"Spanning Tree Protocol"

4th Edition. November 2023

Introduction

In this practice we will use the same virtual machine we have used in the second laboratory so, if you do not remember how to run such virtual machine, please check the instructions included in the Wireshark lab document.

This lab is devoted to understand how the STP works, so we will set up a switch to enable this protocol. In IMUNES, a switch is implemented as an Open Virtual Switch (OVS)[1]. You do not need to be an expert using OVS, because this document will provide the required commands to configure an OVS, but you have to understand that all switches will be configured using the local terminal ($). In other words, it is not possible to double-click a switch in IMUNES to configure it.

One ring to rule them all (aka, configuring STP)

Open imunes by running the command "sudo imunes" in a terminal **inside the virtual machine** (password imunes). Let's load the ring-topology (file name *ring-topology.imn*), shown in the figure below (remember this topology is in your directory, under *imunes → Cloud Networks Lab)*:

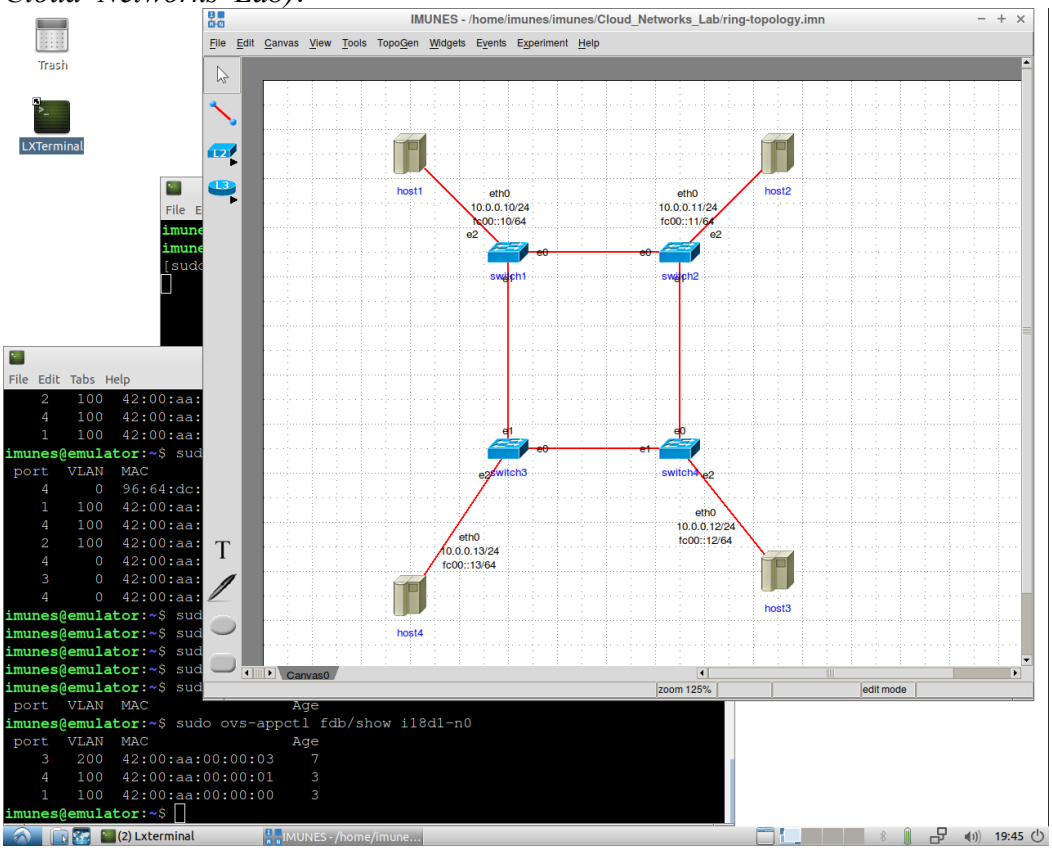

Figure 1. Ring topology

[1] http://www.openvswitch.org/

After the emulation is running (Experiment → Execute) we need to get the experiment identifier assigned by IMUNES to the emulation. The Experiment ID is shown in the bottom-right part of the GUI, but it can be obtained in this other way too:

- Open a new LXTerminal by double-clicking the icon on your desktop. In the terminal, run the following command:
    - $ sudo himage -ln
- This will show the name of the emulation just created by IMUNES. This is important, because all switches identifiers will start with that name followed by the name of the switch. For example, run the following command:
    - $ sudo ovs-vsctl show
- The output should be similar to the one shown in Figure 2, where the identifier of the first switch (switch1) is i3241-n0, with four interfaces, starting with i3211-n0-e0 (in general, switch*X* corresponds to i3241-n(*X-1*))

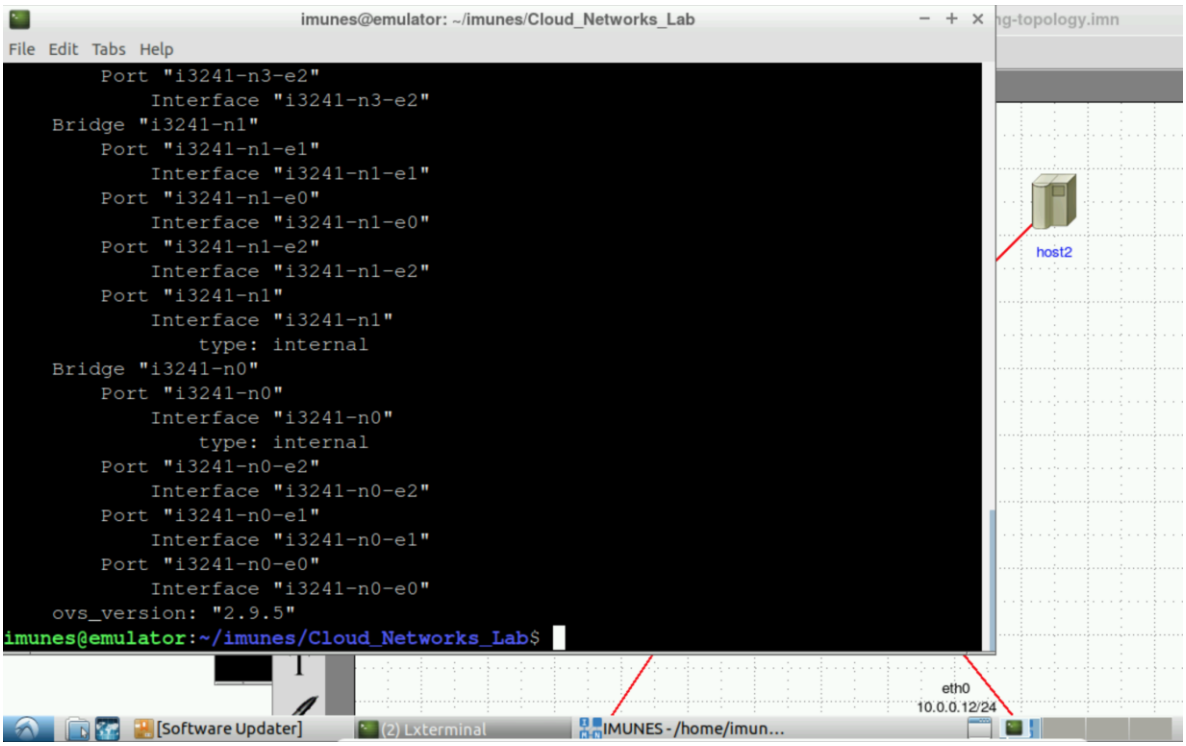

Figure 2. ovs-vsctl show

In topologies with physical loops, learning switches do not work properly, so the Spanning Tree Protocol (STP) has to be enabled.

- In host1, run a ping to host2:
    - host1# ping 10.0.0.11
- Write down (or take a screenshot) the results of the first 10 pings (if any).

Question 1: explain the reason of this result.

- Now, let's enable the STP in every switch, by running the following commands (remember you have to substitute i3241 id with your own id):
  - $ *sudo ovs-vsctl set Bridge i3241-n0 stp_enable=true*
  - $ *sudo ovs-vsctl set Bridge i3241-n1 stp_enable=true*
  - $ *sudo ovs-vsctl set Bridge i3241-n2 stp_enable=true*
  - $ *sudo ovs-vsctl set Bridge i3241-n3 stp_enable=true*

Question 2: repeat the previous pings. What is the main difference between this and the previous execution in question 1? Justify your answer.

- All switches are running the STP now. You can check the status of each switch with the following command (this one shows bridge i3241-n0, so change it appropriately):
  - $ sudo ovs-appctl stp/show i3241-n0
- This command shows **all** switches:
  - $ sudo ovs-appctl stp/show

After reviewing the state of all switches, answer the following questions:
Question 3: which is the root bridge? How do you know that?
Question 4: what is the role of the ports attached to the root bridge?
Question 5: what is the state of the ports attached to the root bridge?

Let's change the root bridge. In order to do that, we will force the selection of a given switch by reducing the stp-priority with the following command (if switch1 was already the root bridge, please change the it properly):

- $ sudo ovs-vsctl set Bridge i3241-n0 other_config:stp-priority=20000

Please, remember these changes may take some time to have effect on the logical topology.

It is also possible to modify the cost of a given link too. Let's modify the cost of the link connecting switch1 and switch2 with the following commands (before running these commands, write down the state of switch2 and switch3):

- $sudo ovs-vsctl set Port i3241-n0-e0 other_config:stp-path-cost=100
- $sudo ovs-vsctl set Port i3241-n1-e0 other_config:stp-path-cost=100

Question 6: which one is the new blocked port in the virtual topology?

- It is time to finish the emulation.

Note: if you find switches whose ID do not belong to the experiment you're running, you can delete them with the following command

- $sudo ovs-vsctl del-br <*bridge-id*>

Wizards and Dwarves

Wizards don't like dwarves, and dwarves don't like wizards[2], so they decide to split their networks. Departing from the physical topology shown in Figure 3 (*simple_topology.imn*), the network admin has to split that physical topology in two VLANs: host1 and host2 will be in the wizards VLAN (id=100) while host3 and host4 will be in the dwarves VLAN (id=200).

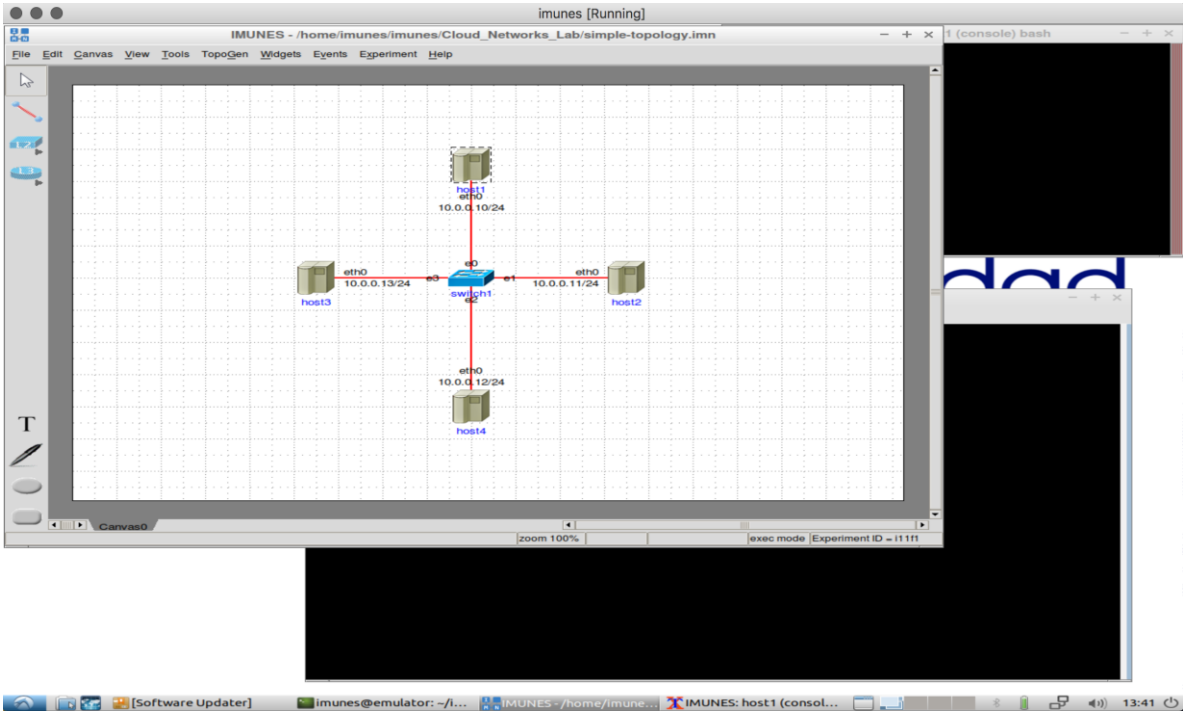

Figure 3. Simple topology

Question 7: show all necessary steps to set up these two VLANs if the following command is used to set up port e0 of switch1 as a VLAN based port with VLAN id equal to 100.

- $ sudo ovs-vsctl set port i3241-n0-e0 tag=100

Optional: check the learning table of any switch using the following command:

- $ sudo ovs-appctl fdb/show i3241-n0

[2] https://github.com/sharkdp/great-puzzles/blob/master/puzzles/wizards-and-dwarves/puzzle.md