HY-335b
Project - Phase A

Configuring a “mini” world wide network.

Mike Bastakis - csd 4406
Manos Kalozimis - csd 4502
February 2021 - April 2021

Question 1.1

The **ASN** of our team is 84, which means that our local subnet is :

Default IP:	84.200.0.0
Def. IP in binary:	01010100.11001000.00000000.00000000
Mask:	11111111.11111111.11111110.00000000
IP address:	01010100.11001000.00000000.00000000
IP`s range from:	84.200.0.1 - 84.200.1.254

For this question we started assigning IP addresses to each host and router from the first IP that was available until all of them had IP's.

Student_1	IP:	84.200.0.3/24	Staff_1	IP:	84.200.0.4/24
Student_2	IP:	84.200.0.5/24	Staff_2	IP:	84.200.0.6/24
Student_3	IP:	84.200.0.7/24	Staff_3	IP:	84.200.0.8/24

student #1 

ifconfig

```
root@student_1:~# ifconfig
84-CERN: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.3  netmask 255.255.254.0  broadcast 84.200.1.255
        ether 46:c9:6d:57:57:5e  txqueuelen 1000  (Ethernet)
        RX packets 1425442  bytes 74312177 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 110  bytes 8288 (8.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

```
netsroot@student_1:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.2      0.0.0.0         UG        0 0          0 84-CERN
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-CERN
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

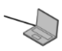
staff #1 

ifconfig

```
root@staff_1:~# ifconfig
84-CERN: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.4  netmask 255.255.254.0  broadcast 84.200.1.255
        ether a6:68:15:70:ef:f4  txqueuelen 1000  (Ethernet)
        RX packets 1425438  bytes 74308037 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 5  bytes 378 (378.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

```
root@staff_1:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.2      0.0.0.0         UG        0 0          0 84-CERN
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-CERN
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

 student #2

ifconfig

```
root@student_2:~# ifconfig
84-ETHZ: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.5  netmask 255.255.254.0  broadcast 84.200.1.255
        ether 12:dd:14:94:3a:cd  txqueuelen 1000  (Ethernet)
        RX packets 1425524  bytes 74312451 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 8  bytes 672 (672.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

```
root@student_2:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.1      0.0.0.0         UG        0 0          0 84-ETHZ
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-ETHZ
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

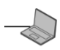
 staff #2

ifconfig

```
root@staff_2:~# ifconfig
84-ETHZ: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.6  netmask 255.255.254.0  broadcast 84.200.1.255
        ether 0a:16:7f:1a:d3:7b  txqueuelen 1000  (Ethernet)
        RX packets 1425453  bytes 74308905 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1050 (1.0 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

```
root@staff_2:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.1      0.0.0.0         UG        0 0          0 84-ETHZ
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-ETHZ
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

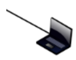
 student #3

ifconfig

```
root@student_3:~# ifconfig
84-EPFL: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.7  netmask 255.255.254.0  broadcast 84.200.1.255
        ether 5a:e2:c2:77:04:e4  txqueuelen 1000  (Ethernet)
        RX packets 1425619  bytes 74320830 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 81  bytes 5838 (5.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

```
root@student_3:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.2      0.0.0.0         UG        0 0          0 84-EPFL
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-EPFL
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

 staff #3

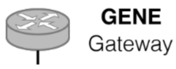
ifconfig

```
root@staff_3:~# ifconfig
84-EPFL: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 84.200.0.8  netmask 255.255.254.0  broadcast 84.200.1.255
        ether a2:f0:57:64:3f:a9  txqueuelen 1000  (Ethernet)
        RX packets 1425497  bytes 74310993 (70.8 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 7  bytes 574 (574.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

netstat

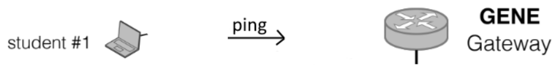
```
root@staff_3:~# netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         84.200.0.2      0.0.0.0         UG        0 0          0 84-EPFL
84.200.0.0      0.0.0.0         255.255.254.0   U         0 0          0 84-EPFL
158.84.0.0      0.0.0.0         255.255.0.0     U         0 0          0 ssh
```

Gene-L2 IP: 84.200.0.2/24 Zuri-L2 IP: 84.200.0.1/24

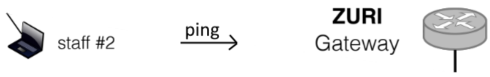



```
GENE_router# show int brief
Interface        Status  VRF              Addresses
---------        ------  ---              ---------
GENE-L2          up      default          84.200.0.2/24
GENE-L2.10       up      default
GENE-L2.20       up      default
GENE-L2.30       down    default
ext_86_LOND      up      default
lo               up      default
port_MIAM        up      default
port_PARI        up      default
ssh              up      default          158.84.13.1/16
```

```
ZURI_router# show int brief
Interface        Status  VRF              Addresses
---------        ------  ---              ---------
ZURI-L2          up      default          84.200.0.1/24
ZURI-L2.10       up      default
ZURI-L2.20       up      default
ZURI-L2.30       down    default
ext_81_ZURI      up      default
lo               up      default
measurement_84   up      default          84.0.199.1/24
port_LOND        up      default
port_PARI        up      default
ssh              up      default          158.84.11.1/16
```




```
root@student_1:~# ping 84.200.0.2
PING 84.200.0.2 (84.200.0.2) 56(84) bytes of data.
64 bytes from 84.200.0.2: icmp_seq=1 ttl=64 time=7.69 ms
64 bytes from 84.200.0.2: icmp_seq=2 ttl=64 time=2.16 ms
64 bytes from 84.200.0.2: icmp_seq=3 ttl=64 time=2.18 ms
64 bytes from 84.200.0.2: icmp_seq=4 ttl=64 time=2.16 ms
64 bytes from 84.200.0.2: icmp_seq=5 ttl=64 time=2.17 ms
```




```
root@staff_2:~# ping 84.200.0.1
PING 84.200.0.1 (84.200.0.1) 56(84) bytes of data.
64 bytes from 84.200.0.1: icmp_seq=1 ttl=64 time=7.64 ms
64 bytes from 84.200.0.1: icmp_seq=2 ttl=64 time=3.10 ms
64 bytes from 84.200.0.1: icmp_seq=3 ttl=64 time=2.19 ms
64 bytes from 84.200.0.1: icmp_seq=4 ttl=64 time=2.20 ms
64 bytes from 84.200.0.1: icmp_seq=5 ttl=64 time=2.21 ms
```

We used a mask of 24 bits for all the IP's that we set up. Next we distributed IP's to each host and router to make them be able to communicate. We also added in each host a default gateway. The hosts connected to CERN and EPFL use GENE as their default gateway and hosts at ETHZ use ZURI (This can be seen at the pictures above where we configure each host).

After setting up the hosts and the routers we can ping from any host to any other host using the Layer 2 connection. Each host can communicate outside of the Layer 2 configuration by sending packets to their default gateways which are GENE for Student_1/3, Staff_1/3 and ZURI for Student_2, Staff_2.

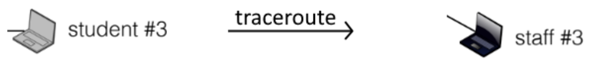
Question 1.2

In this question some changes must be made in order to enable direct layer-2 connectivity between students and staff members, but not in between them. To do this we have to use two subnets one for the students and the other for the staff members. So the new IP's are going to be :

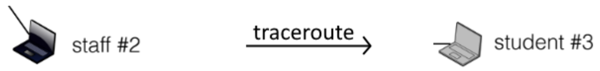
Subnet for Students:	84.200.0.0/24
Subnet for Staff:	84.200.1.0/24

Gene-L2.10	IP:	84.200.1.100/24	Gene-L2.20	IP:	84.200.0.100/24
Zuri-L2.10	IP:	84.200.1.200/24	Zuri-L2.20	IP:	84.200.0.200/24
Student_1	IP:	84.200.0.1/24	Staff_1	IP:	84.200.1.1/24
Student_2	IP:	84.200.0.2/24	Staff_2	IP:	84.200.1.2/24
Student_3	IP:	84.200.0.3/24	Staff_3	IP:	84.200.1.3/24

We now have two subnets so the students who are in the same subnet can communicate in the layer-2 but in order to communicate with another subnet they have to communicate through a router. We made those changes by setting up the VLANs on the switches to let the interfaces that have tag = 10/20 talk to each other but if an interface with tag = 10 is trying to communicate with an interface with tag = 20, the switches will make them communicate through the router.

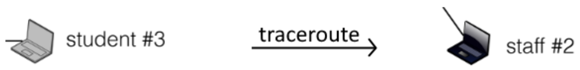



```
root@student_3:~# traceroute 84.200.1.3
traceroute to 84.200.1.3 (84.200.1.3), 30 hops max, 60 byte packets
 1  84.200.0.100 (84.200.0.100)  26.485 ms  26.333 ms  26.321 ms
 2  84.200.1.3 (84.200.1.3)  39.817 ms  38.723 ms  39.794 ms
```

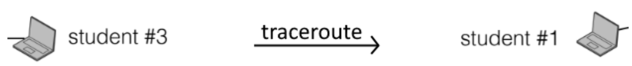



```
root@staff_2:~# traceroute 84.200.0.3
traceroute to 84.200.0.3 (84.200.0.3), 30 hops max, 60 byte packets
 1  84.200.1.200 (84.200.1.200)  46.525 ms  58.500 ms  58.489 ms
 2  84.200.0.3 (84.200.0.3)  108.414 ms  108.400 ms  108.417 ms
```

As we can see in those examples when a student is trying to send packets to a staff member he has to communicate through the router.




```
root@student_3:~# traceroute 84.200.1.2
traceroute to 84.200.1.2 (84.200.1.2), 30 hops max, 60 byte packets
 1  84.200.0.100 (84.200.0.100)  16.638 ms  16.511 ms  16.475 ms
 2  84.200.1.2 (84.200.1.2)  20.242 ms  20.189 ms  20.147 ms
```



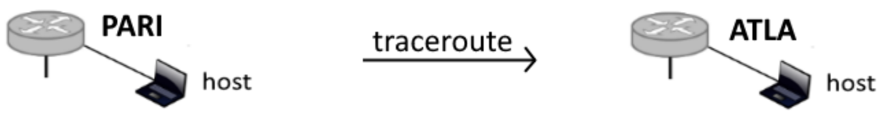

```
root@student_3:~# traceroute 84.200.0.1
traceroute to 84.200.0.1 (84.200.0.1), 30 hops max, 60 byte packets
 1  84.200.0.1 (84.200.0.1)  18.917 ms  18.776 ms  18.715 ms
```

When a student is trying to communicate with another student he can do so via the layer-2 connectivity.

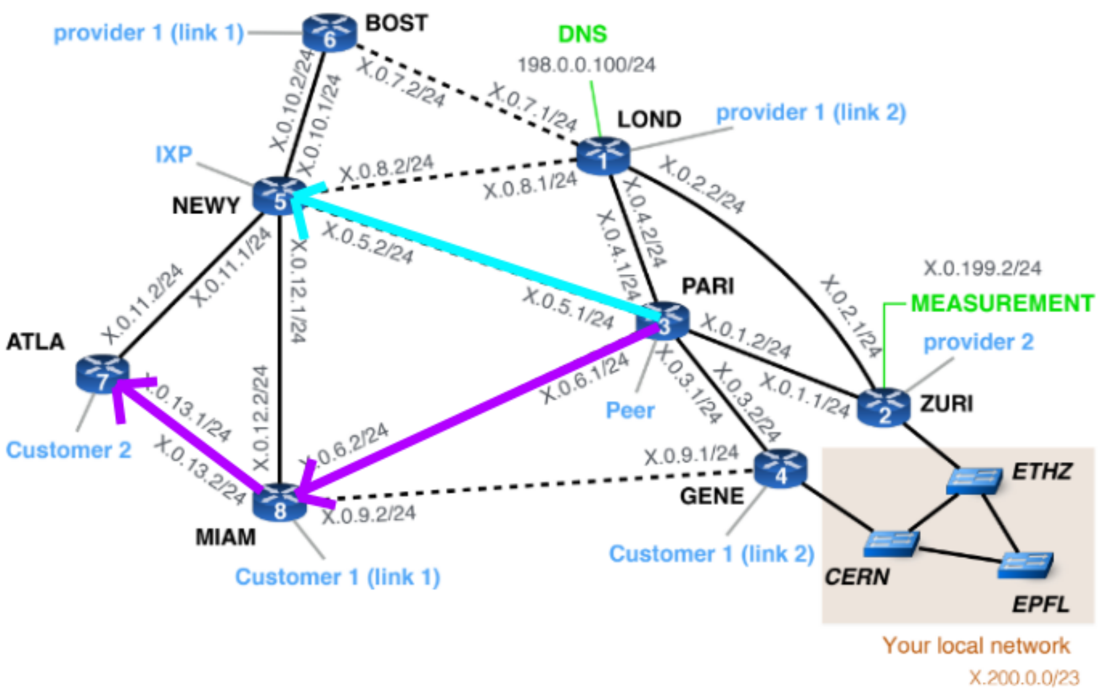
Question 1.3

This question demands the configuration of the OSPF. But before we are able to set the OSPF we have to distribute IP addresses on each interface of our routers and hosts, with some exceptions. After configuring all of the interfaces, we begin setting up the OSPF. By doing that, the topology information is flooded throughout the AS, so that every router within has a complete picture of the topology. Now every host can communicate through the routers and send packets to other hosts.

Here we can see an example of a trace route from the host of PARI to the host of ATLA:



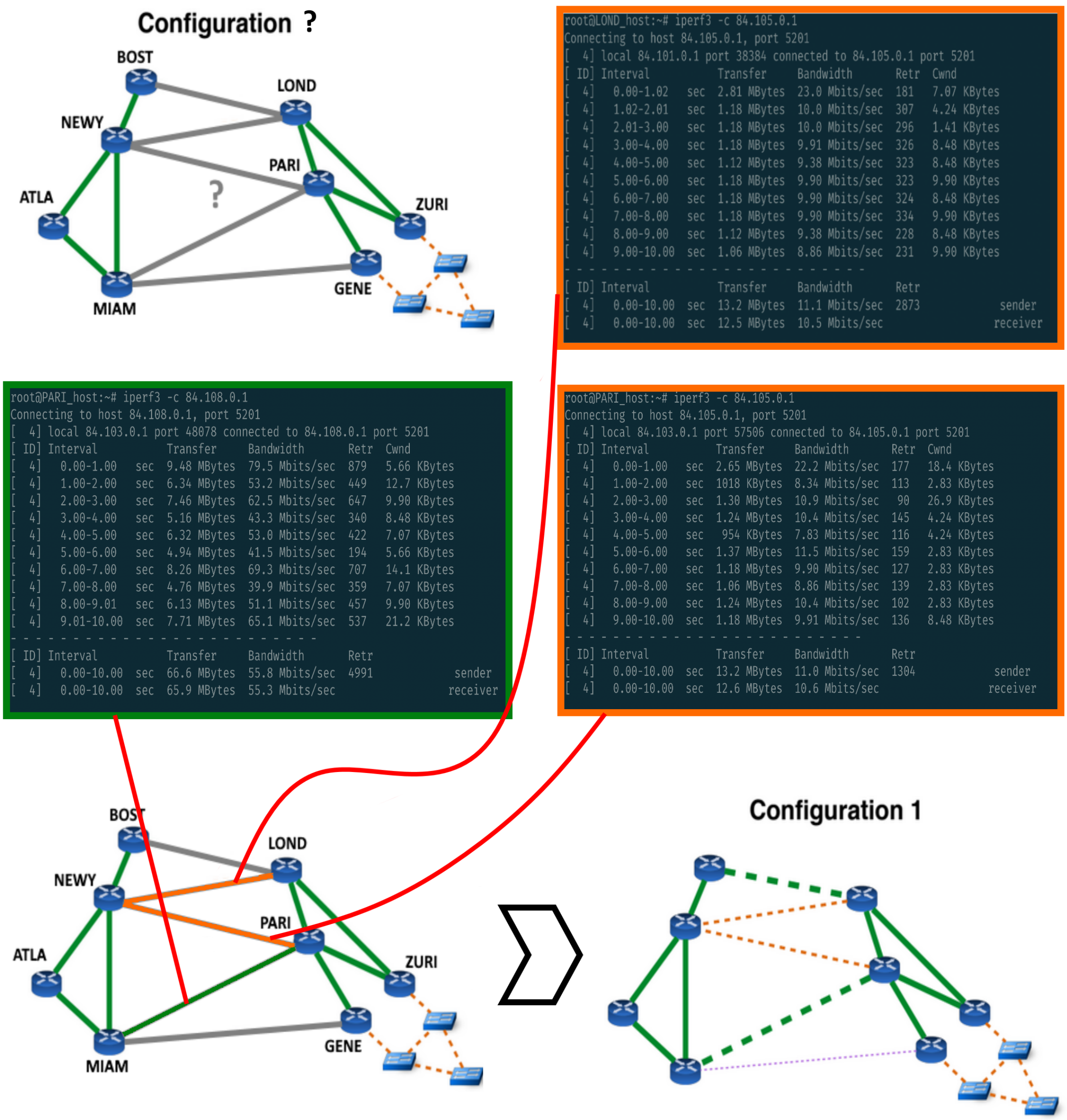

```
root@PARI_host:~# traceroute 84.107.0.1
traceroute to 84.107.0.1 (84.107.0.1), 30 hops max, 60 byte packets
 1  84.103.0.2 (84.103.0.2)  0.461 ms  0.286 ms  0.270 ms
 2  84.0.6.2 (84.0.6.2)  0.554 ms  0.534 ms 84.0.5.2 (84.0.5.2)  2.344 ms
 3  84.0.13.1 (84.0.13.1)  0.783 ms  0.742 ms  0.740 ms
 4  84.107.0.1 (84.107.0.1)  1.815 ms  1.663 ms  1.916 ms
```



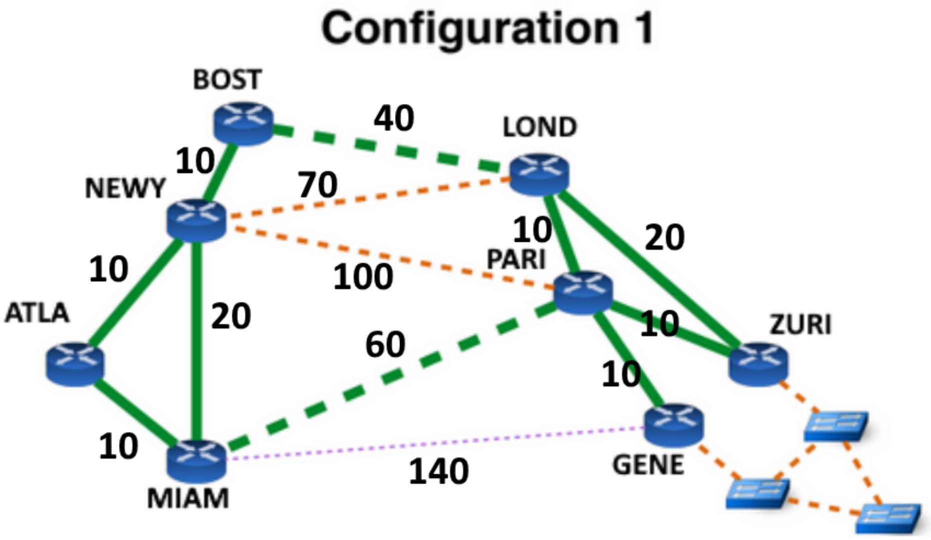
As we can observe from the picture the connections of the interfaces were done correctly as well as the setting of OSPF since the host of PARI can communicate with the host of ATLA.

Question 1.4

Now our goal is to provide the best performance. We must configure the weights of our OSPF to make sure the traffic never traverses two submarine links. Before we can do that, we have to find the bandwidth configuration of our AS. We choose a host to set up a server and another host to send pings to that server and see how much time the packets need to reach their destination. We can determine what configuration our AS has based on the bandwidth of the links.

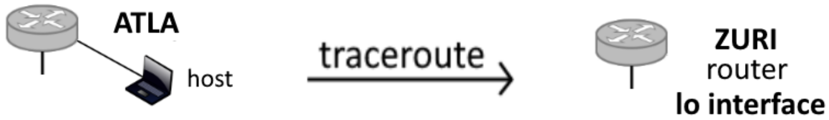


Now that we have determined the configuration of our AS, we can start assigning weight to the OSPF's links, in order to prevent extremely inefficient pathing. We must also keep in mind that all traffic from MIAM to NEWY, ZURI to LOND, and ATLA to ZURI is load balanced. Load balanced traffic is achieved by having multiple optimal paths that weigh the same.

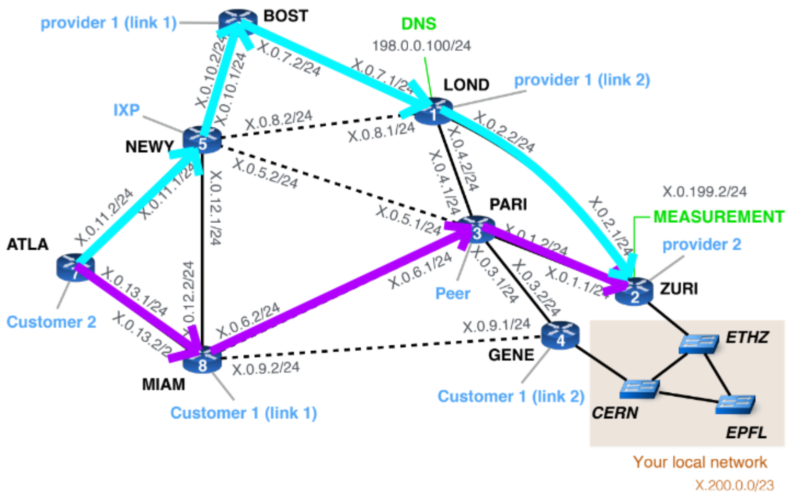


First of all, we set the cost of every green (non submarine) link equal to 10. Then we adjusted the costs of the links that needed to be load balanced accordingly (MIAM-NEWY = 20 <=> MIAM-ATLA + ATLA-NEWY = 20, same procedure has been followed for the ZURI-PARI-LOND triangle). Since the ATLA and ZURI path has to be load balanced, having two high bandwidth submarine links unaccounted for their weight, all we had to do was to make sure that both paths using these links costed the same.

The rest of the submarine links have been assigned costs that will ensure they are slower than higher bandwidth options.




```
root@ATLA_host:~# traceroute 84.152.0.1
traceroute to 84.152.0.1 (84.152.0.1), 30 hops max, 60 byte packets
 1  84.107.0.2 (84.107.0.2)  0.329 ms  0.215 ms  0.300 ms
 2  84.0.11.1 (84.0.11.1)  0.563 ms  0.395 ms 84.0.13.2 (84.0.13.2)  0.439 ms
 3  84.0.10.2 (84.0.10.2)  0.464 ms  0.550 ms 84.0.6.1 (84.0.6.1)  0.530 ms
 4  84.152.0.1 (84.152.0.1)  12.018 ms  12.012 ms 84.0.7.1 (84.0.7.1)  20.615 ms
```

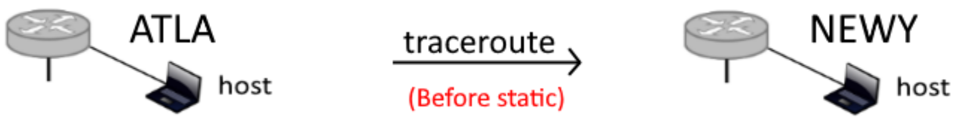


As the screenshot indicates, two paths have been chosen, as predicted. Their OSPF link cost sums are equal. Both paths use the designated high bandwidth submarine links, due to their efficiency.

Question 1.5

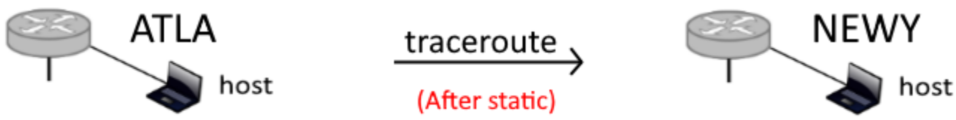
We assigned the lowest value a link can have based on our configuration for ATLA-NEWY because it is a direct link, so being the fastest path makes perfect sense.

Traceroute results from ATLA host to NEWY host can be seen below:




```
traceroute root@ATLA_host:~# traceroute 84.105.0.1
traceroute to 84.105.0.1 (84.105.0.1), 30 hops max, 60 byte packets
 1  84.107.0.2 (84.107.0.2)  0.314 ms  0.464 ms  0.277 ms
 2  84.0.11.1 (84.0.11.1)  0.406 ms  0.335 ms  0.314 ms
 3  84.105.0.1 (84.105.0.1)  2.147 ms  2.174 ms  1.874 ms
```

Now that the direct link is prohibited, static routes must be used. In order to achieve connectivity from ATLA-NEWY, bidirectional static routes have to be established through MIAM (as seen below).




```
root@ATLA_host:~# traceroute 84.105.0.1
traceroute to 84.105.0.1 (84.105.0.1), 30 hops max, 60 byte packets
 1  84.107.0.2 (84.107.0.2)  0.491 ms  0.386 ms  0.386 ms
 2  84.0.13.2 (84.0.13.2)  0.749 ms  0.726 ms  0.720 ms
 3  84.0.12.1 (84.0.12.1)  0.866 ms  0.842 ms  1.122 ms
 4  84.105.0.1 (84.105.0.1)  1.218 ms  1.270 ms  1.560 ms
```

Final thoughts

More material and detailed explanations could be added, but the ten page limit proved to be restricting. Trough the process of completing this project we encountered difficulties that eventually helped us understand computer networking more deeply. Thank you for your time.