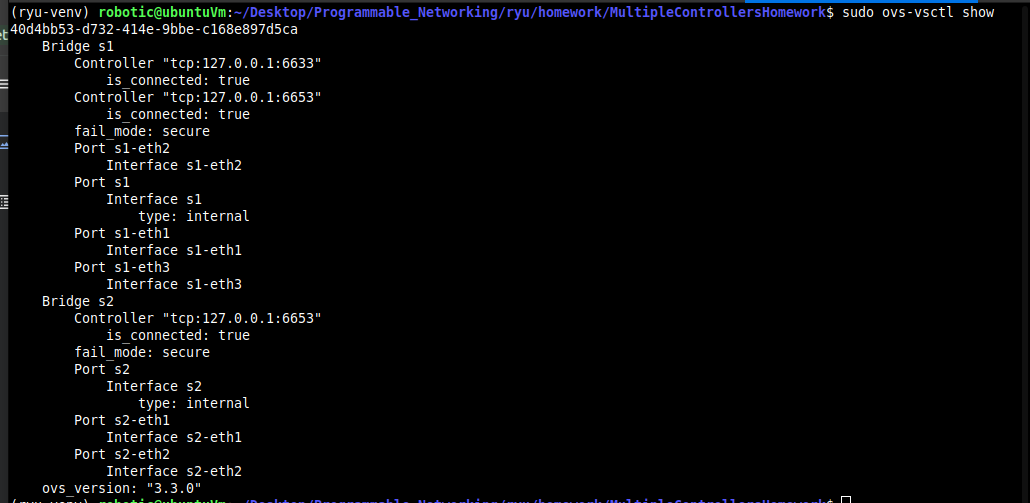
**MULTIPLE CONTROLLERS PROGRAMMING**

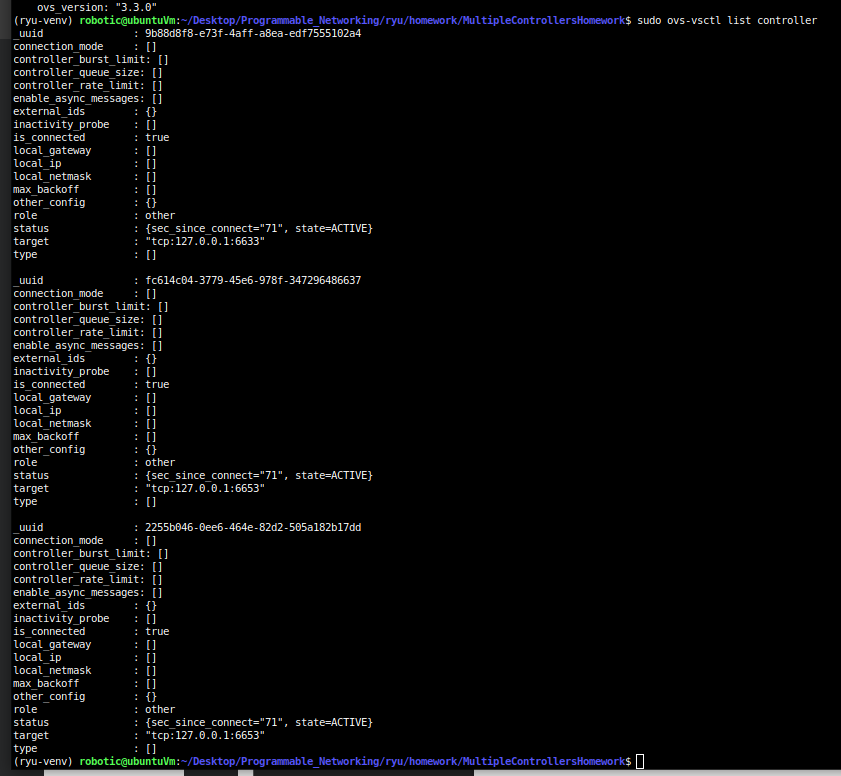
1. STEP1: The default situation is that the controller are in EQUAL mode

after creating the switch and connected each to the switch, this is the situation. As shown the switch is connected to the switch on port 6633 for the first one and 6653 for the second.

***SWITCH VIEW:***



***CONTROLLER VIEW:***

****

1. **STEP2**

**(ryu-venv) robotic@ubuntuVm:~/Desktop/Programmable\_Networking/ryu/homework/MultipleControllersHomework$ curl -X POST -d '{**

**"dpid": 1,**

**"role": "SLAVE"**

**}' http://localhost:8081/stats/role**

**(ryu-venv) robotic@ubuntuVm:~/Desktop/Programmable\_Networking/ryu/homework/MultipleControllersHomework$ curl -X POST -d '{**

**"dpid": 1,**

**"role": "MASTER"**

**}' http://localhost:8080/stats/role**

**(ryu-venv) robotic@ubuntuVm:~/Desktop/Programmable\_Networking/ryu/homework/MultipleControllersHomework$ curl -X POST -d '{**

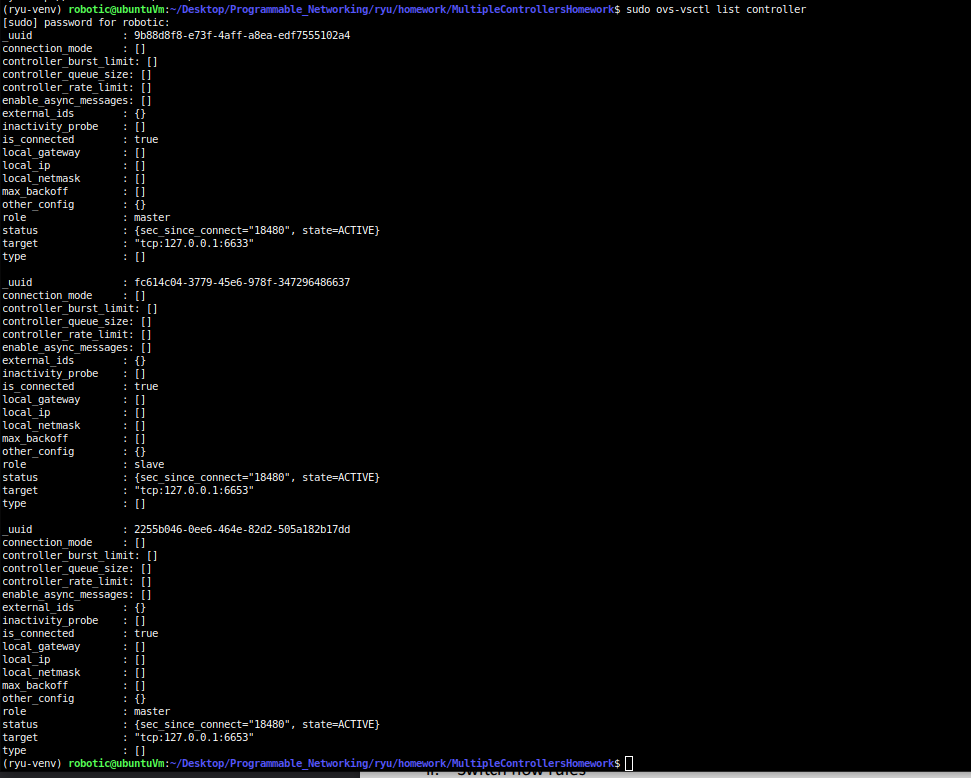
**"dpid": 2,**

**"role": "MASTER"**

**}' http://localhost:8081/stats/role**

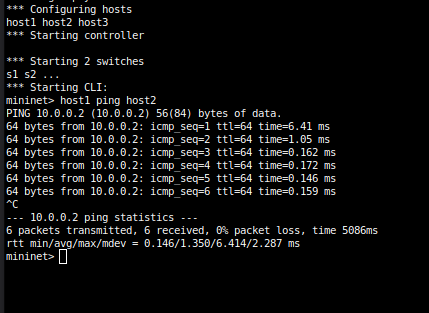
**(ryu-venv) robotic@ubuntuVm:~/Desktop/Programmable\_Networking/ryu/homework/MultipleControllersHomework$**

**STEP3 IMAGE**

****

**STEP 4)**

**IMAGE:**

****

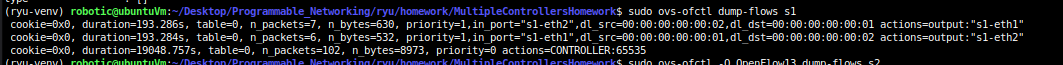
1. **Does ping work? yes**

**b. Why?**

**Perché host1 e host2 sono nello stesso dominio di broadcast connessi a sw1.**

* **sw1 è gestito da C₀, che è MASTER per quello switch.**
* **Il controller MASTER installa regole di forwarding sugli switch.**
* **Quindi i pacchetti ICMP tra h1 e h2 non hanno bisogno di attraversare sw2 né di passare da C₁.**
* **In altre parole: la comunicazione resta confinata a sw1, sotto il controllo diretto di C₀ → i flussi vengono gestiti correttamente e il ping passa.**

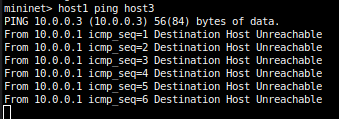
**c. Support your comments by dumping and taking a screenshot of OpenFlow rules on sw1:**

****

**STEP 5**

1. **yes**
2. **because the flow table is innested into s1 so the packet pass throw the injected rules.**

**step 6**

1. ****

**2) the answer to the following questions**

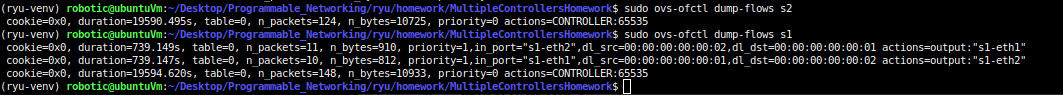
**a. Does ping work? no**

**b. Why? Ping h1–h3 non funziona.**

**Perché sw1 non ha un controller in MASTER, quindi non vengono installate nuove regole per inoltrare i pacchetti verso sw2.**

**Le flow table lo dimostrano: regole presenti solo per h1–h2 su sw1, nessuna per h1–h3; sw2 non riceve pacchetti da h1.**

**Stato controller: C₀ spento, C₁ attivo (MASTER su sw2, SLAVE su sw1).**

****

**STEP 7) SETTO A MASTER IL C1 PER S1**

**(ryu-venv) robotic@ubuntuVm:~/Desktop/Programmable\_Networking/ryu/homework/MultipleControllersHomework$ curl -X POST -d '{**

**"dpid": 1,**

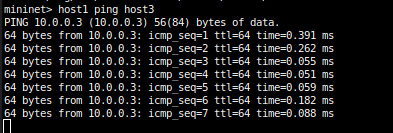
**"role": "MASTER"**

**}'** [**http://localhost:8081/stats/role**](http://localhost:8081/stats/role)

**STEP 8)**

**Provide:**

**1. a screenshot of Mininet CLI of the ping:**

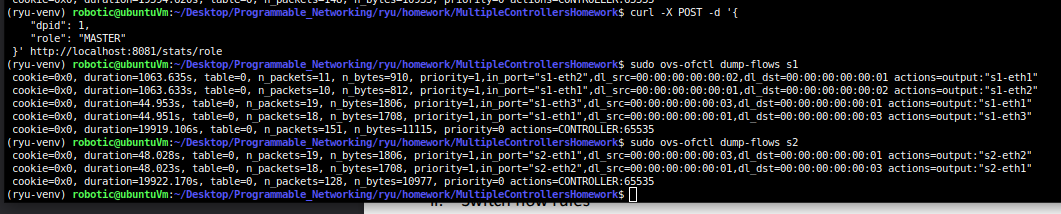
****

**2. the answer to the following questions**

**a. Does ping work? YES**

**b. Why? because now c1 is master so inject the rules.**

**c. Support your comments by dumping and taking a screenshot of OpenFlow rules on sw1 and sw2. (rules were updated.)**

****