UNIVERSITÀ DI TRENTO

On-demand slicing

Mauro Meneghello
Luca Boschiero
Nicola Turniano

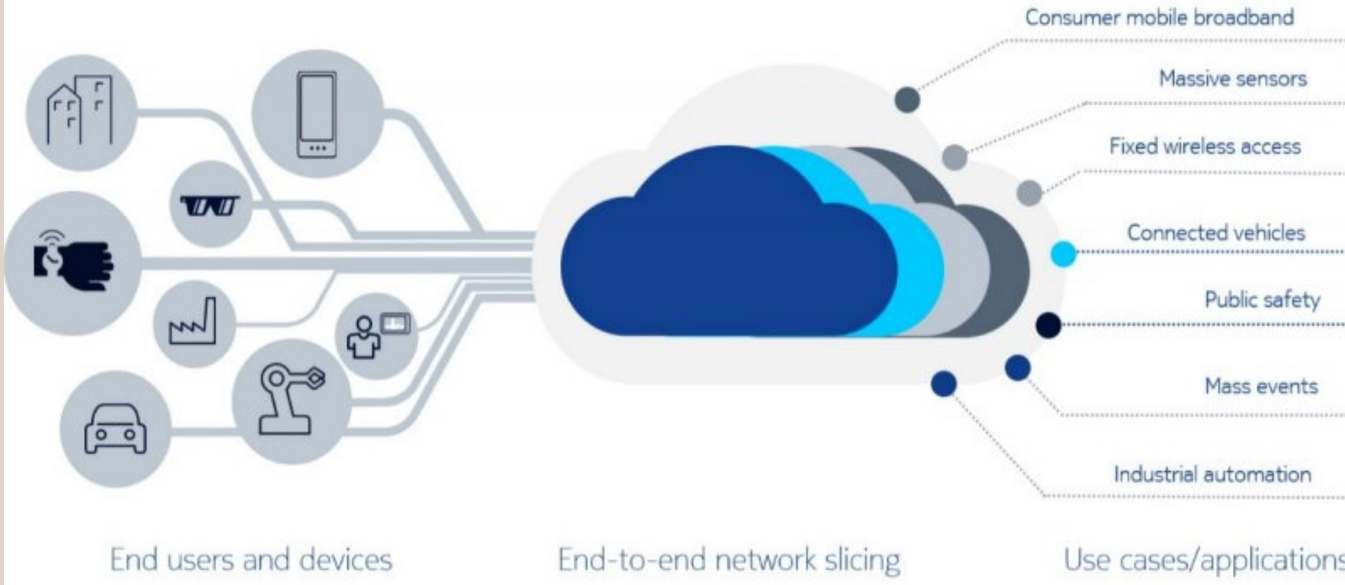
Introduction

Main goal:

- to implement a network slicing approach to enable dynamic activation and de-activation of network slices via CLI command and GUI

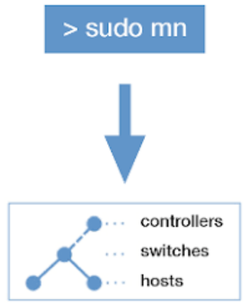
Definition of **network slices**

- permits multiple logical, self-contained networks on a common phisical infrastructure
- offers dedicated resources and a customized network operation.
- supports flexible, on-demand, provision of network resources, network functions and applications.



What do we use?

01



Mininet

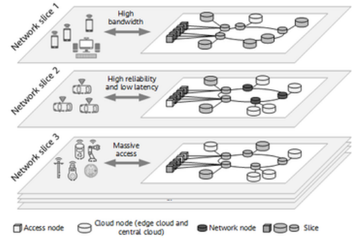
02

Ryu controller

03

OpenFlow

04



Slices and virtual queues

Mininet

A network emulator which creates realistic virtual network.

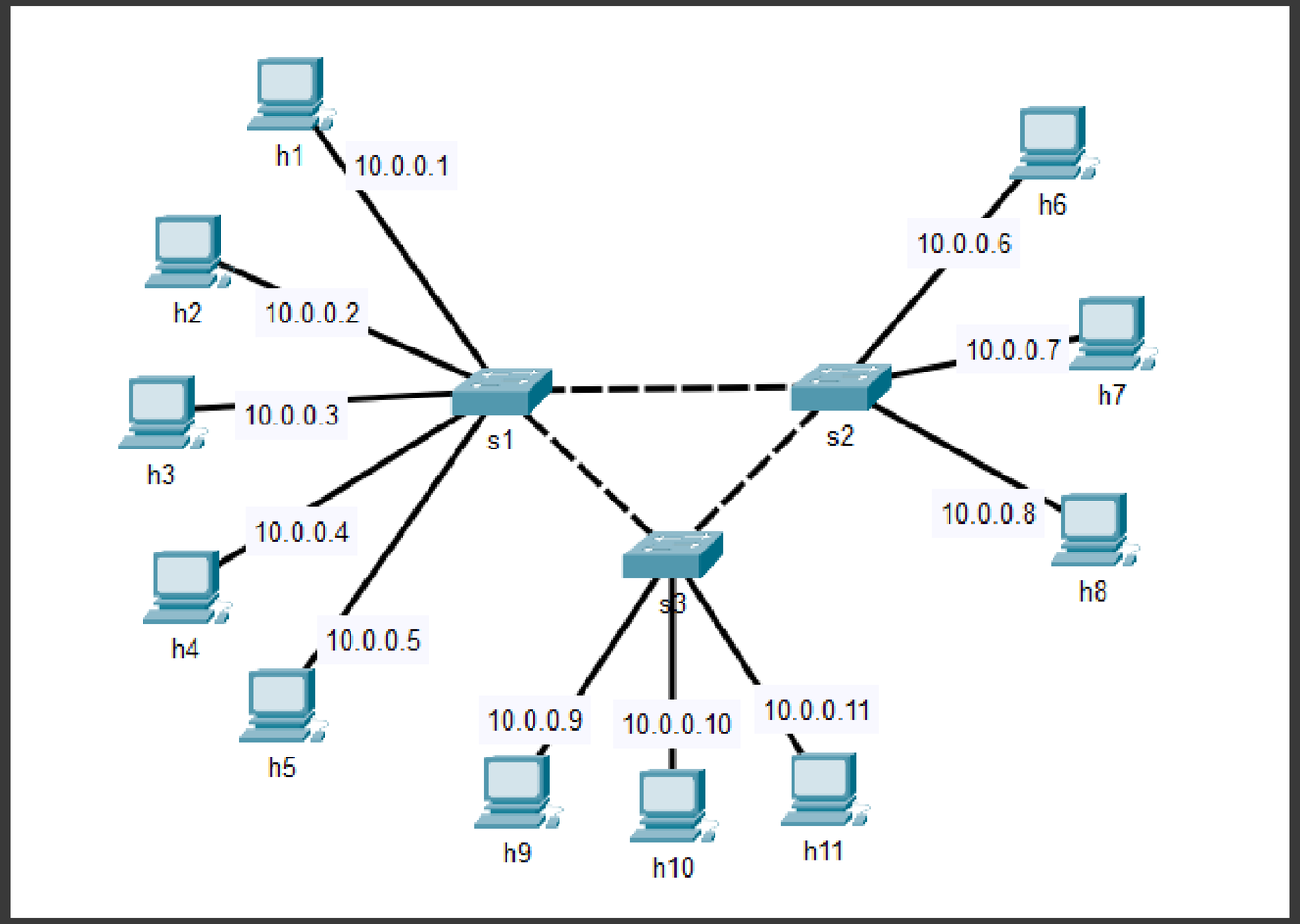
Main commands:

- *addSwitch*
- *addHost*
- *addLink*

- Mininet constructor

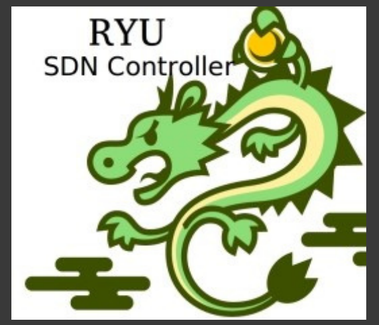
How to run:

```
$ sudo python3 topology.py
```



Topology.py

Ryu controller



Ryu provides software components with well defined API's that make it easy for developers to create new network management and control applications.

Ryu configures the controller by using OpenFlow protocol.

How to run:

```
$ ryu-manager slicing.py
```

The previous command launches the ryu controller and the thread that permits to activate/de-activate the slices.

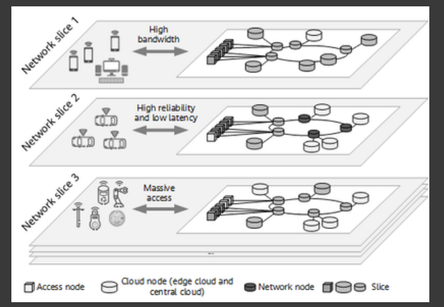
OpenFlow protocol

The OpenFlow (OF) protocol is a standard in software-defined networking (SDN) architecture.

This protocol defines the communication between an SDN controller and the network devices.

On packet arrival, match the header fields with flow entries in a table; if any entry matches, update the counters indicated in that entry and perform indicated actions, otherwise contact controller

Slicing and virtual queues



To implement network slicing, we need to define different **virtual network topologies**, where topology and capacity differ from slice to slice.

To this aim we defined a different file for each slice where:

- we mapped each slice to different virtual queues, that permit also to control throughput.

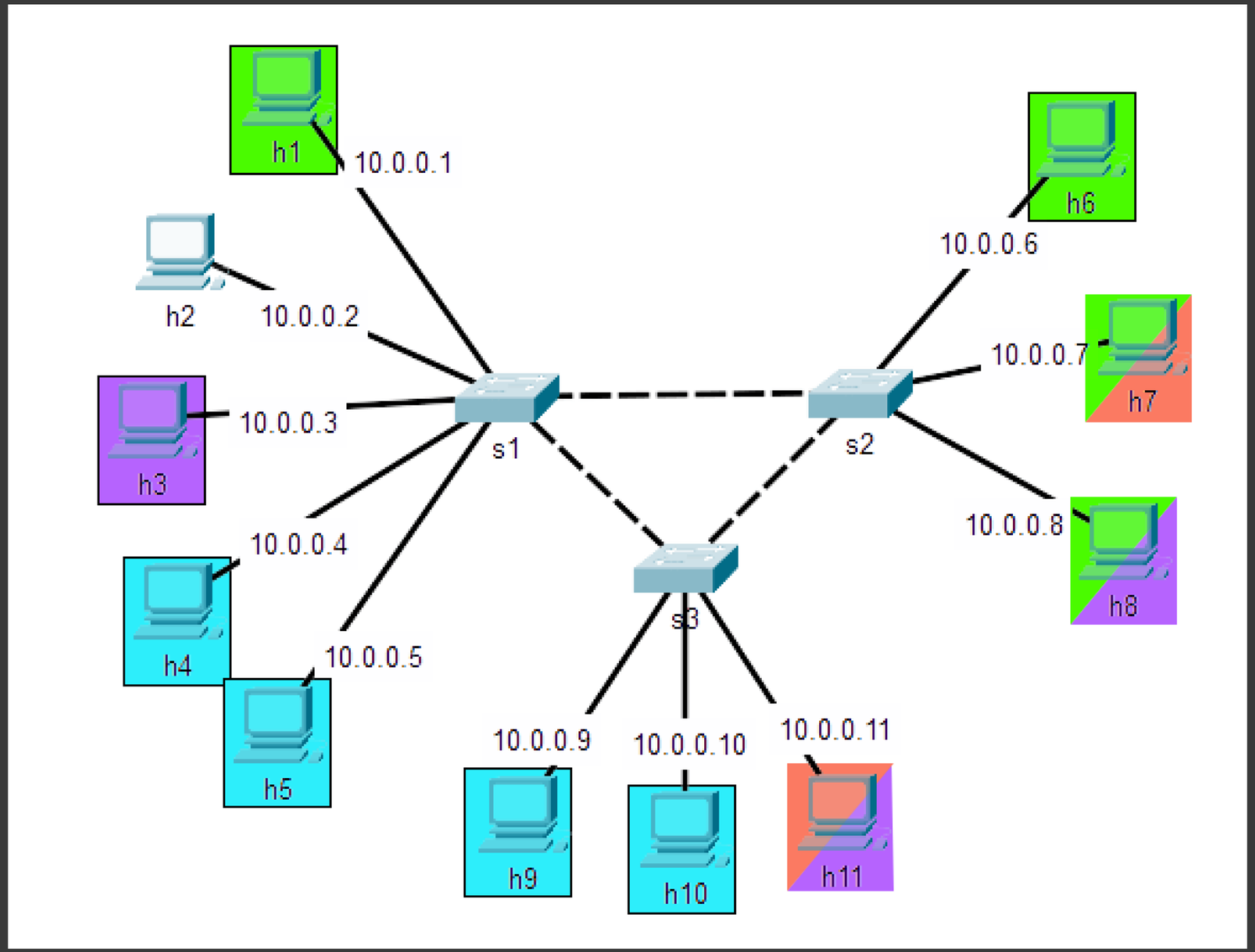
- we associated flows with of protocol to the right queues, permitting and denying communication between hosts.

To check which flows are installed:

```
$ sudo ovs-ofctl dump-flow SWITCH_NAME
```

Network topology with slices

- Slice 1 -> hosts 1, 6, 7, 8
- Slice 2 -> hosts 4, 5, 9, 10
- Slice 3 -> hosts 3, 8, 11
- Slice 4 -> hosts 7, 11



Files
in the repository

topology.py
python script to build the network

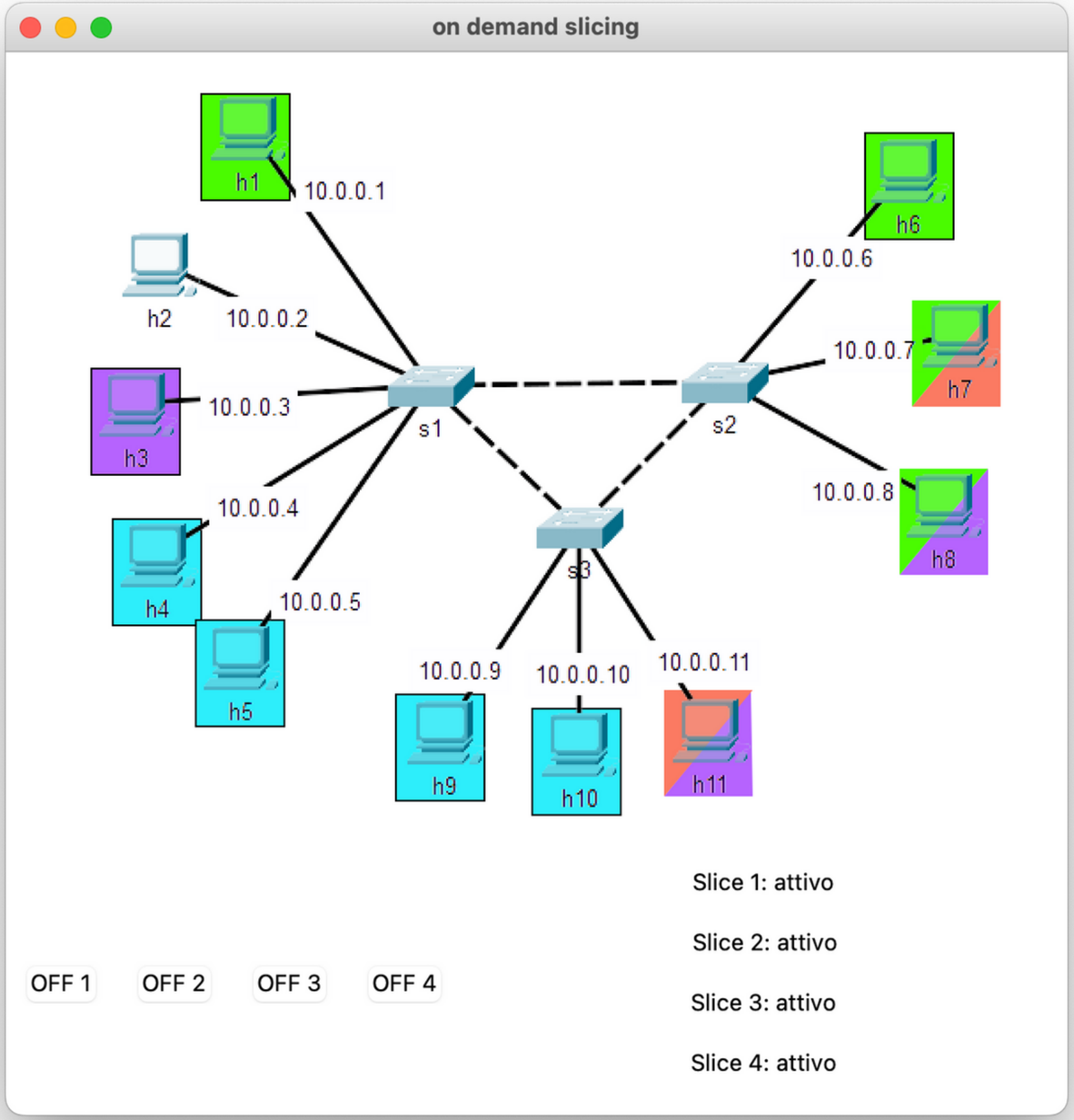
slicing.py
python script that permits to activate/de-activate slices via CLI commands

gui.py
python script that permits to activate/de-activate slices via GUI

on demand slicing

initial_scenario.sh
bash script that deny hosts to communicate with each other

sliceN.sh
bash scripts that build virtual queues that are used for communication between hosts



GUI

- activate/de-activate the chosen slice(s) by clicking on the respective button
- the hosts are coloured to easily recognise each slice

CLI

- activate/de-activate the chosen slice(s) by writing on the console ON or OFF and the number of the slice