Université Libre de Bruxelles

Communication networks : protocols and architectures
ELEC-H417

Lab 1 - Dynamic routing

Group D :
Bienfait Alexandre 513930
Boistel Julien 440915
Humblet Raphaël 514085
Mutkowski Philippe 494470

Teacher :
Dricot Jean-Michel

Teaching Assistant :
Chassagne Aurélien

November 2023

Academic year 2023-2024

Contents

1 Configuration

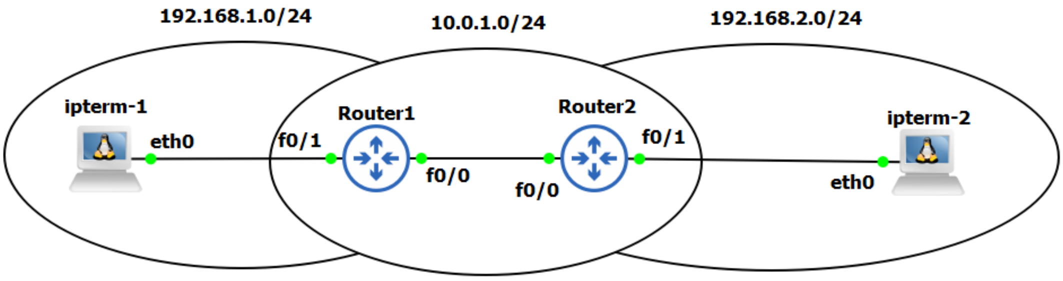


Figure 1: Initial topology

1.1 Ipterm configuration

The topology has been reproduced in the GNS3 window. To set up the ipterm 1 the two following commands have been used:

```
>>> ifconfig eth0 192.168.1.2 netmask 255.255.255.0 up
>>> route add default gateway 192.168.1.1
```

The first line sets up the eth0 port of the 192.168.1.2 host. This is the address of the ipterm 1.
The second line sets up the gateway of the ipterm1 to the router1 (192.168.1.1).

```
root@ipterm-1:~# netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.1.1     0.0.0.0         UG        0 0          0 eth0
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

Figure 2: ipterm-1 config

The same procedure is done for ipterm2 (with 192.168.2.1 for the router2 and 192.168.2.2 for ipterm2).

```
>>> ifconfig eth0 192.168.2.2 netmask 255.255.255.0
>>> route add default gateway 192.168.2.1
```

```
root@ipterm-2:~# netstat -nr
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.2.1     0.0.0.0         UG        0 0          0 eth0
192.168.2.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
```

Figure 3: ipterm-2 config

1.2 Router configuration

Concerning the router configuration, the following commands were run for the router1:

```
1 >>> configure terminal
2 >>> interface fastEthernet 0/0
3 >>> ip address 10.0.1.1 255.255.255.0
4 >>> no shutdown
5 >>> end
6 >>> write
```

1. Open the configuration terminal
2. Open the edition mode of the interface for the FastEthernet0/0 connection
3. Set up the IP address of the router in the subnet 10.0.1.0 (private subnet for the routers). With the mask 255.255.255.0.
4. Enable the interface
5. Leave the terminal
6. Save the result

The same has been done to map the 192.168.1.1 IP address to the FastEthernet0/1.

```
Router1#show ip int brief
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        10.0.1.1        YES NVRAM  up                    up
FastEthernet0/1        192.168.1.1     YES NVRAM  up                    up
FastEthernet1/0        unassigned      YES NVRAM  administratively down down
```

Figure 4: Router 1 config

The Figure 4 shows the IP interface of the router 1 after the configuration. It can be seen that the FastEthernet0/0 link is well-connected to the 10.0.1.1 port of the subnet, the same for FastEthernet0/1 and 192.168.1.1. Similar commands have been entered for Router 2.

```
Router2#show ip int br
Interface              IP-Address      OK? Method Status                Protocol
FastEthernet0/0        10.0.1.2        YES NVRAM  up                    up
FastEthernet0/1        192.168.2.1     YES NVRAM  up                    up
FastEthernet1/0        unassigned      YES NVRAM  administratively down down
```

Figure 5: Router 2 config

2 Mission 1: RipV2

Both routers were configured with RipV2 by adding the networks they are on to the configuration terminal.

```
>>> configure terminal
>>> router rip
>>> version 2
>>> network 10.0.1.0
>>> network 192.168.1.0 (resp 192.168.2.0)
>>> end
>>> write
```

Once both routers are configured with RipV2, a route is created between them (R in the figures).

The route R from Router 1 shows that it is possible to go to the subnet 192.168.2.0 by passing through the router 10.0.1.2 and redirect it with FastEthernet0/0.

```
Router1#show ip rip database
10.0.0.0/8    auto-summary
10.0.1.0/24    directly connected, FastEthernet0/0
192.168.1.0/24    auto-summary
192.168.1.0/24    directly connected, FastEthernet0/1
192.168.2.0/24    auto-summary
192.168.2.0/24
    [1] via 10.0.1.2, 00:00:17, FastEthernet0/0
Router1#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 1 subnets
C       10.0.1.0 is directly connected, FastEthernet0/0
C    192.168.1.0/24 is directly connected, FastEthernet0/1
R    192.168.2.0/24 [120/1] via 10.0.1.2, 00:00:21, FastEthernet0/0
```

Figure 6: Router 1 routes

```
Router2#show ip rip database
10.0.0.0/8    auto-summary
10.0.1.0/24    directly connected, FastEthernet0/0
192.168.1.0/24    auto-summary
192.168.1.0/24
    [1] via 10.0.1.1, 00:00:12, FastEthernet0/0
192.168.2.0/24    auto-summary
192.168.2.0/24    directly connected, FastEthernet0/1
Router2#show ip route
Codes: C - connected, S - static, R - RIP, M - mobile, B - BGP
       D - EIGRP, EX - EIGRP external, O - OSPF, IA - OSPF inter area
       N1 - OSPF NSSA external type 1, N2 - OSPF NSSA external type 2
       E1 - OSPF external type 1, E2 - OSPF external type 2
       i - IS-IS, su - IS-IS summary, L1 - IS-IS level-1, L2 - IS-IS level-2
       ia - IS-IS inter area, * - candidate default, U - per-user static route
       o - ODR, P - periodic downloaded static route

Gateway of last resort is not set

     10.0.0.0/24 is subnetted, 1 subnets
C       10.0.1.0 is directly connected, FastEthernet0/0
R    192.168.1.0/24 [120/1] via 10.0.1.1, 00:00:19, FastEthernet0/0
C    192.168.2.0/24 is directly connected, FastEthernet0/1
```

Figure 7: Router 2 routes

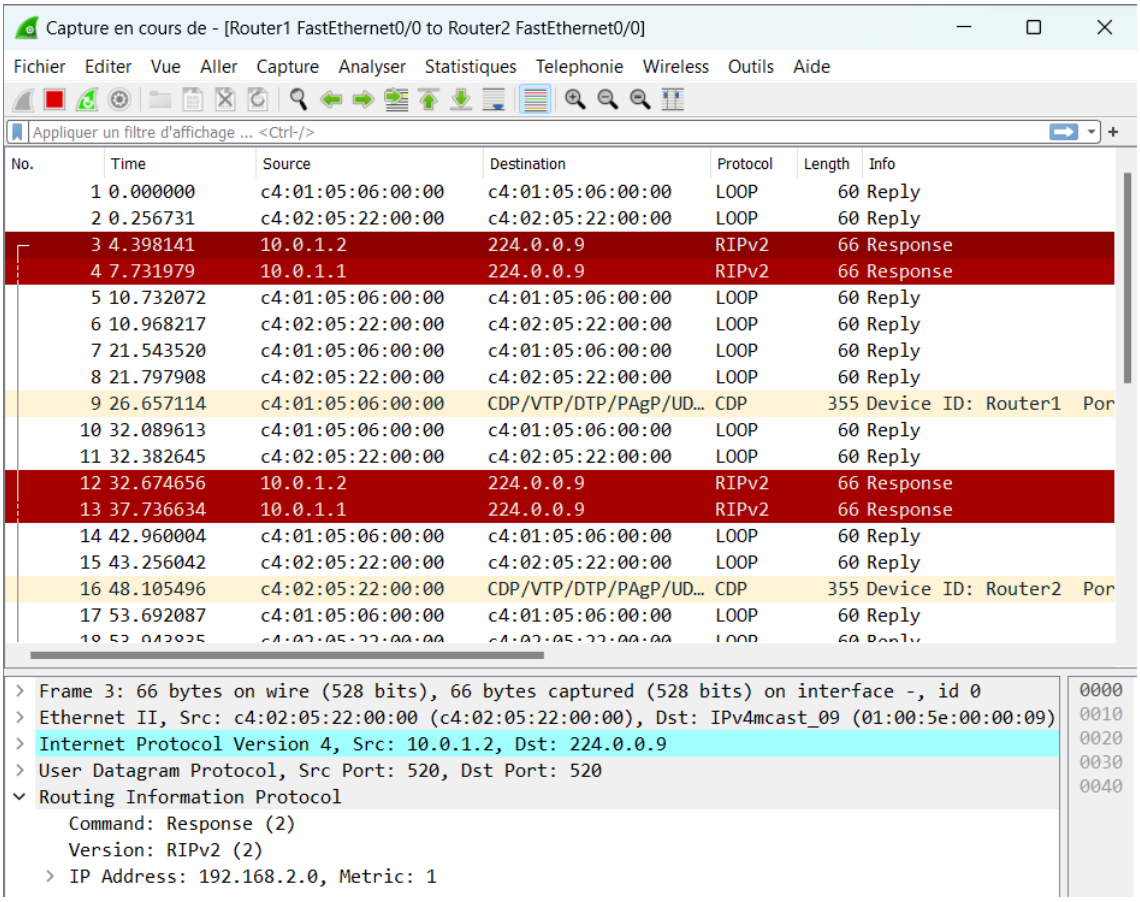


Figure 8: WireShark Capture between the two routers

The figure 8 confirms that the routing is functional. Indeed, the Source 10.0.1.2 (Router2) has a routing information protocol linked to the IP address 192.168.2.0 (the right subnet network). Meaning that the router can receive/send from two different subnets.

3 Mission 2: Network Discovery

3.1 Ping

To send from ipterm 1 to router 1, run the following command in the terminal of ipterm 1:

```
>>> ping 192.168.1.1 -c 3
```

It can be seen on Figure 9 that there are 3 requests and 3 replies as expected.

18	87.463093	192.168.1.2	192.168.1.1	ICMP	98	Echo (ping) request id=0x0043, seq=1/256, ttl=64 (reply in 19)
19	87.473108	192.168.1.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0043, seq=1/256, ttl=255 (request in 18)
20	88.459846	192.168.1.2	192.168.1.1	ICMP	98	Echo (ping) request id=0x0043, seq=2/512, ttl=64 (reply in 21)
21	88.463745	192.168.1.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0043, seq=2/512, ttl=255 (request in 20)
22	89.462202	192.168.1.2	192.168.1.1	ICMP	98	Echo (ping) request id=0x0043, seq=3/768, ttl=64 (reply in 23)
23	89.472150	192.168.1.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0043, seq=3/768, ttl=255 (request in 22)
24	92.865485	192.168.1.1	224.0.0.9	RIPv2	86	Response

Figure 9: Ping from ip1 to R1

To send from ipterm 1 to router 2, the command in the ipterm 1 terminal is the following:

```
>>> ping 192.168.2.1 -c 3
```

Ping from ip1 to R1

No.	Time	Source	Destination	Protocol	Length	Info
301	1048.377028	c4:01:05:06:00:00	c4:01:05:06:00:00	LOOP	60	Reply
302	1053.283684	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=1/256, ttl=63 (reply in 303)
303	1053.293685	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=1/256, ttl=255 (request in 302)
304	1054.287107	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=2/512, ttl=63 (reply in 305)
305	1054.297756	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=2/512, ttl=255 (request in 304)
306	1055.283959	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=3/768, ttl=63 (reply in 307)
307	1055.289339	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=3/768, ttl=255 (request in 306)
308	1057.941435	10.0.1.2	224.0.0.9	RIPv2	66	Response

Ping from R1 to R2

No.	Time	Source	Destination	Protocol	Length	Info
301	1048.377028	c4:01:05:06:00:00	c4:01:05:06:00:00	LOOP	60	Reply
302	1053.283684	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=1/256, ttl=63 (reply in 303)
303	1053.293685	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=1/256, ttl=255 (request in 302)
304	1054.287107	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=2/512, ttl=63 (reply in 305)
305	1054.297756	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=2/512, ttl=255 (request in 304)
306	1055.283959	192.168.1.2	192.168.2.1	ICMP	98	Echo (ping) request id=0x0044, seq=3/768, ttl=63 (reply in 307)
307	1055.289339	192.168.2.1	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0044, seq=3/768, ttl=255 (request in 306)
308	1057.941435	10.0.1.2	224.0.0.9	RIPv2	66	Response

Figure 10: Capture from the link between ip1 and R2 (through R1)

First, the ipterm 1 sends the data to the router 1 (192.168.1.1), this is what Figure 10 shows.

Then, the router 1 will resend it to the router 2 (192.168.2.1) but it can be seen that the source shown in the Figure 10 is the ipterm 1 (192.168.1.2) and not the router 1 (192.168.1.1). This is because the router 1 is just a "transitive host".

To send from ipterm 1 to ipterm 2 write this command in the console of ipterm 1:

```
>>> ping 192.168.2.2 -c 3
```

ping from ip1 to r1

No.	Time	Source	Destination	Protocol	Length	Info
208	1010.198597	c4:01:05:06:00:01	c4:01:05:06:00:01	LOOP	60	Reply
209	1012.533053	192.168.1.1	224.0.0.9	RIPv2	86	Response
210	1020.842414	c4:01:05:06:00:01	c4:01:05:06:00:01	LOOP	60	Reply
211	1030.311878	c4:01:05:06:00:01	DEC-MOP-Remote-Cons…	0x6002	77	DEC DNA Remote Console
212	1031.471986	c4:01:05:06:00:01	c4:01:05:06:00:01	LOOP	60	Reply
213	1035.215397	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=1/256, ttl=64 (reply in 214)
214	1035.260184	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=1/256, ttl=62 (request in 213)
215	1035.970571	c4:01:05:06:00:01	CDP/VTP/DTP/PAgP/UD…	CDP	355	Device ID: Router1 Port ID: FastEthernet0/1
216	1036.216672	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=2/512, ttl=64 (reply in 217)
217	1036.257599	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=2/512, ttl=62 (request in 216)
218	1037.217732	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=3/768, ttl=64 (reply in 219)
219	1037.253164	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=3/768, ttl=62 (request in 218)

ping from r1 to r2

No.	Time	Source	Destination	Protocol	Length	Info
572	1831.617650	c4:02:05:22:00:00	c4:02:05:22:00:00	LOOP	60	Reply
573	1834.983008	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=1/256, ttl=63 (reply in 574)
574	1834.995506	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=1/256, ttl=63 (request in 573)
575	1835.714443	c4:01:05:06:00:00	CDP/VTP/DTP/PAgP/UD…	CDP	355	Device ID: Router1 Port ID: FastEthernet0/0
576	1835.947337	10.0.1.1	224.0.0.9	RIPv2	66	Response
577	1835.969552	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=2/512, ttl=63 (reply in 578)
578	1835.991232	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=2/512, ttl=63 (request in 577)
579	1836.965684	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=3/768, ttl=63 (reply in 580)
580	1836.987076	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=3/768, ttl=63 (request in 579)

ping from r2 to ip2

No.	Time	Source	Destination	Protocol	Length	Info
183	999.761114	c4:02:05:22:00:01	c4:02:05:22:00:01	LOOP	60	Reply
184	1010.190254	c4:02:05:22:00:01	c4:02:05:22:00:01	LOOP	60	Reply
185	1013.506426	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=1/256, ttl=62 (reply in 186)
186	1013.506605	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=1/256, ttl=64 (request in 185)
187	1014.501802	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=2/512, ttl=62 (reply in 188)
188	1014.502253	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=2/512, ttl=64 (request in 187)
189	1015.497811	192.168.1.2	192.168.2.2	ICMP	98	Echo (ping) request id=0x0048, seq=3/768, ttl=62 (reply in 190)
190	1015.499238	192.168.2.2	192.168.1.2	ICMP	98	Echo (ping) reply id=0x0048, seq=3/768, ttl=64 (request in 189)

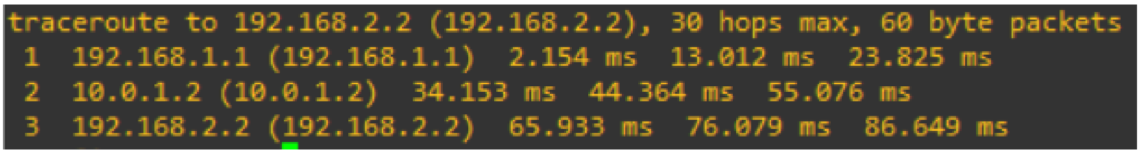
Figure 11: Capture for the route between ipterm 1 and ipterm 2

As expected, the router 1 resend to the router 2 that also resends to the ipterm 2 (192.168.2.2). Router 1 and 2 have been used as "transitive hosts", and are not a source nor a destination in all the parts of the route.

3.2 Traceroute

The following result is obtained when entering the following command in the terminal of the ipterm 1:

```
>>> traceroute 192.168.2.2
```



```
traceroute to 192.168.2.2 (192.168.2.2), 30 hops max, 60 byte packets
 1  192.168.1.1 (192.168.1.1)  2.154 ms  13.012 ms  23.825 ms
 2  10.0.1.2 (10.0.1.2)  34.153 ms  44.364 ms  55.076 ms
 3  192.168.2.2 (192.168.2.2)  65.933 ms  76.079 ms  86.649 ms
```

Figure 12: Route between ipterm 1 (192.168.1.2) to ipterm 2 (192.168.2.2)

As expected, the route goes from ipterm 1 (192.168.1.2) to router 1 (192.168.1.1) while staying in the left subnet (192.168.1.0). Then it goes from router 1 (10.0.1.1) to router 2 (10.0.1.2) while staying in the centre subnet (10.0.1.0). Finally, it goes from router 2 (192.168.2.1) to ipterm 2 (192.168.2.2) while staying in the right subnet (192.168.2.0).

4 Mission 3: Redundancy and dynamic re-routing

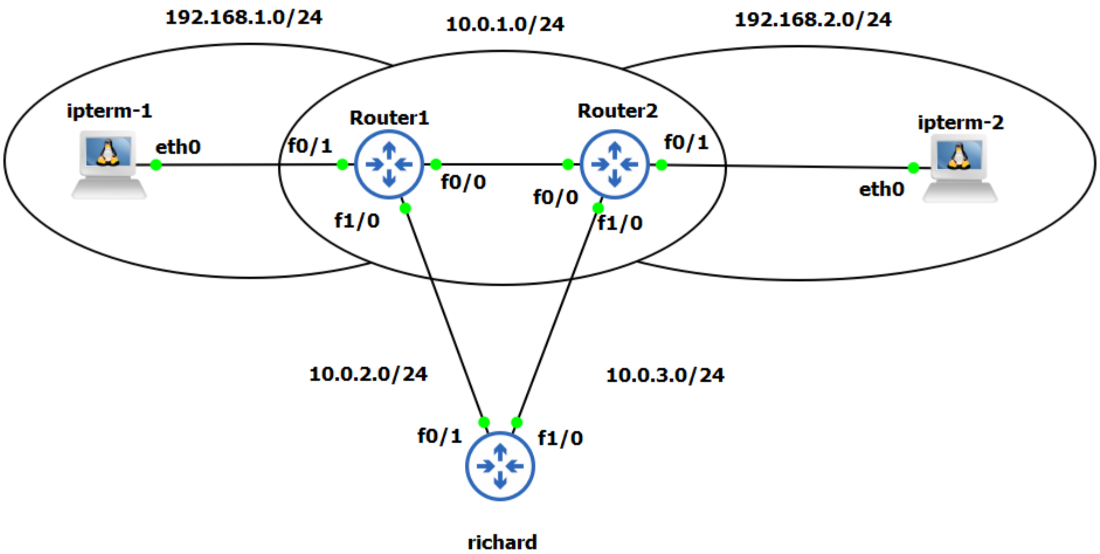


Figure 13: Topology

For this mission, a third router (richard) has been added and therefore, two new peering needs to be configured. Figure 14 represents the brief of richard. It is connected to the `10.0.2.0` peering network and the `10.0.3.0` peering network. For simplicity, the R3 router was connected as host 2 on both networks (so 10.0.2.2 and 10.0.3.2).

```
richard#show ip int br
Interface                  IP-Address      OK? Method Status                Prot
ocol
FastEthernet0/0            unassigned      YES unset  administratively down down

FastEthernet0/1            10.0.2.2        YES manual up                    up

FastEthernet1/0            10.0.3.2        YES manual up                    up
```

Figure 14: Richard Setup

Figures 15 and 16 shows the updated configuration of the ip interface of routers 1 and 2.

```
Router1#show ip int br
Interface                  IP-Address      OK? Method Status                Protocol
FastEthernet0/0            10.0.1.1        YES NVRAM  up                    up
FastEthernet0/1            192.168.1.1     YES NVRAM  up                    up
FastEthernet1/0            10.0.2.1        YES manual up                    up
```

Figure 15: New R1 setup

```
Router2#show ip int br
Interface                  IP-Address      OK? Method Status                Prot
ocol
FastEthernet0/0            10.0.1.2        YES NVRAM  up                    up

FastEthernet0/1            192.168.2.1     YES NVRAM  up                    up

FastEthernet1/0            10.0.3.1        YES manual up                    up
```

Figure 16: New R2 setup

Figure 17 show the updated Rip database and Rip routes of the three routers. It is interesting to see that a new route exists. Indeed, instead of going directly through Router 1 and Router 2. A new route going through Router 1 - Richard - Router 2 exists is also possible.

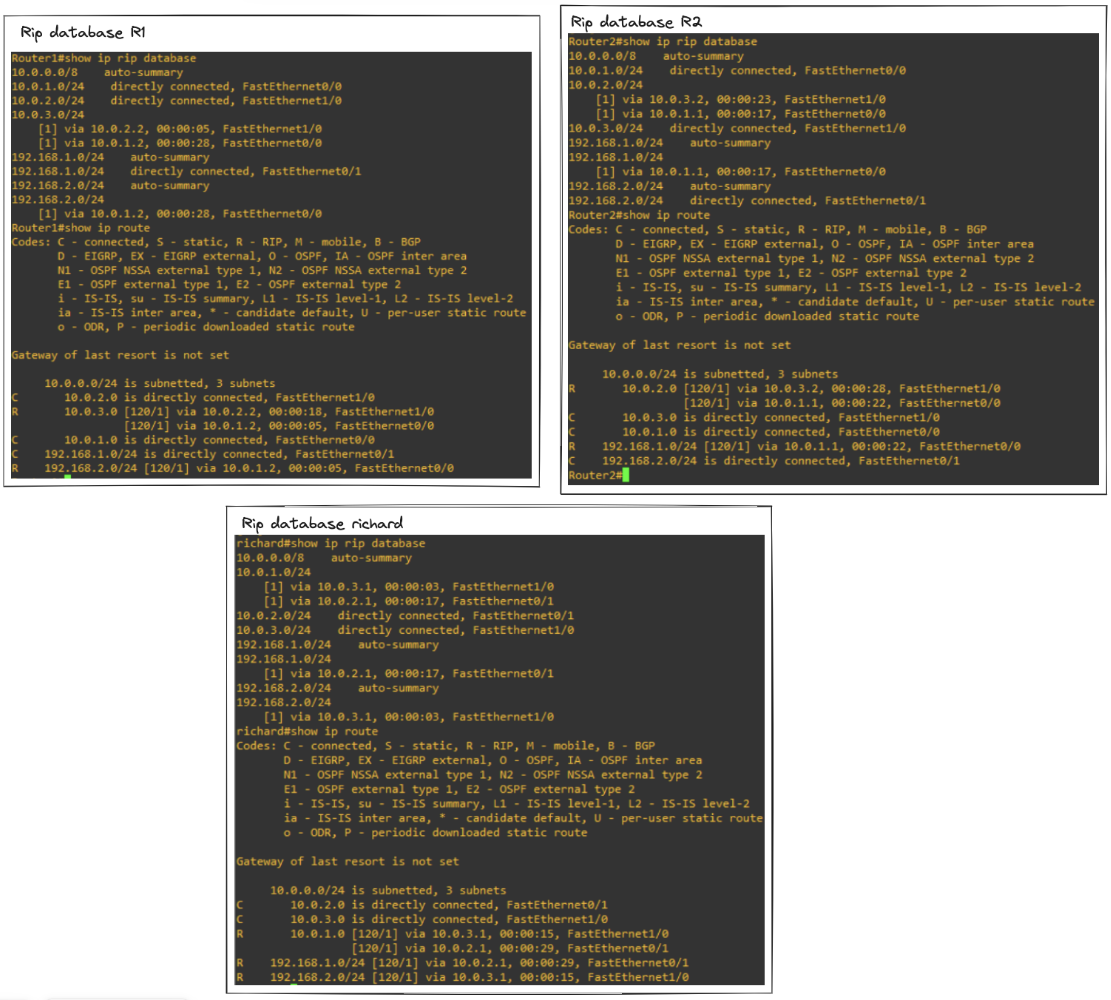


Figure 17: The Rip databases of the 3 routers

To check the route used to go from ipterm1 to ipterm2, the following command has been inserted in the Router 1 console.

```
>>> traceroute 192.168.2.2
```

It can be seen that the route has not changed since the arrival of Richard.

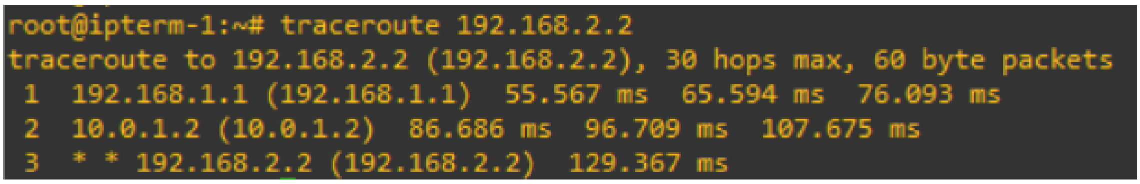


Figure 18: TraceRoute between ipterm 1 (192.168.1.2) and ipterm 2 (192.168.2.2)

But when removing the link between R1 and R2, the traceroute from ipterm 1 (192.168.1.2) to ipterm 2 (192.168.2.2) shows that it goes through router 1, then router Richard then router 2 and finally to ipterm 2.

```
traceroute to 192.168.2.2 (192.168.2.2), 30 hops max, 60 byte packets
 1  192.168.1.1 (192.168.1.1)  10.751 ms  20.713 ms  31.101 ms
 2  10.0.2.2 (10.0.2.2)  41.674 ms  51.861 ms  61.908 ms
 3  10.0.3.1 (10.0.3.1)  71.891 ms  81.963 ms  98.755 ms
 4  192.168.2.2 (192.168.2.2)  110.384 ms  120.426 ms  129.846 ms
```

Figure 19: Traceroute when the link between R1 and R2

It can thus be deduced that the RipV2 adapts the route automatically if something happens, in this case a broken link in the route between ipterm1 and ipterm2. And it will also try to take the route that passes through the fewest amount of hosts (here routers) possible.