P1: Switching Vlan

Exercise1– The goal of this practice is to test a LLC over Ethernet transmission with emulated equipment (hosts and switches). In particular, we will use the simple LLC1 client/server chat. Since you are root on the virtual machine you
can simply execute them as:

simctl switching-vlan sh
start
vms
get user
usuari: root
contrasenya: xxxx

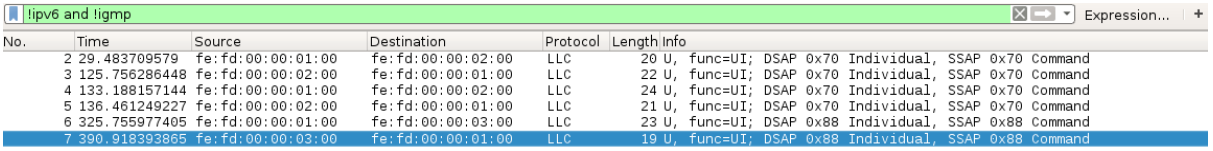
a.

we get the MAC of bob by using ifconfig
root@bob:~# server-chat-LLC1.py
(bob waits)
root@alice:~# send-frame-LLC1.py -d fe:fd:00:00:02:00
(we write a message as alice)
root@alice:~# server-chat-LLC1.py
(alice waits)
(we write a message as bob)
(conversations continues)

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000000	fe:fd:00:00:01:00	fe:fd:00:00:02:00	LLC	20	U, func=UI; DSAP 0x88 Individual, SSAP 0x88 Command
2	3.153574736	fe:fd:00:00:02:00	fe:fd:00:00:01:00	LLC	19	U, func=UI; DSAP 0x88 Individual, SSAP 0x88 Command
3	52.663365514	fe:fd:00:00:01:00	fe:fd:00:00:02:00	LLC	19	U, func=UI; DSAP 0x88 Individual, SSAP 0x88 Command
4	67.397100946	fe:fd:00:00:02:00	fe:fd:00:00:01:00	LLC	18	U, func=UI; DSAP 0x88 Individual, SSAP 0x88 Command
5	76.918200011	fe:fd:00:00:01:00	fe:fd:00:00:02:00	LLC	34	U, func=UI; DSAP 0x88 Individual, SSAP 0x88 Command

b.
r**oot@bob:~# server-chat-LLC1.py --sap 0x70**
(bob listens)
root@alice:~# send-frame-LLC1.py -d fe:fd:00:00:02:00 --ssap 0x70 --dsap 0x70
(we write a message as alice.0)
root@alice:~# server-chat-LLC1.py --sap 0x70
(alice.0 listens)
(we write a message as bob)

//Now we chat with carla
root@carla:~# server-chat-LLC1.py
(carla waits)
root@alice:~# send-frame-LLC1.py -d fe:fd:00:00:03:00
(we write a message as alice.1)
root@alice:~# server-chat-LLC1.py
(alice.1 listens)
(we write a message as carla)

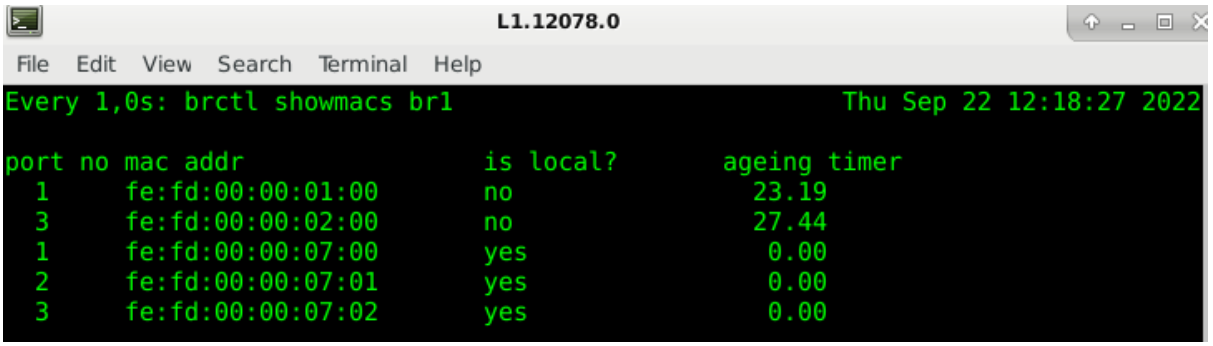


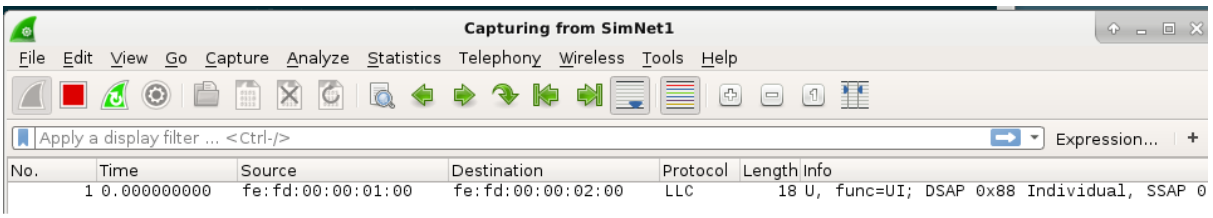
Exercise2– The goal of this practice is to study switching and the MAC learning process of switches using the emulated scenario. To do so, we are going to use a simple application called send-frame-LLC1.py that you can find in the directory /tmp. This application can be used to send LLC1 frames (use the -h option to see how it works).

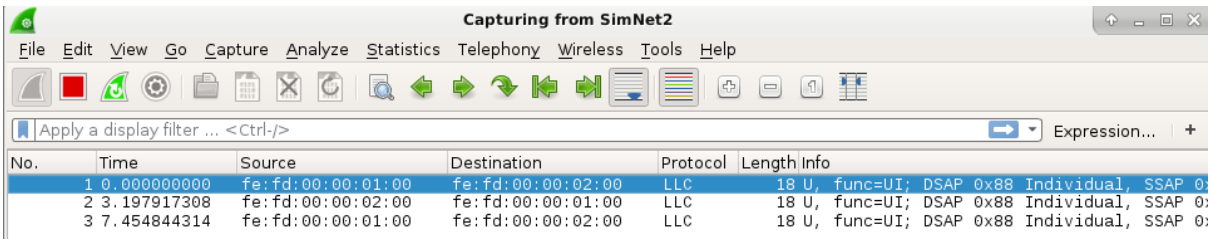
simctl switching-vlan sh
start
vms
get L1
get L2
get L3

root@L1:~# watch -n 5 brctl showmacs br1

port no	mac addr	is local?	ageing timer
1	fe:fd:00:00:07:00	yes	0.00
2	fe:fd:00:00:07:01	yes	0.00
3	fe:fd:00:00:07:02	yes	0.00







root@alice:~# send-frame-LLC1.py -d fe:fd:00:00:02:00
root@bob:~# send-frame-LLC1.py -d fe:fd:00:00:01:00
root@alice:~# send-frame-LLC1.py -d fe:fd:00:00:02:00

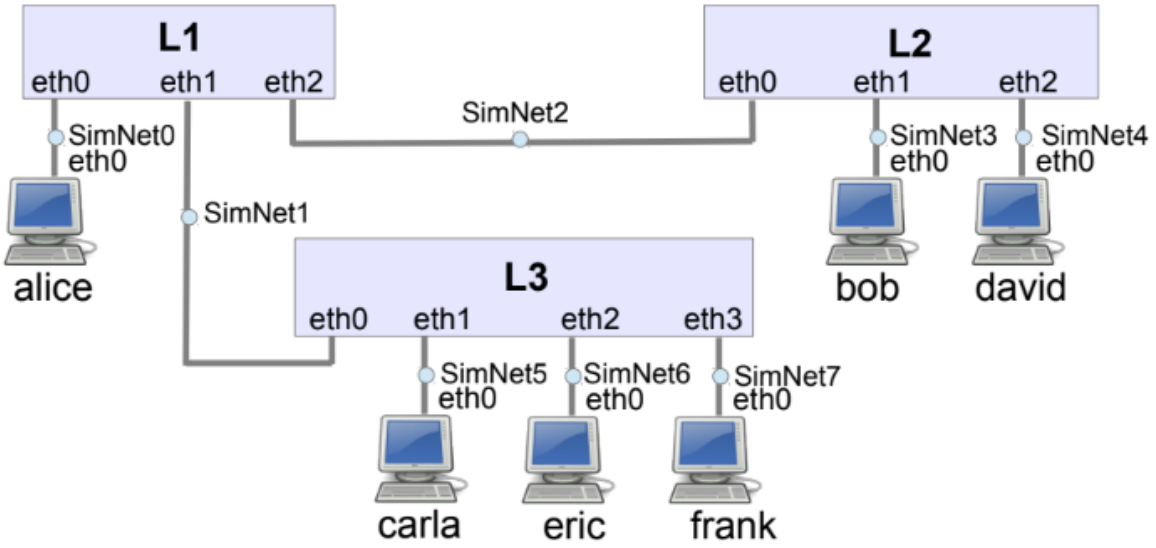


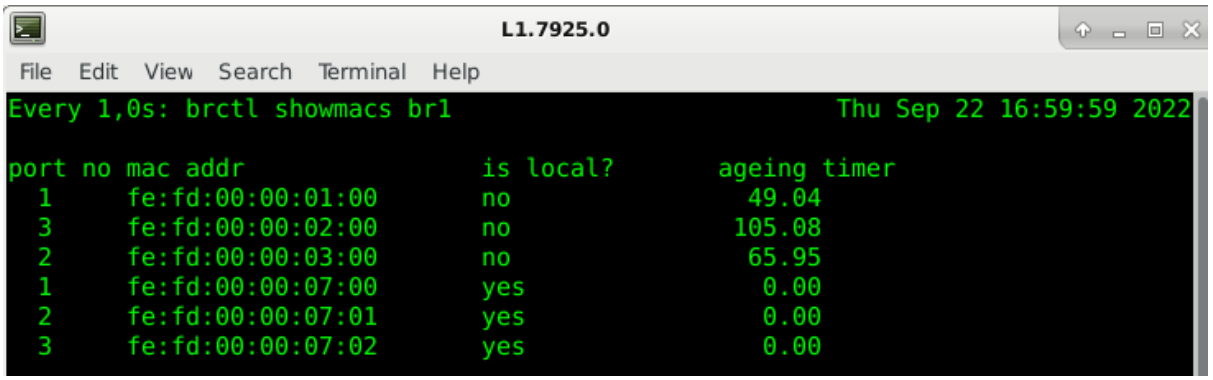
Figure 1.1: Scenario for testing Switching

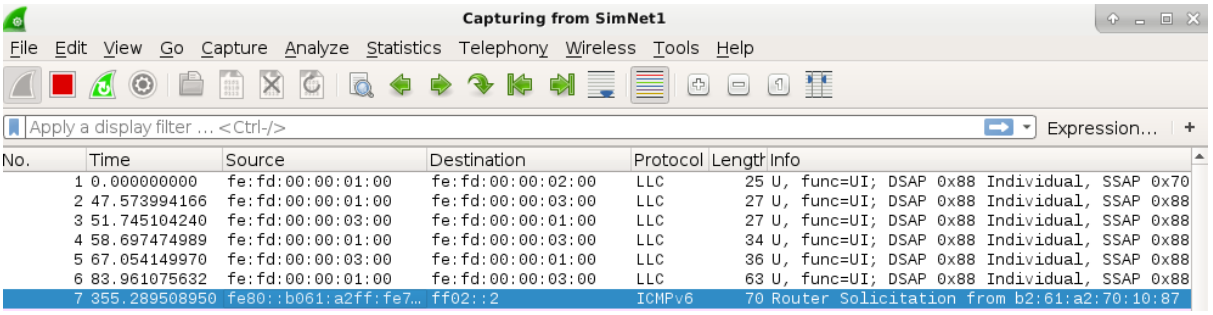
MAC learning:
The first message sent from alice to bob goes to simnet1 and simnet2. (The switch L1 learns that alice is on eth0)
Bob sends a message to alice that only goes to simnet2. (The switch L1 learns that bob is on eth2)
The next message that alice sends goes only to simnet2 because L1 already knows that bob is on eth2.
The table of MAC addresses refreshes in 1 minute.

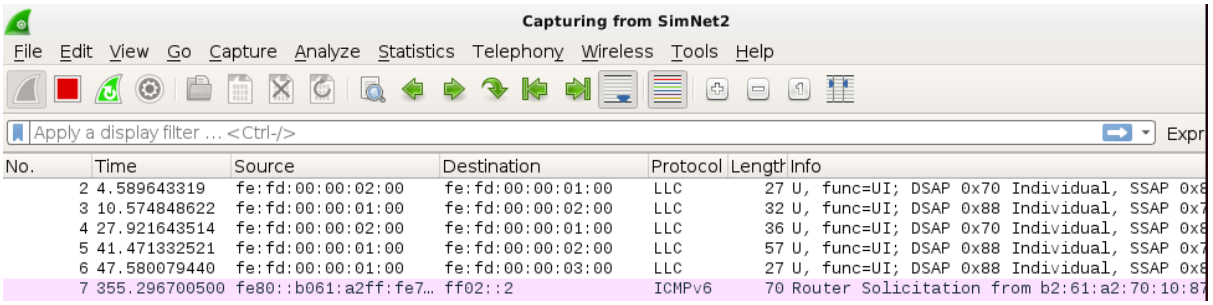
4.
root@L1:~# brctl setageing br1 600
(put 10min to the timer of the mac reset)
root@L1:~# watch -n 1 brctl showmacs br1

root@L1:~# brctl setageing br1 10
root@carla:~# server-chat-LLC1.py
root@alice:~# client-chat-LLC1.py -d fe:fd:00:00:03:00
(alice 0 sends a message to carla with ssap 0x80)

root@bob:~# server-chat-LLC1.py
root@alice:~# client-chat-LLC1.py -d fe:fd:00:00:02:00 --ssap 0x70
(alice 1 sends a message to bob tih ssap 0x70)







We can observe that the first message of each communication goes to both nets because the switch doesn't know in which net is the destination mac, then it does a broadcast. If we wait until the time limit, then the memory of the switch is reseted and it does broadcast again.

Exercise3– The goal of this practice is to study and test VLANs. In the first exercise you have to configure a port
level VLAN. In the second exercise, you have to configure switching between tagged and untagged frames. Finally, in
the last exercise, you have to configure switching between trunks.

1. In L3, modify the bridge configuration (removing the previous one) to create one VLAN for the ports 1 and 3 and another VLAN for the ports 2 and 4.
Explain your configuration. Do you need any VLAN id? Test your configuration with send-frame-LLC1.py

and the Ethernet broadcast address.

//from L3

ifconfig br3 down
brctl delbr br3
brctl addbr br3a
brctl addif br3a eth0
brctl addif br3a eth2
ifconfig br3a 192.168.1.1

brctl addbr br3b
brctl addif br3b eth1
brctl addif br3b eth3
ifconfig br3b 192.168.1.2

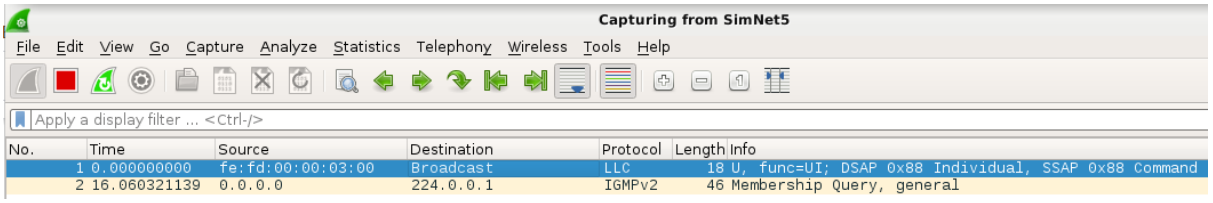
We separate the bridge in to bridges:
br3a has the ports eth0 and eth2
br3b has the ports eth1 and eth3

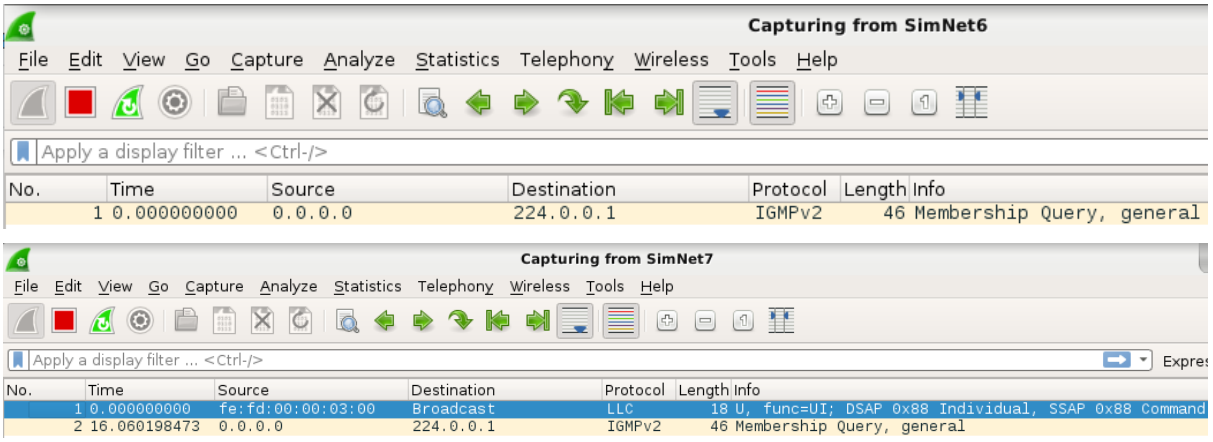
```
root@L3:~# brctl show
bridge name     bridge id               STP enabled     interfaces
br3a            8000.fefd00000900       no              eth0
                                                        eth2
br3b            8000.fefd00000901       no              eth1
                                                        eth3
```

We don't need VLAN ids because they are separated and we don't mix the traffic.

If we send a broadcast frame from carla:
root@carla:~# send-frame-LLC1.py -d ff:ff:ff:ff:ff:ff
only simnet5(carla's net) and simnet7(frank's net) receive the message, but simnet6(eric's net) doesn't receive the message.

¿¿¿Porque al enviar un mensaje en broadcast el server-chat.py no lo recive, pero al enviarse a la mac en específico sí lo lee???





2. Create a configuration in the switches in which all the hosts send untagged frames but:
• alice and carla belong to one data link network (use VLAN id 10).
• bob, david, eric and frank belong to another data link network.
Explain your configuration.
Capture traffic in the interfaces SimNet0 and SimNet1. Then, test your configuration appropriately using
the Ethernet broadcast address and send-frame-LLC1.py. Discuss the results and the format of the VLAN
frames.

//from L3
ifconfig br3 down
brctl delbr br3

brctl addbr br3a
vconfig add eth0 10
brctl addif br3a eth0.10
brctl addif br3a eth1
ifconfig eth0.10 up
ifconfig eth1 up
ifconfig br3a 192.168.1.1

brctl addbr br3b
brctl addif br3b eth0
brctl addif br3b eth2
brctl addif br3b eth3
ifconfig eth0 up
ifconfig eth2 up
ifconfig eth3 up
ifconfig br3b 192.168.2.1

```
                  root@L3:~# brctl show
bridge name     bridge id               STP enabled     interfaces
br3a            8000.fefd00000900       no              eth0.10
                                                        eth1
br3b            8000.fefd00000900       no              eth0
                                                        eth2
                                                        eth3
root@L3:~# br3b: port 3(eth3) entered forwarding state
                                                br3b: port 2(eth2) entered forwarding state
                br3b: port 1(eth0) entered forwarding state
```

//from L1

ifconfig br1 down
brctl delbr br1

vconfig add eth1 10
brctl addbr br1a
brctl addif br1a eth0
brctl addif br1a eth1.10
ifconfig eth0 up
ifconfig eth1.10 up
ifconfig br1a 192.168.3.1

brctl addbr br1b
brctl addif br1b eth1
brctl addif br1b eth2
ifconfig eth1 up
ifconfig eth2 up
ifconfig br1b 192.168.4.1

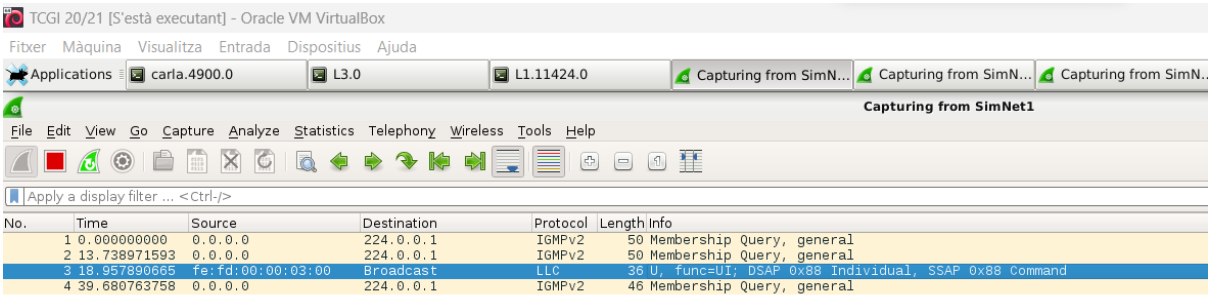
//configurations

```
root@L1:~# brctl show
bridge name     bridge id               STP enabled     interfaces
br1a            8000.fefd00000700       no              eth0
                                                        eth1.10
br1b            8000.fefd00000701       no              eth1
                                                        eth2
```

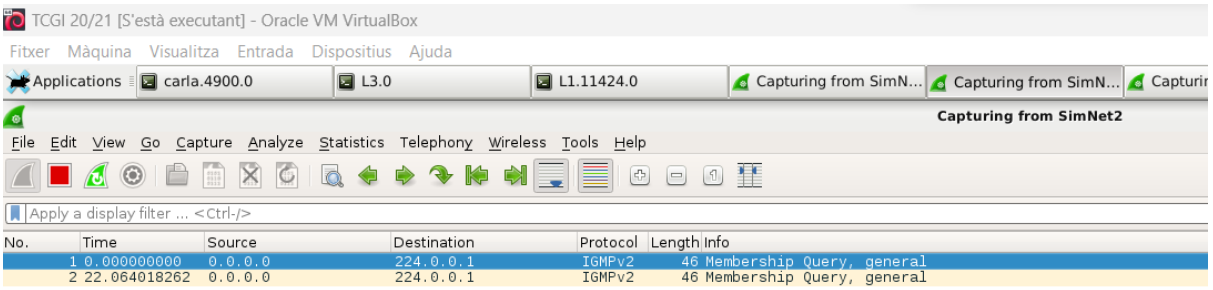
```
root@L3:~# brctl show
bridge name     bridge id               STP enabled     interfaces
br3a            8000.fefd00000900       no              eth0.10
                                                        eth1
br3b            8000.fefd00000900       no              eth0
                                                        eth2
                                                        eth3
```

//broadcast from carla

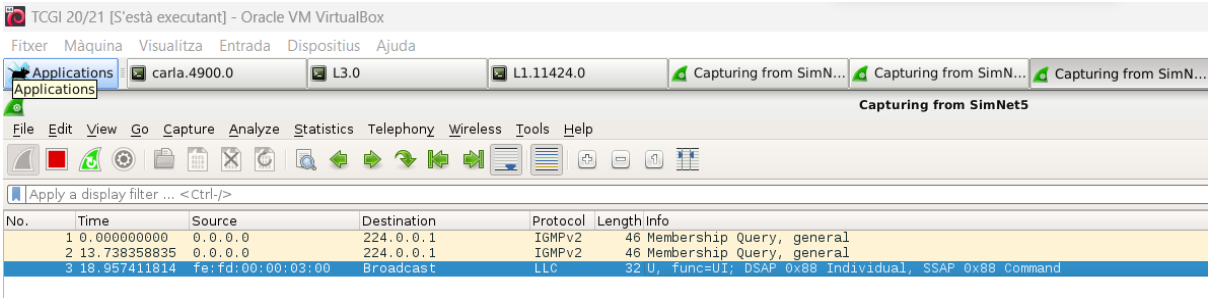
```
root@carla:~# send-frame-LLC1.py -d ff:ff:ff:ff:ff:ff
Using interface:  eth0
Using ssap:  0x88
Using dsap:  0x88
Type payload for LLC1 frame: hello soy carla
```



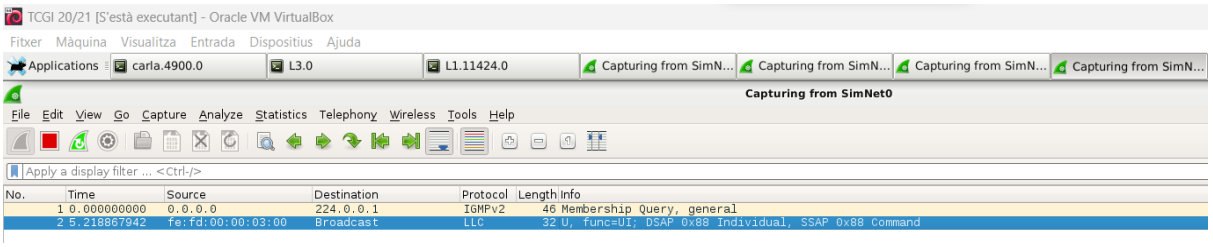
simnet 1 (alice-carla link)



simnet 2 (bob,david-erik,frank link)



simnet 5 (carla)

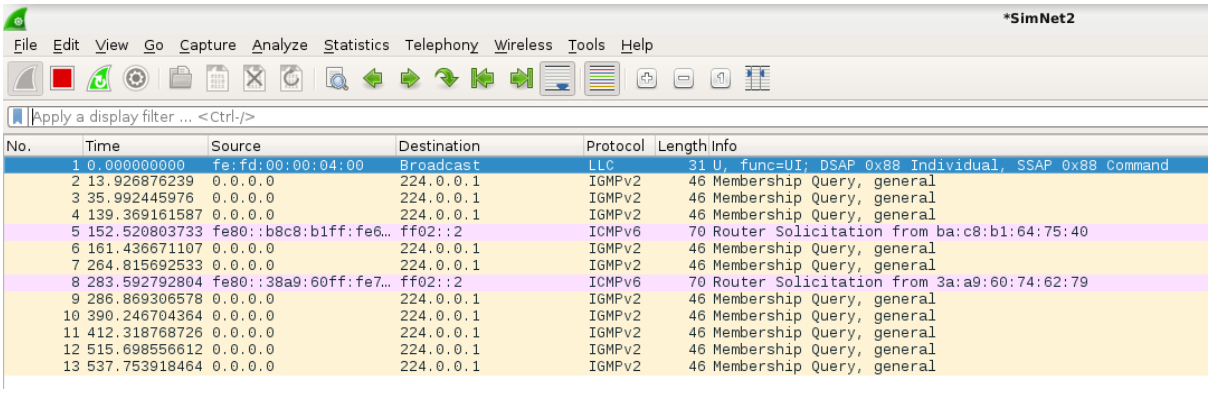


simnet 0 (alice)

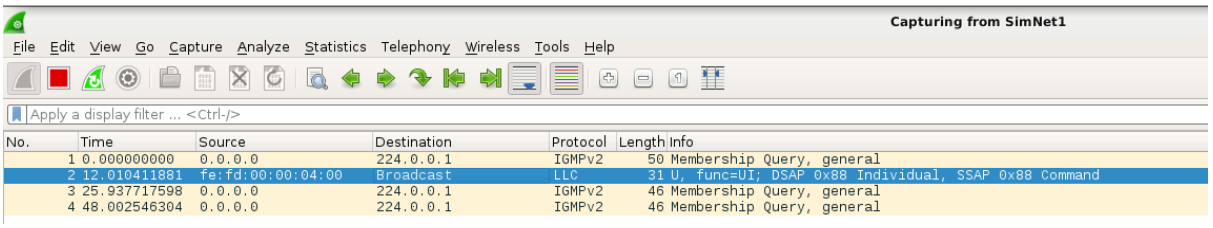
From wireshark we can see that the broadcast sent from carla is only received by alice.

//broadcast from david

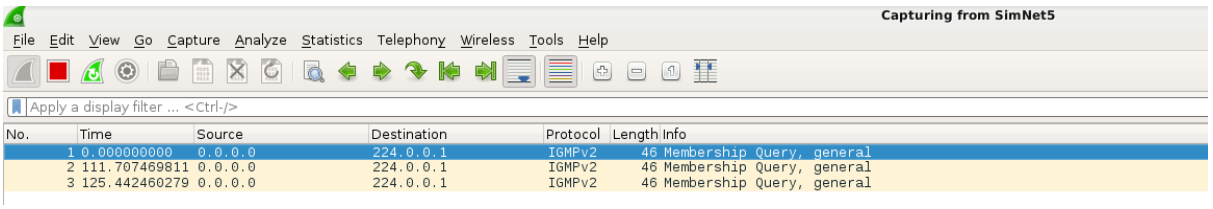
```
root@david:~# send-frame-LLC1.py -d ff:ff:ff:ff:ff:ff
Using interface:  eth0
Using ssap:  0x88
Using dsap:  0x88
Type payload for LLC1 frame: Hola soy david
.
Sent 1 packets.
```



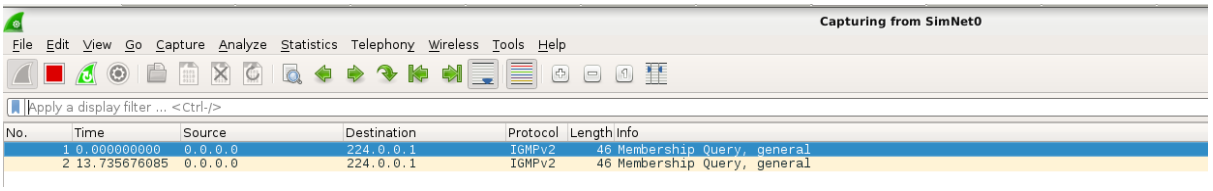
simnet 2 link l2-l1



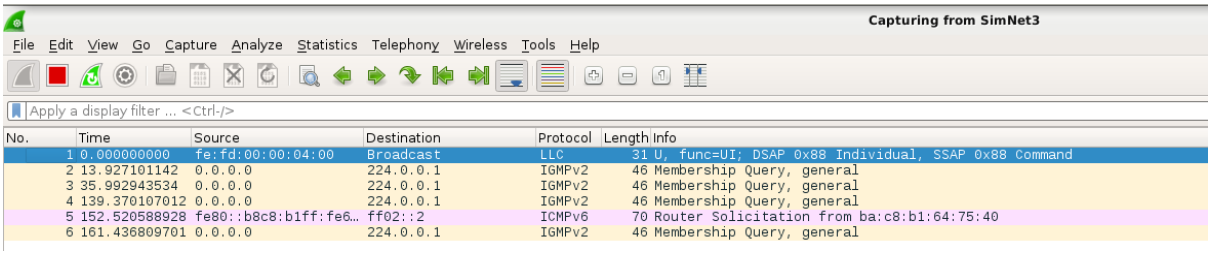
simnet 1 link l2-l3



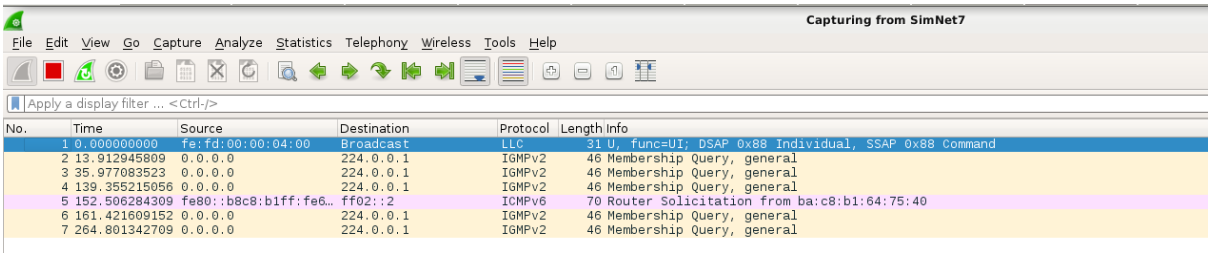
simnet 5 (carla)



simnet 0 (alice)



simnet 3 (bob)

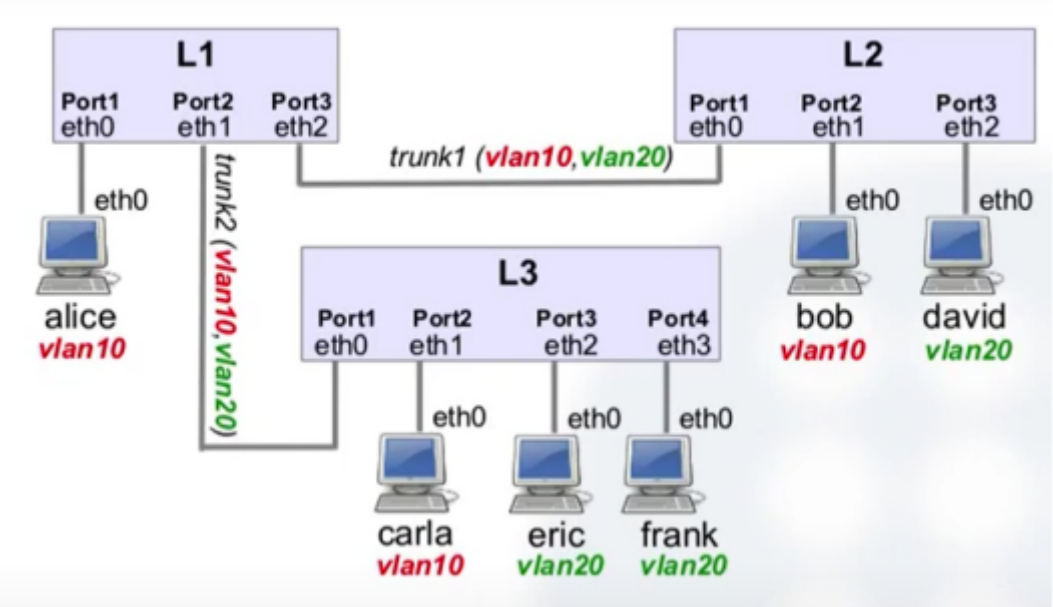


simnet 7 (frank)

If we make a broadcast from david the receivers are bob, erik and frank, but alice and carla don’t receive the broadcast trace.

3. Create a configuration in the switches in which all the hosts send untagged frames but they belong to the follow-
ing VLANs:

• VLAN 10: alice, bob and carla.
• VLAN 20: david, eric and frank.
Explain your configuration.
Capture traffic in the trunk interfaces (SimNet1 and SimNet2) and in any other SimNet interface that you consider useful. Then, test your configuration appropriately using send-frame-LLC1.py and the Ethernet broadcast address. Discuss the results and the format of the VLAN frames.



we will use the previous configuration so en L1 we will add vlan 10 to eth2

//from L1
ifconfig br1 down
brctl delbr br1

vconfig add eth1 10
vconfig add eth1 20
vconfig add eth2 10
vconfig add eth2 20

brctl addbr br1a
brctl addif br1a eth0
brctl addif br1a eth1.10
brctl addif br1a eth2.10
ifconfig eth0 up
ifconfig eth1.10 up
ifconfig eth2.10 up

ifconfig br1a 192.168.1.1/24

brctl addbr br1b
brctl addif br1b eth1.20
brctl addif br1b eth2.20
ifconfig eth1.20 up
ifconfig eth2.20 up
ifconfig br1b 192.168.2.1/24

//from L2
ifconfig br2 down
brctl delbr br2

vconfig add eth0 10
vconfig add eth0 20

brctl addbr br2a
brctl addif br2a eth0.10
brctl addif br2a eth1
ifconfig eth0.10 up
ifconfig eth1 up
ifconfig br2a 192.168.3.1/24

brctl addbr br2b
brctl addif br2b eth0.20
brctl addif br2b eth2
ifconfig eth0.20 up
ifconfig eth1 up
ifconfig br2b 192.168.4.1/24

//from L3
ifconfig br3 down
brctl delbr br3

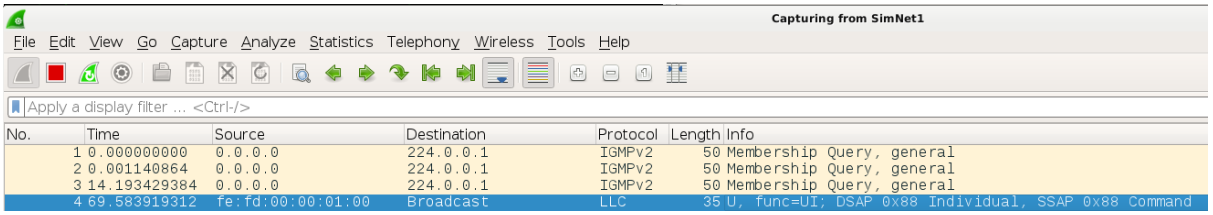
vconfig add eth0 10
vconfig add eth0 20

brctl addbr br3a
brctl addif br3a eth0.10
brctl addif br3a eth1
ifconfig eth0.10 up
ifconfig eth1 up
ifconfig br3a 192.168.5.1/24

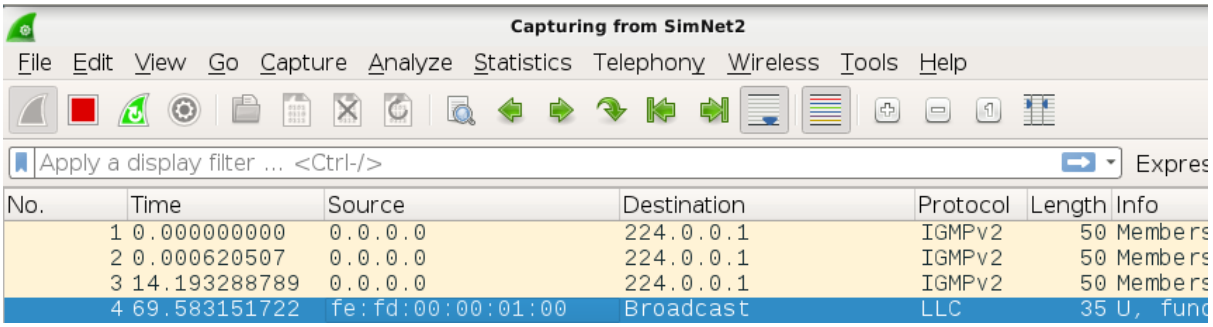
brctl addbr br3b
brctl addif br3b eth0.20
brctl addif br3b eth2
brctl addif br3b eth3

ifconfig eth0.20 up
ifconfig eth2 up
ifconfig eth3 up
ifconfig br3b 192.168.6.1/24

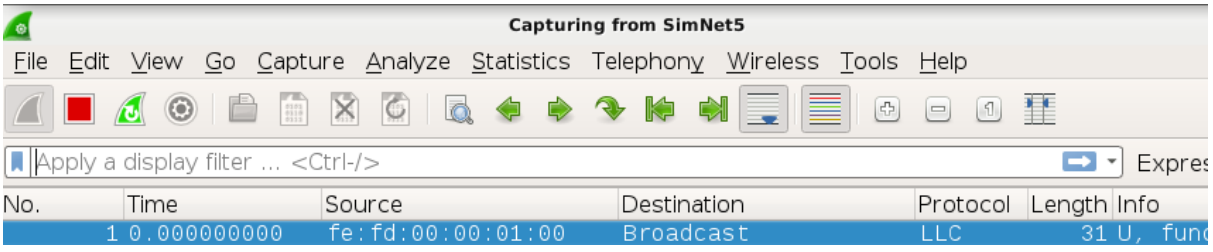
//broadcast from alice
send-frame-LLC1.py -d ff:ff:ff:ff:ff:ff



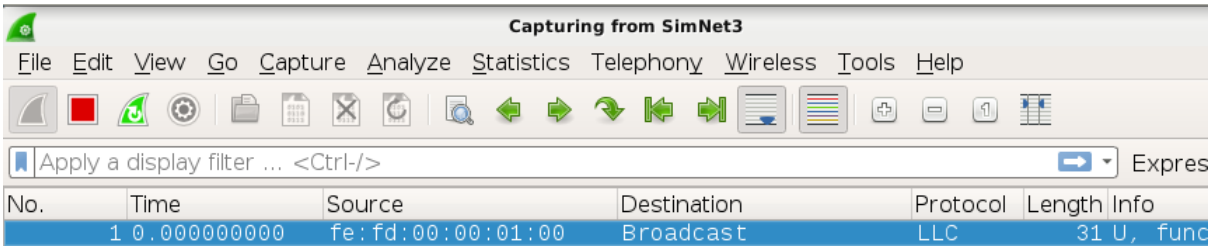
L1 to L3



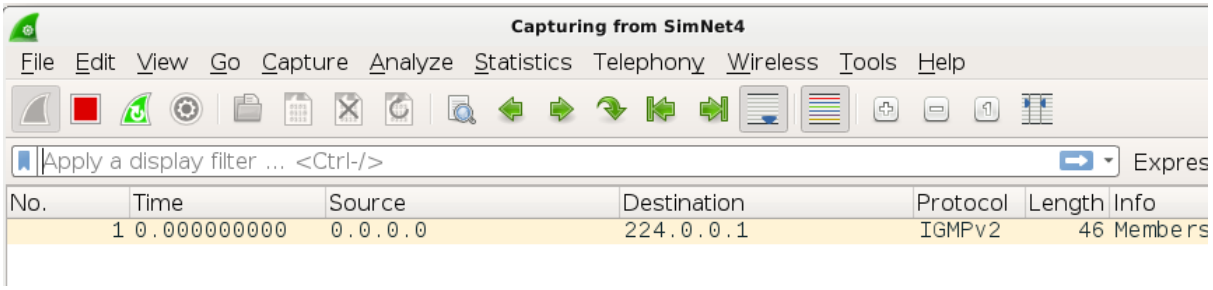
L1 to L2



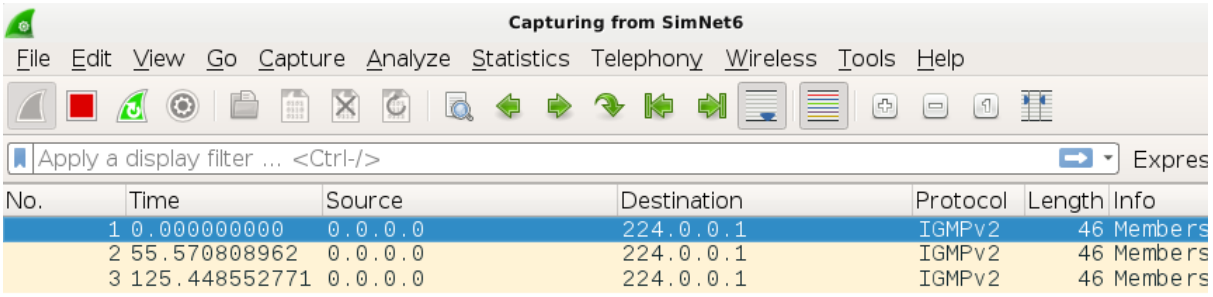
carla



bob



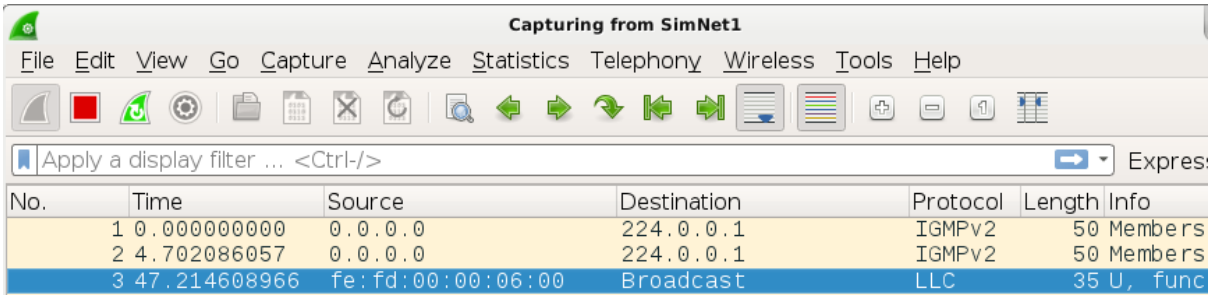
david



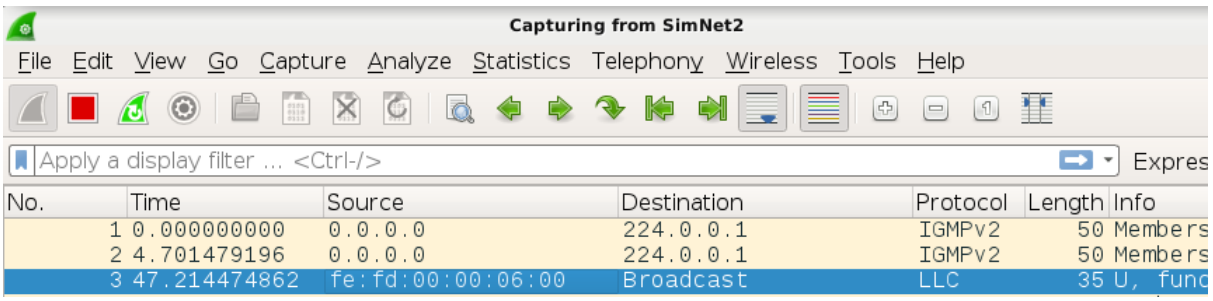
eric

Only bob and carla receive alice's broadcast.

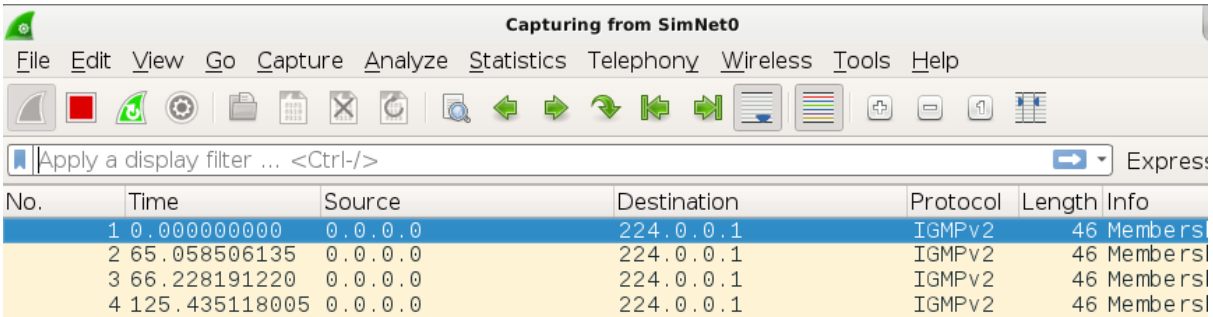
//broadcast from frank
send-frame-LLC1.py -d ff:ff:ff:ff:ff:ff



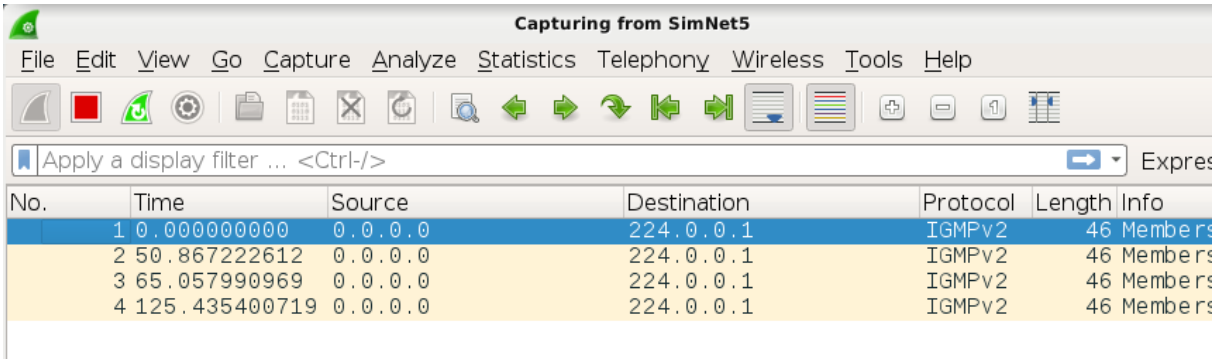
L1 to L3



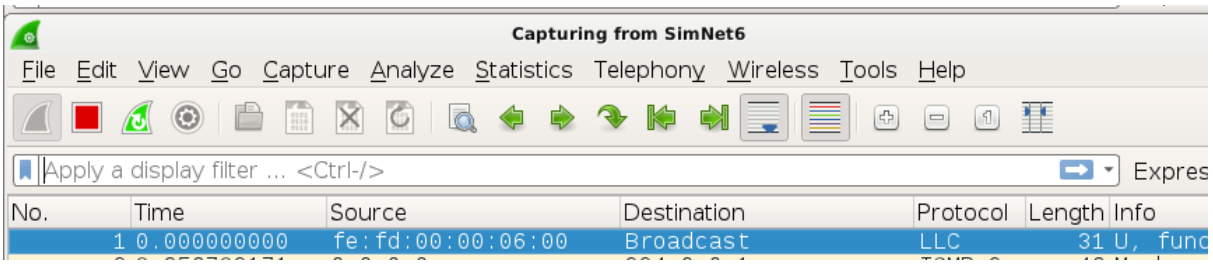
L1 to L2



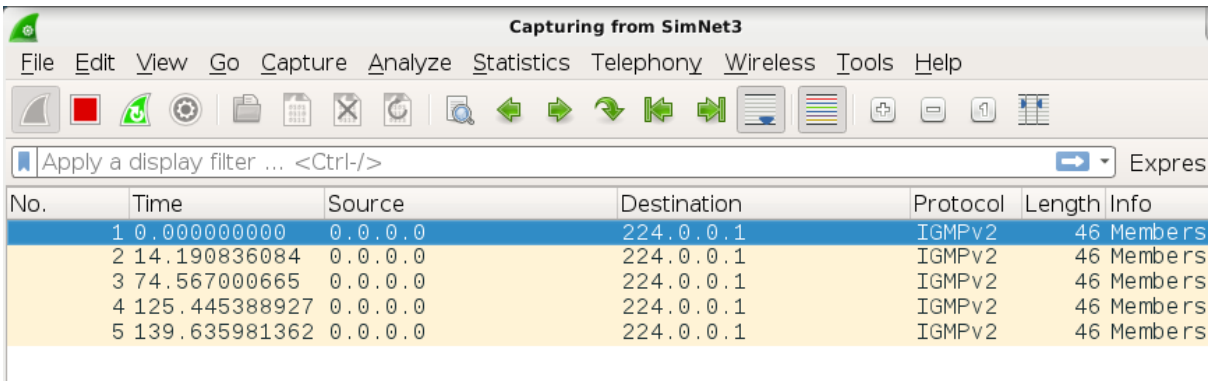
alice


Capturing from SimNet5
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expres
No. Time Source Destination Protocol Length Info
1 0.000000000 0.0.0.0 224.0.0.1 IGMPv2 46 Members
2 50.867222612 0.0.0.0 224.0.0.1 IGMPv2 46 Members
3 65.057990969 0.0.0.0 224.0.0.1 IGMPv2 46 Members
4 125.435400719 0.0.0.0 224.0.0.1 IGMPv2 46 Members

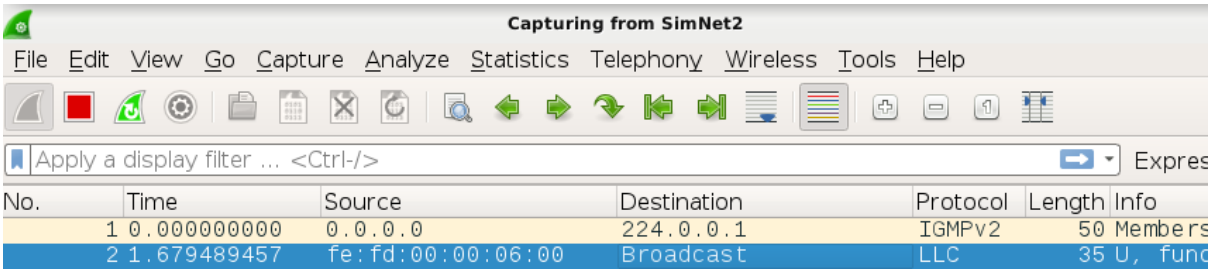
carla


Capturing from SimNet6
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expres
No. Time Source Destination Protocol Length Info
1 0.000000000 fe:fd:00:00:06:00 Broadcast LLC 31 U, func

eric



bob



david

If we send a broadcast from frank only david and eric receive it.