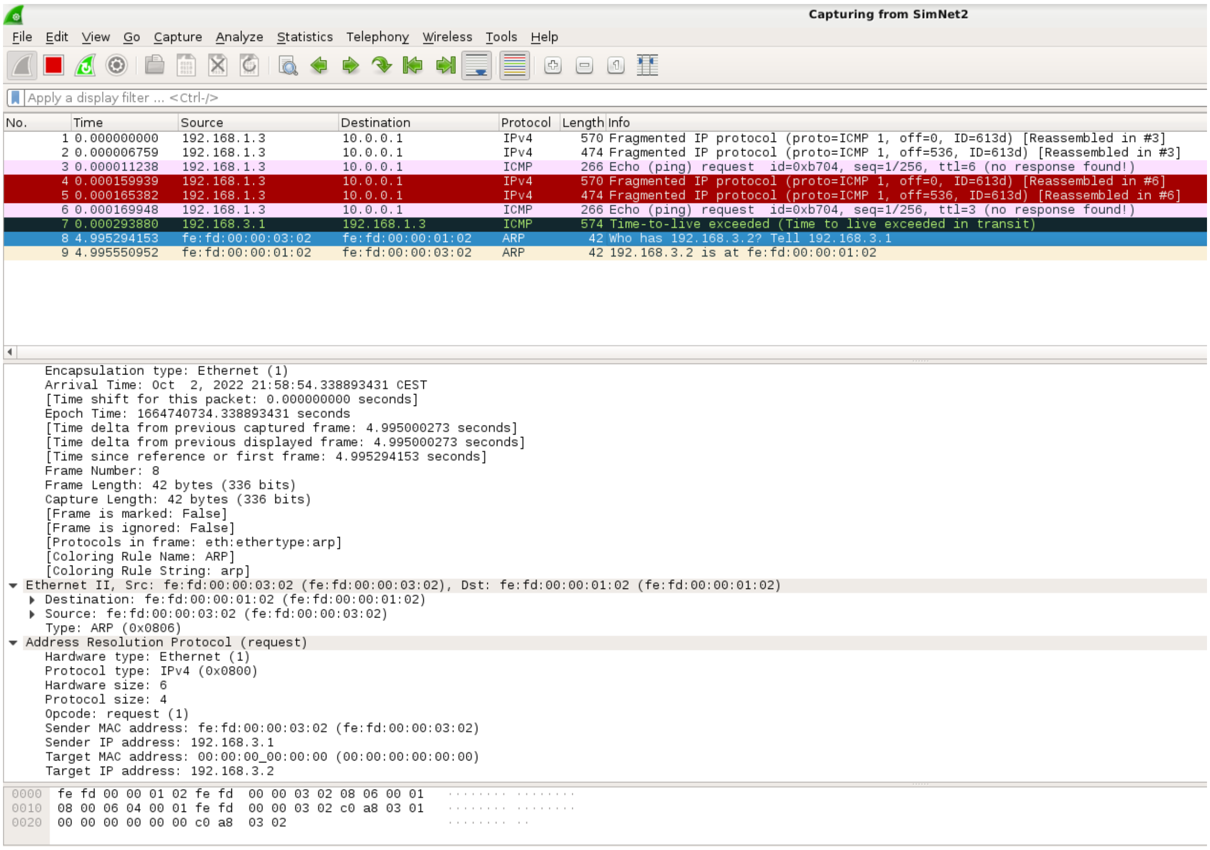
//fom host1

ip route flush cache

ping -M dont -c 1 -t 8 -s 1200 10.0.0.1







13. Given the configuration of the routers and hosts, which is the path that a datagram will follow in our network from host1 to 10.0.0.1? If TTL=8, which router will detect the error condition?

14. If the router that produces the error condition sends the Time To Live exceeded ICMP message to host1, which path will this packet follow? Which will be the source IP address of this datagram? Now, execute the ping command from host1.

Now, execute the ping command from host1.

15. Capture on the three SimNet interfaces and explain the captured traffic.

16. What happens if we set TTL=9?

//fom host1
ip route flush cache
ping -M dont -c 1 -t 9 -s 1200 10.0.0.1