

POLITECNICO DI TORINO



**Politecnico  
di Torino**

## **Lab #1 on SDN**

*“Computer network design and control”* module of  
Communication and network systems

Academic Year 2025/26

Andrea Bianco, Paolo Giaccone,  
Alessandro Cornacchia, Matteo Sacchetto

Version: November 19, 2025

© 2025-26

# Chapter 1

## Laboratory #1

The aim of this lab is to experiment with Mininet, an SDN network emulator running on a Virtual Machine (VM) within CrownLabs. Open vSwitch will be used as the software switch within Mininet.

The lab's primary goal is to familiarize students with the emulated Mininet environment and to address basic and advanced routing problems in an SDN scenario. No SDN controller will be used, and no central routing policy will be defined. Instead, forwarding tables that enable proper routing schemes will be manually installed using SDN switch configuration commands.

The lab follows a sequence of incremental steps, with each step preparatory to the following ones. As such, it is strongly recommended to proceed step by step.

### 1.1 Starting the Lab

1. Navigate to the CrownLabs website: <https://crownlabs.polito.it/>
2. Click on the "Login @Polito" button.
3. On the login page, click on the "PoliTo SSO" button at the bottom of the form.
4. Log in using your PoliTo credentials (the same credentials used to access the "Portale della didattica").
5. Once logged in, you will be on the "Dashboard" tab. You should see the "Computer Network Design" workspace on the left side of the user interface (UI).
6. Click on that workspace to reveal a new section containing the VM called "Lab".
7. Click the "Create" button to instantiate the VM. Once creation is complete, the "Connect" button will become active.
8. When the "Connect" button becomes active, click it. A new browser tab will open, connecting you to the VM desktop.

#### 1.1.1 What to Do if You Close CrownLabs by Mistake

If you accidentally close the web browser or browser tab, **don't panic!** The VM instance is still running, and you can simply reconnect to it.

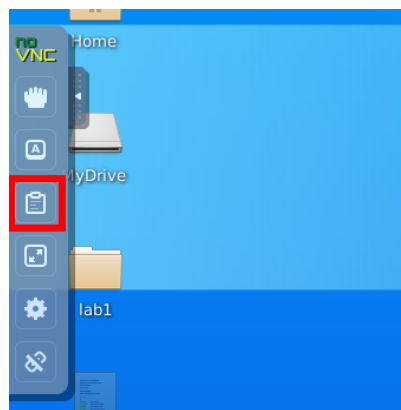
1. Follow the steps in Section 1.1 up to step 6.

2. Once you reach the VM list, you will find the previously created VM instance under the “Lab” VM.
3. Click the “Connect” button again, and you will return to where you left off.

### 1.1.2 How to Copy-Paste Between CrownLabs and Your PC

Copying and pasting between your PC and the VM instance requires some additional steps beyond the usual `CTRL+C` and `CTRL+V`.

1. On the left side of the UI, you should see a small tab with an arrow. If it is not already open, click on the tab to open it.
2. Once open, you will see a small menu titled “noVNC” at the top, with a list of icons below.
3. The clipboard icon is shown in the figure below (typically the second or third icon).



4. Click the clipboard icon to open a text area (click again to close it). This text area serves as the interface for copying and pasting between CrownLabs and your PC.
5. When you select text in CrownLabs, it will automatically appear in this clipboard. From there, you can copy and paste it to your PC as usual.
6. To copy text from your PC and paste it into the VM, first paste it into this clipboard. Then, you can paste it wherever needed within CrownLabs.

### 1.1.3 Sharing Files Between CrownLabs and Your PC

If you need to upload a file to the VM or download a file from it, you can do so through the CrownLabs drive functionality. You have 1 GB of drive space, which can be accessed in two ways:

- **From the VM:** Open the file manager (click with the right button of the mouse on the VM desktop and select the Applications item in the popup menu, then choose File Manager) and select “MyDrive” device.
- **From the CrownLabs UI:** Select the “Drive” tab at the top of the UI.

Files uploaded to the “Drive” functionality will be available from within the VM and viceversa. Only files stored on this drive will be persistently saved. Any other file created or modified within the VM will be deleted once the VM instance is deleted.

### 1.1.4 Stopping the VM Instance

Once you finish the lab, please stop the VM instance to avoid wasting CrownLabs resources. Before deleting the VM, transfer all files you wish to keep to the “MyDrive” directory (as described in the previous subsection).

1. Close the tab with the VM desktop.
2. Navigate back to CrownLabs by repeating steps 1–6 from Section 1.1.
3. Next to the “Connect” button of your VM instance, you will see a trash bin icon. Click on it to open a popup window.
4. In the popup window, confirm the deletion by clicking the “Delete” button.

Before beginning the lab, note that you have a reference file with a list of useful Linux and Mininet commands. Keep this reference handy while doing the lab.

## 1.2 Overview of Shell Commands

As a preliminary requirement, you must become familiar with both the Linux terminal and the `gedit` editor. Both are available from the menu that opens when you click on the icon located at the top left of the desktop. The most useful terminal commands are listed below. Practice with each command at least once before proceeding to the next steps.

**NOTE:** For additional details on what each command does and which options are used, please refer to the Mininet command reference file available on the course website.

Open a terminal using one of the following methods:

- Click the top left icon, and when the window opens, select the terminal emulator icon.
- Right-click on the desktop and select “Open Terminal Here” from the pop-up menu.

Remember two important tips when issuing commands in the terminal:

- Use the TAB key to autocomplete file names and commands.
- Use the UP and DOWN arrow keys to scroll through previously entered commands.

Try the following commands:

1. `man` to display the manual for a command. E.g., `man ls`
2. `pwd` to display the current directory.
3. `cd` to change directory. E.g., `cd /home/netlab/Desktop/lab1`
4. `ls` to list folder contents. E.g., `ls -l`
5. `gedit` to create or edit a file. Type `gedit readme.txt`, write something in the file, save, and close it.
6. `cat` to print file contents. E.g., `cat readme.txt`
7. Type `command1 && command2` (adding `&&` between two commands) to run commands in sequence. `command2` will execute only if `command1` completes successfully. E.g., `pwd && ls`
8. `ifconfig` to display network interface card configuration.

## 1.3 Step 1: Getting Familiar with the Environment

1. Open a new terminal window.
2. We will be using a network performance tool called `iperf3` to generate synthetic traffic in our network. Read the tool's documentation by typing `man iperf3`.
3. We will be emulating a real network using Mininet. Instead of transferring packets over physical wires, Mininet creates virtual network interfaces and emulates communication between them. The performance in terms of bitrate of such an emulated network highly depends on ... (see question below) of the host machine.

Run the command `sudo mn --test iperf`.

This command first starts, as root user, a Mininet instance with a simple topology comprising two hosts interconnected via a switch. Then, Mininet performs an `iperf` test between the two hosts and outputs the average data rate.

Check the result of your test. You should observe output similar to:

```
Results: ['x bits/sec', 'y bits/sec']
```

What is the meaning of the two values? Hints? What do they depend on?

Repeat the test 5 times. Write your results:

Now run a CPU-intensive application on your system (e.g., a YouTube video or execute `stress-ng --cpu 2 --io 2 --timeout 50` in a new terminal) and repeat the five tests.

Write your results:

Comment on the results:

## 1.4 Step 2: Single Switch Topology

1. Before continuing the lab, in the terminal run the following command:

```
xrdb -merge /home/netlab/Desktop/.Xresources
```

2. We start with a simple topology.

In the terminal, run this chain of commands:

```
sudo mn --clean && sudo mn --topo=single --switch=ovsk --controller=none --mac --arp
```

The startup may take a few seconds. You will know the network has successfully started when you see the following line:

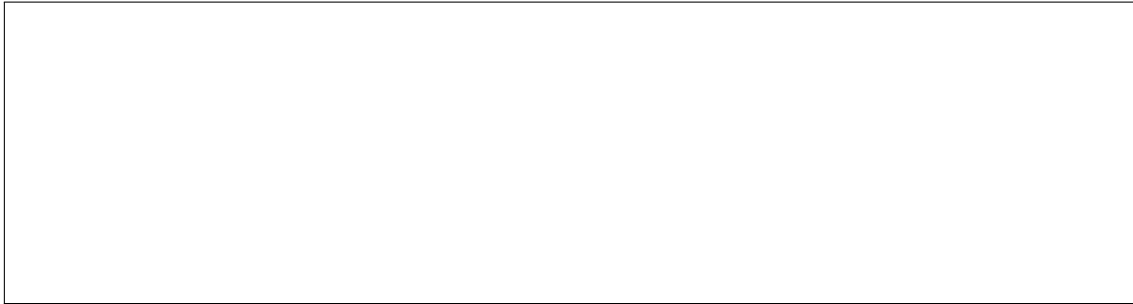
```
mininet>
```

This is the command line interface (CLI) of Mininet, which we will use to interact with the emulated network.

3. Inside the Mininet CLI, type `help`. This displays a message with all available Mininet commands alongside hints about their usage. To get detailed information about a particular command, type `help [command]`.
4. The first topology contains a certain number of devices: hosts and switches interconnected via links, with no controller. Use the command `nodes` to identify the available devices. Hosts are denoted as `h{ID}` (e.g., `h1`, `h2`), while switches are denoted as `s{ID}` (e.g., `s1`, `s2`).
5. Each switch port (e.g., `s1-eth1:1`) is associated with a name (`s1-eth1`) and a port ID (an integer). These IDs are internally used by switches to define the network interfaces for packet forwarding. Use the command `ports` to find the association between each network interface and its corresponding name/port ID.

Report all the ports below:

Using the `links` command, draw the test topology, including port names and IDs.



6. Use the `ifconfig` command to view information on the network interfaces of each node. This can be done from the Mininet CLI using `[node name] ifconfig`, or by opening an individual terminal inside each node with the `xterm [node name]` command (e.g., `xterm h1`) and running `ifconfig` from inside the node's xterm.

**NOTE:** Each network node contains a loopback interface `lo` for testing purposes, which is always associated with port ID 0. You can ignore this port.

Discover the IP addresses of all hosts. What is the corresponding port ID for each network interface of each node? Redraw the full topology with all ports and IP addresses.



7. Inside the Mininet CLI, check the connectivity between `h1` and `h2` by executing either `h1 ping h2` in the Mininet CLI or by opening a terminal for host `h1` using `xterm h1` and executing `ping [h2 destination IP address]` in the xterm. Press `CTRL+C` to terminate the test. Was the test successful? Why? Compare with `h1 ping h1`.

Hint: Recall the content of an SDN switch when firstly activated.

Dump the SDN switch flow table by using the command:

```
sh ovs-ofctl dump-flows s1
```



8. Define the SDN switch flow table to route traffic based on the traffic arrival port (**port-based forwarding**). For example, to instruct switch `s1` to forward all traffic from port 2 to port 1, use the following command (launched within the Mininet CLI):

```
sh ovs-ofctl add-flow s1 in_port=2,actions=output:1
```

Dump the switch flow table.

How many entries are needed in the `s1` flow table to make ping work? Keep executing ping, adding rules and check the installed flow rules until it works.

Report below the rules:

9. Now configure the forwarding table to route traffic based on IP addresses (**IP-based forwarding**). For example, to route based on the destination IP address, use the command:

```
sh ovs-ofctl add-flow s1 dl_type=0x800,nw_dst=10.0.0.1/32,actions=output:1
```

What is the meaning of each option used in the previous command?

10. Before adding the new entry above to the forwarding table, empty the forwarding table using the command: `sh ovs-ofctl del-flows s1`.

Report the installed flow rules and test host connectivity. Add rules until the test works properly.

11. Use the `tshark` command to run a network protocol analyzer that captures packets. Run `xterm h1 h2 s1` to open terminals on the switch and on the two hosts.

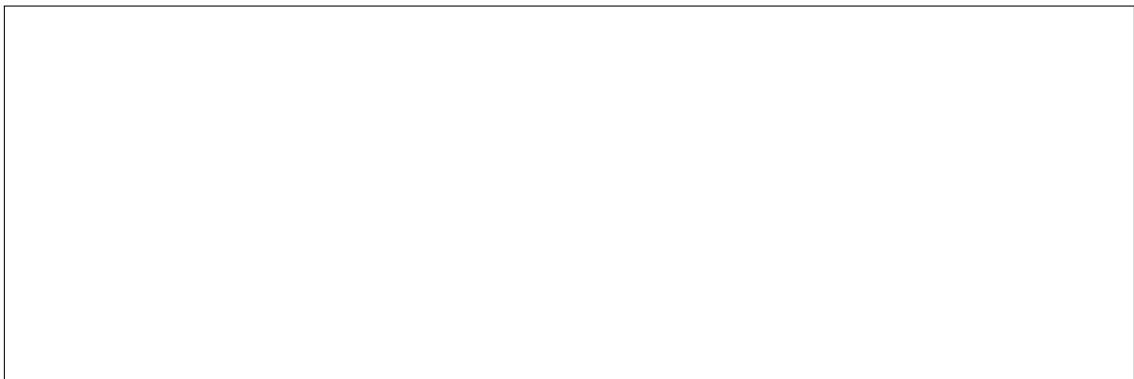
In the `s1` terminal, run:

```
tshark -i s1-eth1 -q -z ip_srcdst,tree -z conv,udp -z conv,eth
```

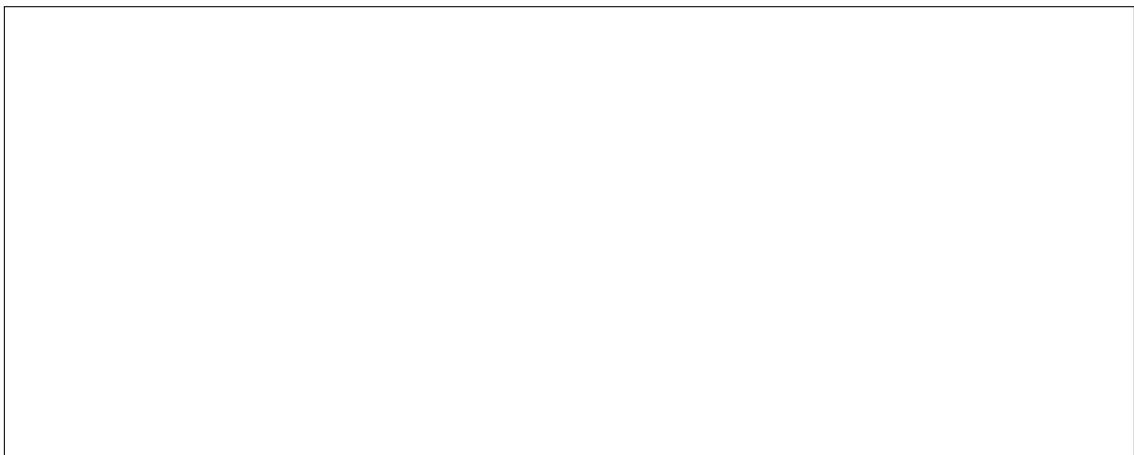
Run `iperf3 -s` as a server in h2 and run `iperf3 -c 10.0.0.2 -u -b 100k` as a client in h1. When `iperf3` ends in h2, you will see a concise summary of the statistics in both h1 and h2. Press `CTRL+C` in the s1 terminal to stop `tshark`. Report a summary of the information provided by `iperf3` and `tshark`.



12. Use `man tshark` and `man iperf3` to determine the options used for the above `tshark` and `iperf3` commands, and briefly report their meaning.



13. Compare and report the information above with what you obtain by running `ifconfig` on the hosts and on the switch.



14. Use the `exit` command to stop the network.

## 1.5 Step 3: Two Switches Topology

1. In the terminal, run this command to start the new network:

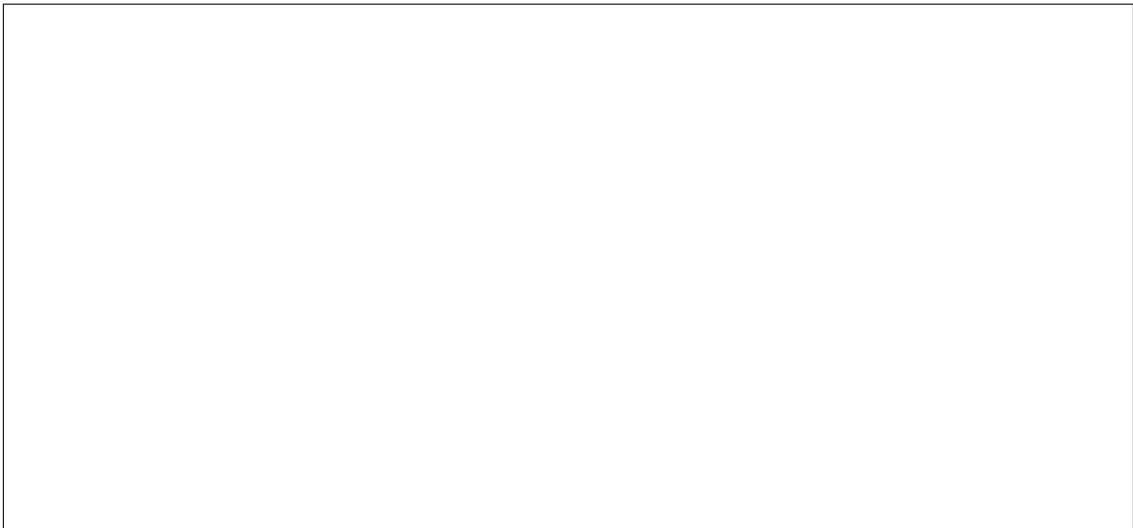
```
sudo mn --clean && sudo mn --topo=linear --switch=ovsk --controller=none --mac --arp
```

2. Discover and draw the topology with all port numbers.



3. Check the connectivity between the hosts.

What should be done to enable communication between the hosts? Report the needed commands.



4. Use the `exit` command to stop the network.

## 1.6 Step 4: Network with Multiple Switches

1. To start the new network, run the following command in the terminal:

```
sudo mn --clean && sudo mn --custom=/home/netlab/Desktop/lab1/topos.py \  
--topo=topo3 --controller=none --switch=ovsk --mac --arp
```

2. Discover and draw the topology with all port numbers and device addresses.



3. In the previous step, routing was established defining flows on the basis of either the incoming port or the IP address. Do you think the two rules are equivalent in this scenario? Motivate your answer.



4. Define the flow tables to properly route traffic between hosts `h1` and `h2`. Report the flow tables of the involved switches below.



5. Define the flow tables to properly route traffic among all available hosts and check host connectivity. Write the complete flow tables below.



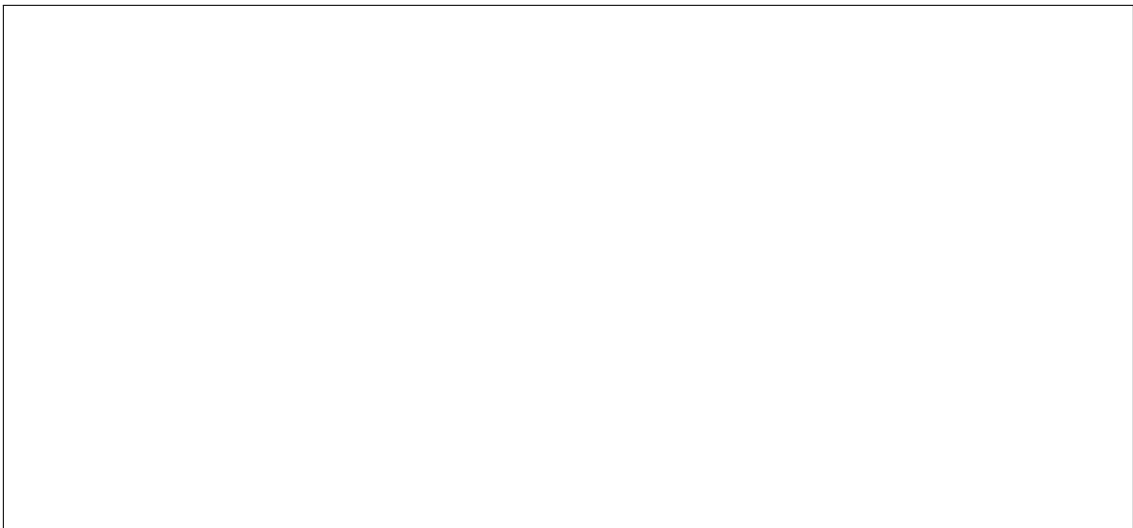
6. Use the `exit` command to stop the network.

## 1.7 Step 5: Multipath Routing

1. To start the new network, run the following command in the terminal:

```
sudo mn --clean && sudo mn --custom=/home/netlab/Desktop/lab1/topos.py \  
--topo=topo4 --controller=none --switch=ovsk --mac --arp
```

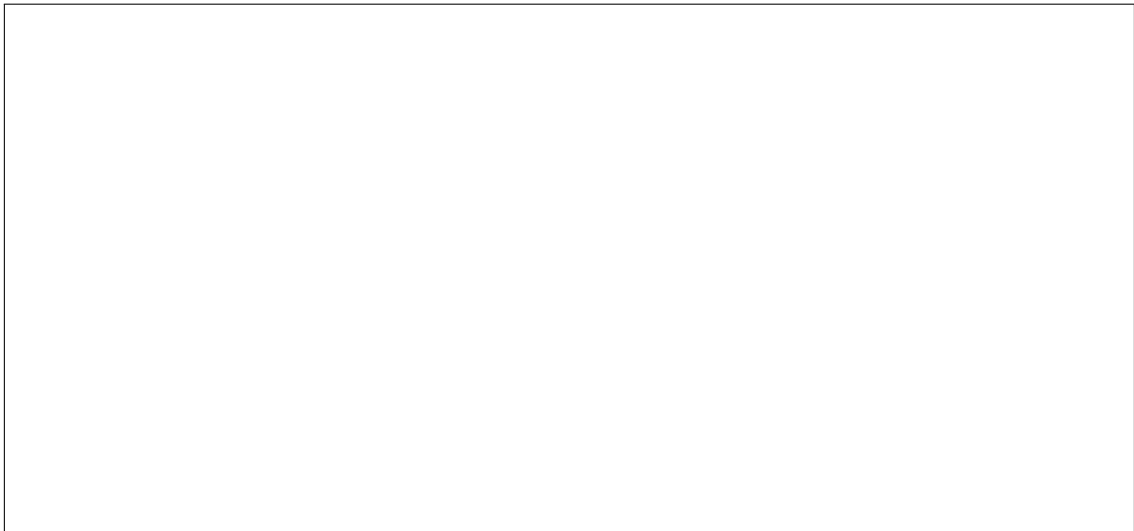
2. Discover and draw the topology with all port numbers.



3. How many paths exist between the hosts? For each switch, define the proper flow rules such that TCP and UDP flows are routed through different distinct paths. This is an

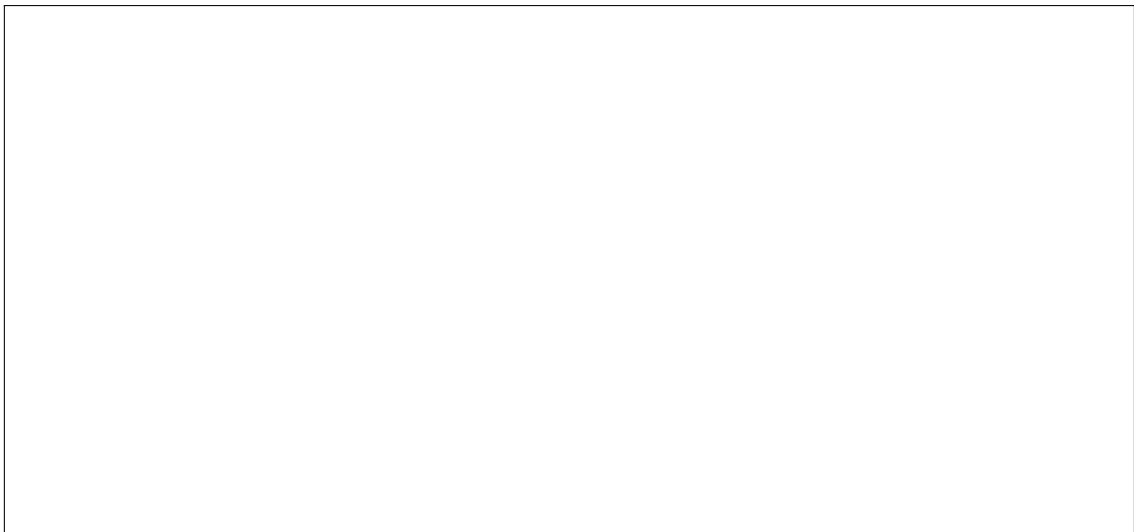
example of protocol-based forwarding.

(HINT: You need to use `nw_proto=<proto>` in the flow rule as well to distinguish between TCP and UDP in the switch.)



4. How can you prove that the multipath routing works as expected?

**HINT:** Inspect the switch interface using `tshark` or `ifconfig`.



5. Use the `exit` command to stop the network.

## 1.8 Step 6: Dynamic Routing

1. To restart the same network as in Step 5, run the following command in the terminal:

```
sudo mn --clean && sudo mn --custom=/home/netlab/Desktop/lab1/topos.py \  
--topo=topo4 --controller=none --switch=ovsk --mac --arp
```

2. Run `sh /home/netlab/Desktop/lab1/routes.sh` in the Mininet CLI. This script will install the needed flow tables in `s1`, `s2`, and `s3` to provide a bidirectional route on the `H1-S1-S2-S3-H2` path.

3. Check the rules installed in the mentioned switches and report the rules below:

4. Open an xterm terminal for h1 and run `ping -i 0.1 10.0.0.2`. When you stop ping, you will get statistics about the number of lost packets (if any). What does the `-i` option do? Are losses experienced?

5. Now run ping again and in real time (i.e., while the ICMP traffic flows) install the required flow rules such that the ICMP traffic flow is redirected to the H1-S1-S4-S3-H2 path.
6. Check the rules installed in the mentioned switches and report them below:

7. Did you experience any loss when the route was changed? Report the number of lost packets (if any) and explain whether losses are permanent or temporary, and why.

8. Now, two cases can occur:

- If losses were experienced, what is the reason? How can the problem be solved? Report the revised flow commands that avoid such losses. Check that no losses are now experienced.
- If you did not experience any loss, what is the reason? What would be an incorrect

way to reroute traffic that would produce losses? Report the corresponding flow commands and check if losses are experienced.



9. Use the `exit` command to stop the network.