TP2 - OSPF

GIN 201 - Leonardo Linguaglossa, Jean-Louis Rougier

October 2024

1 Introduction

The purpose of this Practical Lab (TP) is to get familiar with the IP routing (IGP) protocols used in Intranets, in particular OSPF. The lab will allow you to see:

- how OSPF works (hello, link state database) with focus in case of link/node failures;
- how OSPF is configured on routers.

A particular attention is given to OSPF convergence time in case of link/node failures, which is an important performance metric. Even if such failures are rare, their impact can be detrimental, especially when transporting specific "critical" services. The IGP will re-converge and find alternate routes, but this won't be immediate. During re-convergence, packets are lost, and this could be problematic for some applications (e.g. payment transactions, high speed trading, ...). Having "fast" convergence time is thus important for a carrier (as a differentiation factor in a very competitive market). "Fast re-route" services may also be offered to customers as added value services.

The practical lab is based on the `kathara` framework equipped with `quagga`. The framework will instantiate containers with two roles: either a *router* executing a routing protocol, or a regular *PC*. We will perform "crash tests" and emulate equipment failures by, for instance, powering off a router, as well as "configuration procedures", where we will interact with the command line interface of an IP router based on (but not equivalent to) Cisco IOS.

In this (and following) lab descriptions you are going to find the steps that you will run on the machines, as well as some $\boxed{\text{questions}}$ to help you write a report. Please refer to the following guidelines for the submission of your report.

1.1 Guidelines to follow for the report (valid for TP2 and TP3)

- You can use either English or French, as you prefer.
- You have to use *handwriting* in the precompiled spaces.
- You must be **concise**: go straight to the point.
- You can upload an **Appendix** that includes screenshots and/or scripts, but you must **label and comment** all your figures. If a screenshot is left without explanation, it is ignored and considered as a negative point. In all cases, the Appendix is optional, and you must include all important points in the handwritten file.
- You can work in groups: you have to **specify if that is the case** in the first page of the report. Please, provide only a single report per group.
- Proofread your work: when you read it, you have to be **proud (and not ashamed) of it**.
- Submit your work using the Moodle submission tool. You have one week to submit the report, and all delay is a negative point.

1.2 Environment

You can download all the environment from Gitlab `https://gitlab.telecom-paris.fr/linguaglossa/gin201-ospf-lab`. You can choose whether you want to download the source as a zip file and extract it, or if you want to `git clone` the repository directly.

The network consists of 6 Routers (R1 to R6), a switch and 3 Linux end-points servers (PC1 to PC3). The network is fully configured. Routers are identified by their loopback address X.X.X.X (X=1, ..., 6).

For the other addresses, we adopt the following scheme:

- 10.10.99.X/24 for the central subnet (swichted) linking R4,R5 and R6. (10.10.99.4 is the address of R4 on this subnet)
- 10.10.XY.X/24 for the point to point link between node X and node Y (with X<Y). Ex: 10.10.14.4 for node R4 on the link between R1 and R4
- 10.10.X.X/24 is assigned to PCX.

Later we will enable IPv6 as well with prefix 2001:DB8::/32 (as always for illustration purposes, RFC3849) with a similar address scheme (2001:db8::1/128 for loopback of R1, 2001:db8:10:14::4/64 for link R1-R4 etc.).

Remark. IPv6 uses a hexadecimal notation so the address 10.10.14.4 should logically become 2001:db8:a:e::4. We keep the decimal numbers for the sake of simplicity.

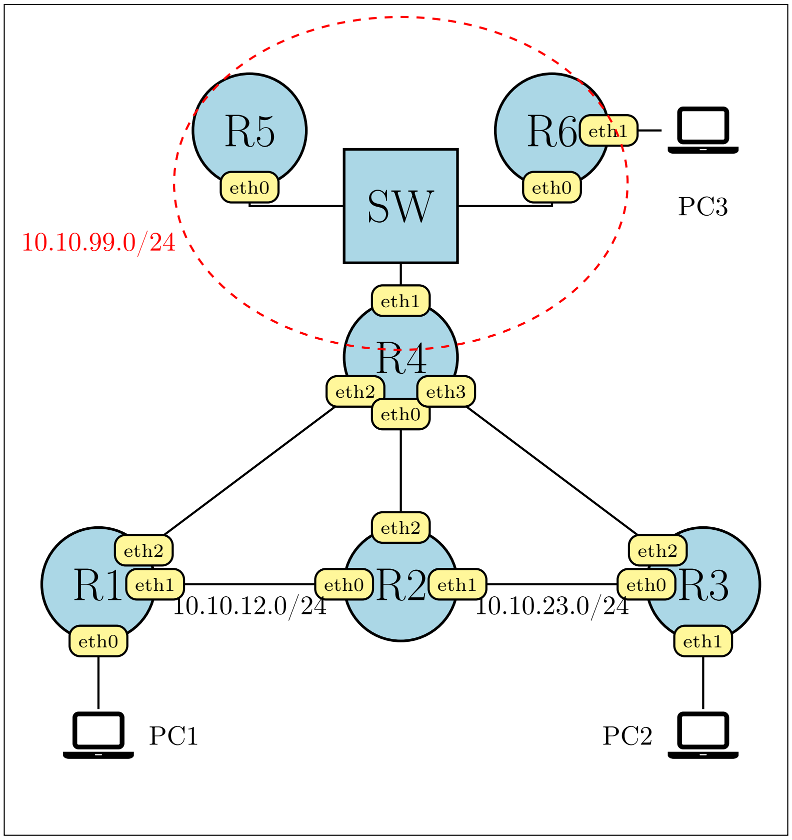

Figure 1: The network that you will be using in this lab.

Start the lab

In order to start the lab you need to prepare the environment. First of all, both **kathara** and **docker.io** need to be installed. If you are using a pre-configured virtual machine, you can skip this first session.

```
sudo apt-get install docker.io xterm

sudo add-apt-repository ppa:katharaframework/kathara

echo "deb␣http://ppa.launchpad.net/katharaframework/kathara/ubuntu␣focal␣main" | sudo tee /etc/apt/sources.list.d/kathara.list

echo "deb-src␣http://ppa.launchpad.net/katharaframework/kathara/ubuntu␣focal␣main" | sudo tee -a /etc/apt/sources.list.d/kathara.list

sudo apt-get update && sudo apt-get install kathara
```

After the environment is ready (or after you have logged in your virtual machine) you can start the lab. As an example, if you choose to download the source from Moodle, these are the steps to do:

1. Download the file `lab-tp-2` from the moodle
2. Extract it to your home folder.
3. Access the lab extracted folder via the command line: `cd $HOME/tp-lab-2/`
4. Start the laboratory: `sudo kathara lstart`

5. Optional : it may be possible that some errors will be shown (See Fig.2). In such case, clean the lab and redo step 4: `sudo kathara lclean && sudo kathara lstart`

> **Question 1.** Write the IP address and the Ethernet port number assigned to each of the 3 PCs. Verify the connectivity with the help of the `ping` tool. If you encounter connectivity issues, try again after a few seconds and explain what you are observing.

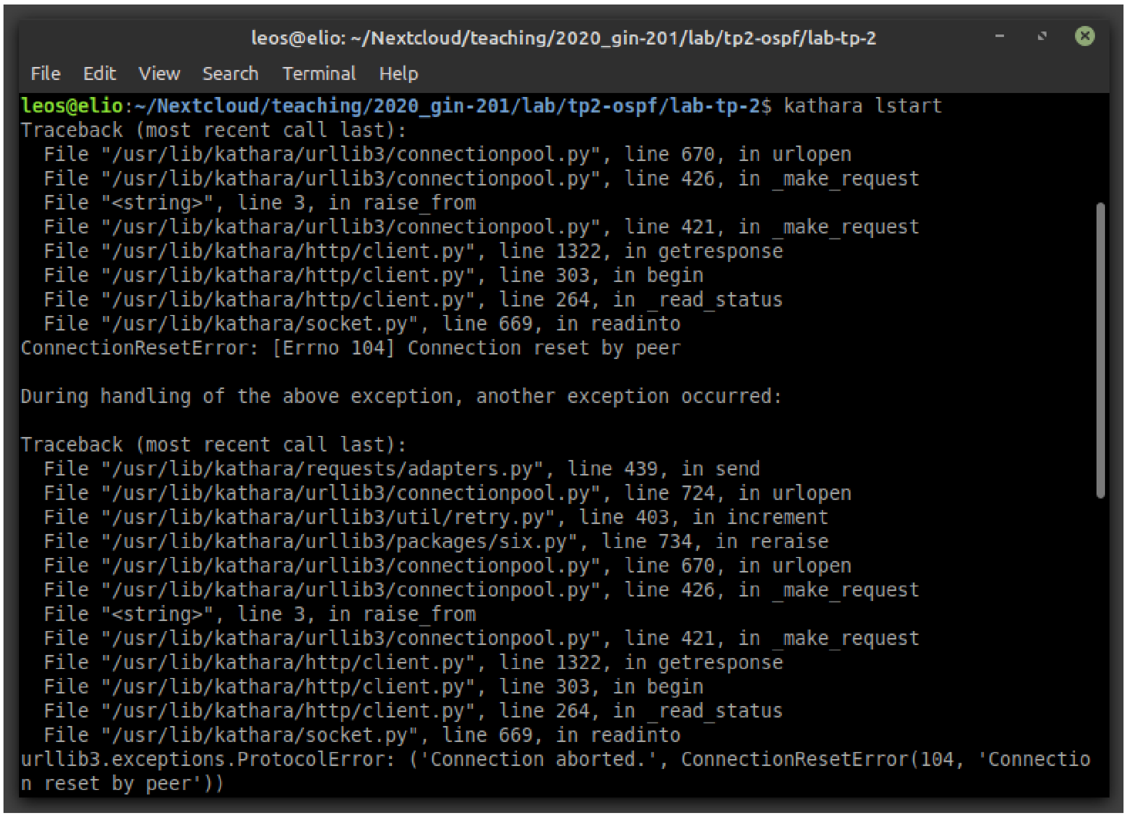

Figure 2: Possible errors during the startup of the lab.

1.3 Connection to the devices

When the lab starts, all the devices will be created and a terminal (`xterm` by default) will be connected each of them. If you want, you can close all these terminals and reconnect to the devices at any time. In order to connect to a specific device, use the command:

```
kathara connect pc1
```

Remember to launch the command while in the same directory of the lab. Of course, for all subsequent connections you will need to replace pc1 with the name of the device you wish to connect to. Names are case-sensitive, and we conventionally use all lower letters. The network will be brought up with a delay that depends on the configuration of the routing protocol. This means that if you try to ping between any two machines, you can receive a "destination unreachable" message for a few seconds.

Question 2. With the help of the `traceroute` tool, show the path between the three PCs.

1.4 Router configuration

When kathara is started, the routing facilities are also deployed on the devices. In particular, each router is equipped with `quagga`, a router emulator which takes inspiration from Cisco-like devices. After connected to any of the routers, it is possible to use telnet to connect to the router's interface. If asked for password, the password to be used is always **zebra**.

The following commands:

```
kathara connect r1
telnet localhost ospfd
```

will connect to R1 and then to the OSPF interface of the router.

Once connected to a router with `telnet`, you can get more information about the router's interfaces and addresses (type **?** to get interactive help, even after a command).

First, you'll need to connect as "super user" (root):

`enable` (or "ena" for short)

To get the topology table:

```
show ip ospf database
```

To look at the router's configuration:

```
show running-config
```

To see its routing table:

```
show ip ospf route
```

2 Understanding OSPF

There are three main data structures involved in the OSPF protocol:

- The OSPF database
- The neighbor table
- The routing table

OSPF implements the construction of these data structures in a distributed manner. This means that some sort of communication between the routers must be executed. OSPF allows the exchange of messages using different packets:

- Hello
- Database Descriptor
- Link-state request
- Link-state update

- Link-state acknowledgement

We explore some of these concepts in the following sections.

2.1 The OSPF database

The OSPF protocol is based on a link-state algorithm (the Dijkstra algorithm) which creates a full topology of the network. Such topology is stored in the LSDB (Link-state Database) or simply the OSPF database. The entries of this database are the **Link-state advertisements**, or LSAs. There exist several types of LSAs, but we will focus on the first two types:

- Type-I (Router LSA);
- Type-II (Network LSA).

You can see the OSPF database and more detailed information using the following commands:

```
show ip ospf database
show ip ospf database router
show ip ospf database network
```

Question 3. How many LSAs are available in the database (of type router and network LSA in particular)? Hint: look for the values LS-type: router-LSA or LS-type: network-LSA

Hello packets You can observe OSPF "in action" by capturing traffic on a given interface, or by the logs available on the routers. Logging can be activated using the "debug" command, while on the telnet ospfd command line interface in super user (root) mode. After the activation, logs can be accessed on the Linux terminal of the device where you activated the debug under the location `/var/log/zebra/ospf.log`.

Connect now to any router you wish to analyze, then open the OSPF console and activate the root mode. Then type:

```
debug ospf packet hello
```

To quit the ospfd interface, hit CTRL-D to log out of the session, and then access the log files.

This command logs all exchanges involving the Hello packets in a separate file. These packets are sent periodically to a multicast address representing all OSPF routers in the network (to be precise, all OSPF routers within the **same area**). The main function of this packet type is to discover all OSPF-speaking routers in a network (and then calculate the full topology). Furthermore, additional information transported in the Hello packets can determine if two routers are neighbors.

In order to capture live traffic and check the Hello packets, open another terminal with CTRL-ALT-T, connect to R3 and use the `tcpdump` command.

```
cd $HOME/tp-lab-2/
kathara connect r3
tcpdump -i eth0
```

Question 4. What is the frequency of transmission of Hello packets? How did you verify it?

2.2 The neighbor table

All OSPF routers maintain a table that contains all directly connected OSPF routers, aka neighbors. The neighbor table contains information about the state of adjacency between neighbors, and relies on the concept of Hello packets exchange.

You can have a look at the neighbor table with the command:

```
show ip ospf neighbor
```

Question 5. How adjacencies are acknowledged (i.e. how does one router know that communication can take place **in both directions**)?

It is possible to retrieve additional information by appending the keyword `detail` at the previous command. As we shall see in Sec.4, the adjacency state may be different in multi-access network segments.

2.3 Routing table

The last table OSPF uses is the routing table. This is not specific to OSPF; rather, it is created via the OSPF and other protocols to compute the shortest path between any two points in the network.

Check the routing table with the command:

```
show ip ospf route
```

You will see three sections in the output of this command, but for now we will consider only the first: the OSPF network routing table (the other parts include the routers ID of area-border routers, and the routes generated by other external protocols).

This section contains the following data:

Entry type	Network prefix	Path cost	Area	Next hop	Output interface
N (as in network)	10.10.10.1/24	[30]	area: 0.0.0.0	via 10.10.99.4	eth0

We will observe some of these values in Sec.3

Question 6. Show a simplified representation of the OSPF database and the OSPF routing table on two different routers. Are they the same? What are the differences?

3 OSPF convergence

It is difficult to observe link state messages as they are only refreshed every 30 minutes! (One would have to be patient...) But we will have the opportunity to observe Link State Advertisements during the "crash test" in this part of the lab, that explores OSPF convergence in case of node failure.

We will image that a router, for example R2, is shut down (due power outage for instance). The loss of the router will be simulated by stopping the container. This represents a somehow "brutal" event (not letting time to the OSPF process to stop properly, canceling its adjacencies and warning its neighbors), corresponding to a power outage or hardware failure – worst case scenarios.

Now follow this experimental procedure:

- On one terminal, connect to PC1 and launch `traceroute` between PC1 and PC2. Double-check that the route involves R2.

- On the same terminal, start a ping between PC1 and PC2 (and vice versa if you want...)
- On a second terminal, connect to to R1 and listen to all the packets received from any interface with the command `tcpdump -i any`
- On a third terminal, `cd` to the main folder of the lab (on your host operating system) and kill the container of R2.

In order to do so, get the information of the R2 container, and use the following commands:

```
sudo docker container ls
```

Find the hash code associated with R2, and then do:

```
sudo docker container kill 8b105a7ad65f
```

Replace the hash code in this sample command with the container id, as shown in Fig.3. Your container id corresponds to the string: kathara_[YOUR-USER-NAME]_r2_[LAB-HASH-ID]

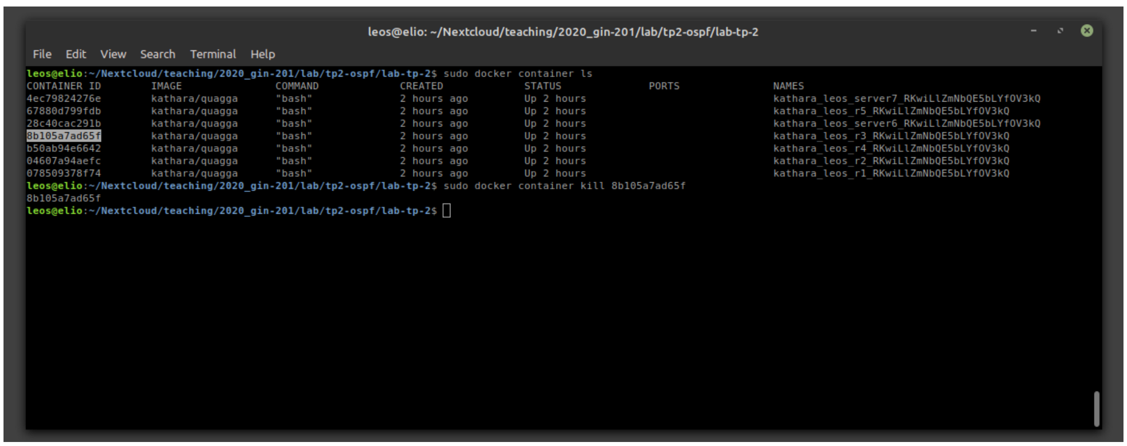

Figure 3: This screenshot is only valid locally, and your configuration may be different.

The expectation is that some pings may be lost during the convergence of OSPF (ie. before alternate routes, bypassing the dead router, are computed). Note that pings are sent by default every 1s on Linux. If you want, you can enable logging ("`debug ospf event`" and/or "`debug ospf packet all`") on the neighbors of R2. Wait for a few minutes and then analyze what has happened...

> **Question 7.** What is the convergence time in this scenario? Describe what has happened and find the new route between the two servers in the new scenario. Can you infer how many packets have been lost via the ping command?

You may also want to analyze what's going on when node R2 is restarted (e.g. by launching a packet capture before the reconfiguration of R3). This will be less impressive: the neighbors already have a route towards all destinations, although the new paths may not be the optimal ones. Hence no packet loss will occur (you would probably see nothing with pings). It is worth observing, however, to see the router synchronization phase (R2 will ask its neighbors to send him the missing link states etc.). You can imagine that, after R2 is brought back, all the routers will eventually discover that a better path is back again in the network.

In order to restart R2, you can follow these steps:

- CTRL-ALT-T to open a new terminal, and cd to the main directory of the lab.

- Start the container that we previously killed with:
  `sudo docker container start 8b105a7ad65f`
- Connect to R2: you will see that you will need to reconfigure the ip addresses. With the help of the command `/sbin/ifconfig eth[X] [IP] netmask [MASK] up` configure all the interfaces of R2. (Replace [X], [IP] and [MASK] with the correct values: refer to Fig.1 and Sec.1.2).
- Start the quagga routing interface with the command: `/etc/init.d/quagga start`

You will now see that the ping between the two PCs will begin to take the privileged path.

Question 8. With the help of the `show ip ospf route` command, can you tell what is the cost of reaching PC2 from the router R1? What would have been the cost of reaching this PC when the intermediate router was taken down? Explain your findings.

3.1 Changing the metrics

You can manually configure the OSPF cost assigned to different interfaces to force packets to take a specific path. You can do so by entering the configuration console and then configuring the specific interface. For example, on router R1 you can do :

```
telnet localhost ospfd
enable
configure terminal
interface eth1
ip ospf cost 500
```

Remember that the cost is assigned to a link **through a specific interface**: this means that the same link can have different costs on opposite directions.

Question 9. Change the configuration such that a new path is taken between PC1 and PC2. How would you do it? Is the reverse path from PC2 to PC1 affected by your changes? Justify your results.

4 Designated routers

Let's now concentrate on subnet 10.10.99.0/24 (switched network between R4, R5 and R6). In this particular example of "multiple access" network, a DR (Designated Router) is elected. This DR will be responsible of generating the network LSA for this subnet (otherwise 3 Network LSAs would be generated for the same subnet, by R4, R5 and R6 respectively). You can have more details about the roles of the different routers (DR, backup DR, DROTHER) using the following commands. Use `kathara connect` to connect to R4 and then:

```
telnet localhost ospfd

show ip ospf neighbor

show ip ospf neighbor detail
```

Question 10. Which device is the Designated Router/Backup router? What are the elements involved in the decision process?

4.1 Changing the DR/BDR election process

Here we configure the routers to make the excluded router become the DR. In OSPF, the priority is set at the **interface level**: you will need to select the appropriate interface to modify. In fact, the whole DR/BDR scope is per-interface (a router may have one interface acting as a DR in one link, and as a DROTHER on another link).

To modify the desired router `R[N]`, first clean the lab with `sudo kathara lclean`. Then, open the file located in `$HOME/lab-tp-2/R[N]/etc/quagga/ospfd.conf`.

Finally, at the end of this file, add the following lines:

```
interface  eth[y]

    ip ospf priority 100
```

Now, the next time the DR/BDR election process will take place, the priority level will be higher for the previously discarded router. By default, all routers are initialized with the same priority of 1. Remember to adapt these lines with the correct values for `R[N]` and `eth[y]`. You can now prove your work by restarting the lab and checking the OSPF neighbors of this subnet.

Question 11. Provide the configuration snippet (i.e. the modifications you made) and briefly justify it. Is there another way to influence the DR/BDR election process?

4.2 Changing the priority in real-time

You can also try to change these parameters without closing the lab. In this case, your change will not overwrite the current configuration since a DR/BDR already exists, unless you force it with some hard configuration parameter (e.g., setting a priority of 0 forces a previous DR to relinquish its role because the special value of priority=0 means to never become a DR/BDR). Moreover, if the DR is forcefully removed from a working network, the BDR will automatically take its place (which is the purpose of the backup DR to begin with).

Feel free to explore the configuration console by using the following commands:

```
configure terminal
  interface eth0
    ip ospf priority 0
```

5 Multi-area configuration of OSPF

In this part we will now configure OSPF to have a backbone area including the 4 routers R1-R4 (aka area 0) and a separate area for the multi-access network and the PC3 network.

First of all, we need to have three terminals:

- one terminal continuously pinging between PC1 and PC3;

- a second terminal on R1, live capturing traffic from all interfaces;
- a third terminal on R4, then R5 and R6 where we will configure the network.

5.1 Area-border router

The router R4 requires a special treatment as it is an **area-border router** (ABR): while the other routers are all **internal** to a single area, R4 has some interfaces belonging to different areas, and we need to configure it appropriately.

After you have access to all the required terminals, on the third terminal connect to R4 and start checking the configuration (insert the password when asked, it is always zebra):

```
cd $HOME/tp-lab-2/
kathara connect r1
telnet localhost ospfd
enable
```

Enter in configuration mode:

```
configure terminal
```

Type `?` to explore the commands available. Afterwards, enter in the router configuration with the command

```
router ospf
```

Type `show run` to observe what is the current configuration of the OSPF protocol in the router. You will see a line that includes the network 10.10.0.0/16: this informs that router R1 should activate OSPF on all interfaces belonging to such network as part of the backbone area (0, or 0.0.0.0).

We should now remove this entry and add it as a separate area.

```
no network 10.10.0.0/16 area 0
```

The "no" command negates the result of following commands, or sets the default values. In this case, it removes the network from the advertisements.

> **Question 12.** What happens on router R1? Can you still see the request/reply of the ICMP packets? Is the connection maintained? Explain your findings.

Now, the network 10.10.99.0/24 is removed from the OSPF protocol. In order to assign the correct area, we need to consider each link separately. In practice, you have to run the command:

```
network A.B.C.D/24 area N
```

as many times as the interfaces of R4. This means, for instance, that you will have to assign the area 0 to the network 10.10.14.0/24 which represents the link R1-R4, and so on.

All routers in area 1 must be configured accordingly: run the same steps for R5 and R6. Remember: in R6 you will have to configure two networks: the multi-access network 10.10.99.0/24 and the PC3 network 10.10.3.0/24.

You can see that the ping is now working again.

> **Question 13.** Show the OSPF database, and check the differences w.r.t. previous versions. You will see an additional LSA called **summary LSA**. Show the OSPF database for different routers

in different networks, and try to interpret the results.

6 Extra

This part of the assignment is optional and will not be graded. However, you can have fun by playing with a fully unconfigured network from scratch, and you can test yourself if you are able to make it work!

With the help of the documentation of both kathara and quagga, you can set up another lab which will use ospf6 (which is based on IPv6 rather than IPv4). You will need multiple steps:

- Activate the ospf6d daemon via the zebra.conf configuration file
- For each device, add a file under `r[X]/etc/quagga/ospf6d.conf`
  Attention! The configuration of `ospf6` IS DIFFERENT than simple `ospf`. As a reminder, all protocols in a router are implemented as different processes, and they may have different syntax. The best way is to use some standard samples, such as `https://github.com/6WIND/quagga/blob/master/ospf6d/ospf6d.conf.sample`
- After the lab is started, you will need to manually assign the ipv6 addresses to all the devices and interfaces
- You will need to manually configure the `ospf6d` routers to advertise the networks
- You will need to change the routing cost if you want to have different paths taken in the network.

Question 14. After the configuration, propose potential solution(s) to decrease OSPF convergence times. Your proposal should not require any additional protocol or technology (just plain OSPF). Discuss the potential side effects of your proposal(s).