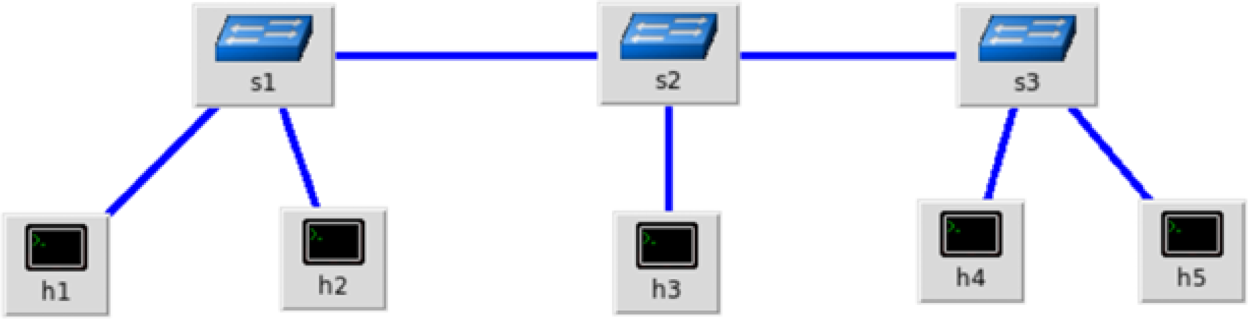
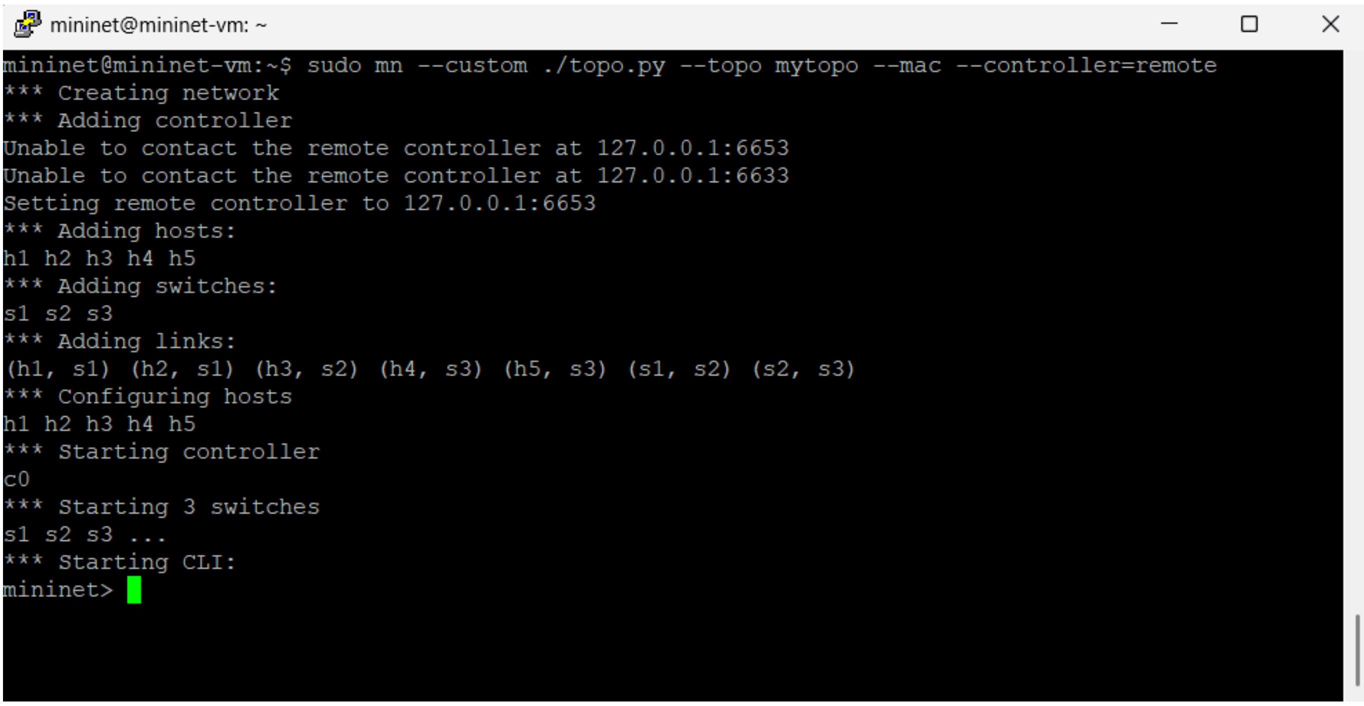
Exercício 2

> Criar um código python para a topologia



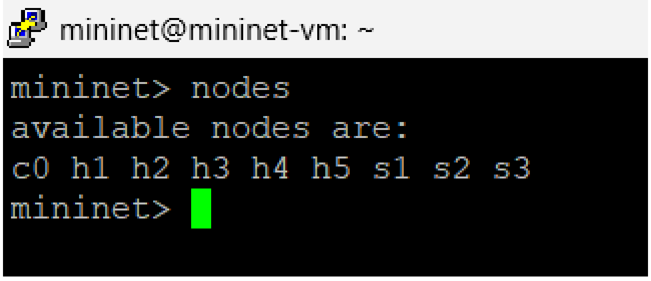
a) Criar uma topologia customizada considerando controlador manual e MAC padronizado

Arquivo python criado para a topologia (topo.py)

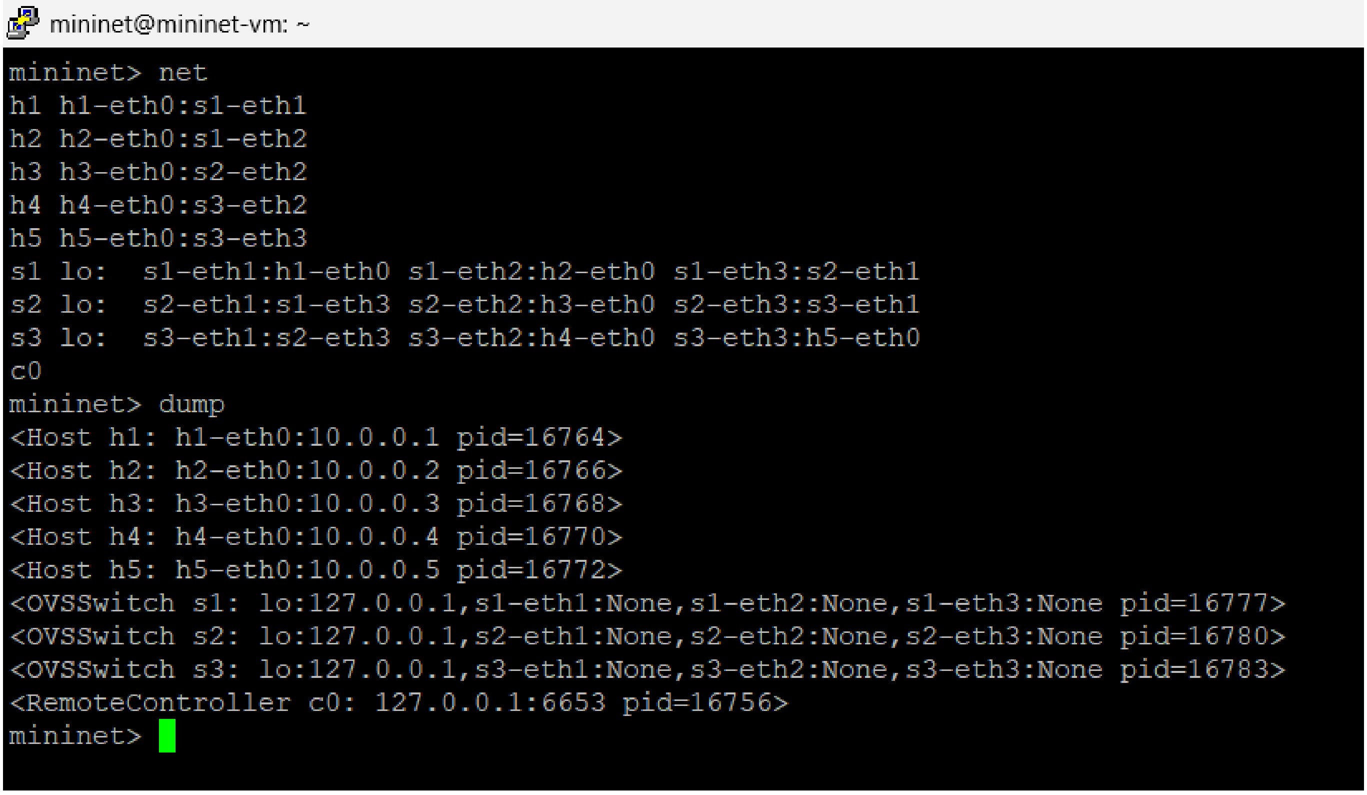


```
mininet@mininet-vm:~$ sudo mn --custom ./topo.py --topo mytopo --mac --controller=remote
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6653
Unable to contact the remote controller at 127.0.0.1:6633
Setting remote controller to 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3 h4 h5
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s1) (h3, s2) (h4, s3) (h5, s3) (s1, s2) (s2, s3)
*** Configuring hosts
h1 h2 h3 h4 h5
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
```

b) Inspeção das informações da rede:



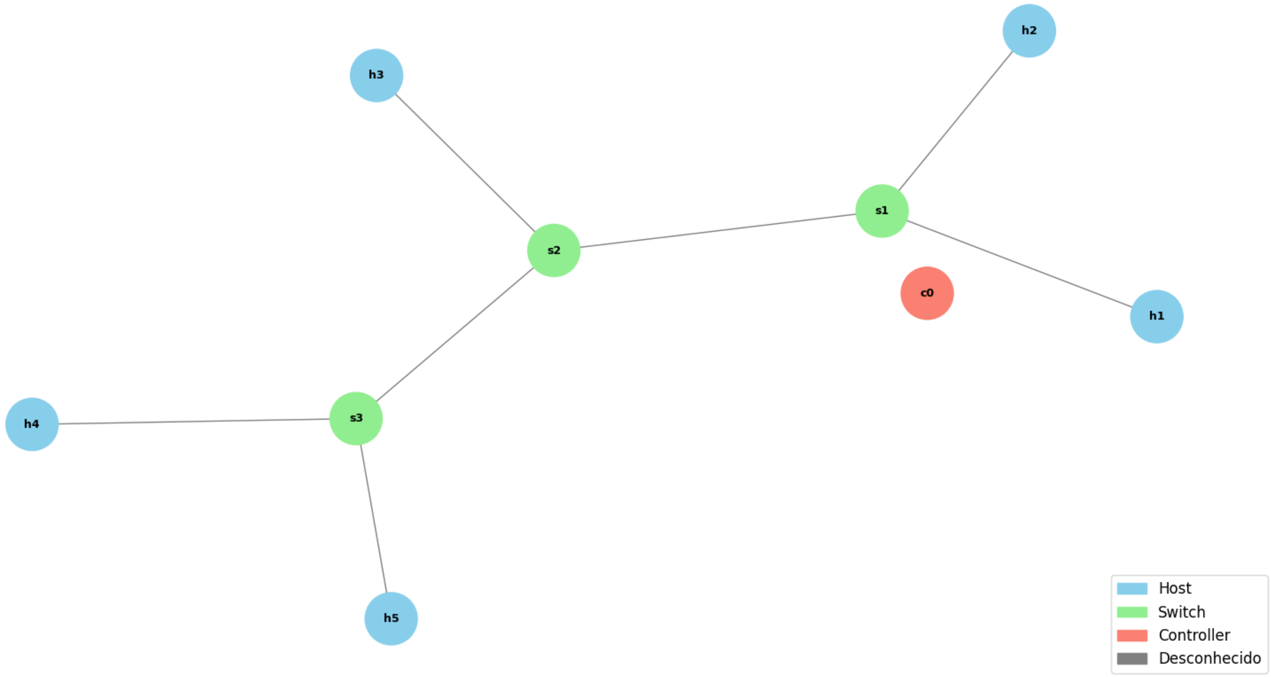
```
mininet> nodes
available nodes are:
c0 h1 h2 h3 h4 h5 s1 s2 s3
mininet>
```



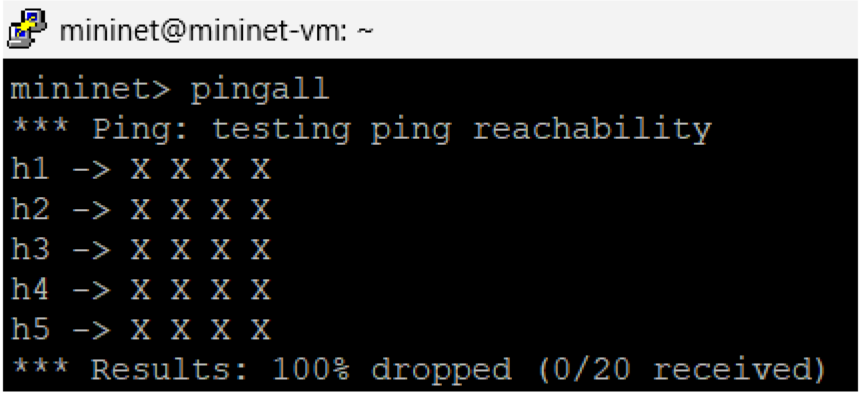
mininet@mininet-vm: ~

```
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s2-eth2
h4 h4-eth0:s3-eth2
h5 h5-eth0:s3-eth3
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:s2-eth1
s2 lo:  s2-eth1:s1-eth3 s2-eth2:h3-eth0 s2-eth3:s3-eth1
s3 lo:  s3-eth1:s2-eth3 s3-eth2:h4-eth0 s3-eth3:h5-eth0
c0
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=16764>
<Host h2: h2-eth0:10.0.0.2 pid=16766>
<Host h3: h3-eth0:10.0.0.3 pid=16768>
<Host h4: h4-eth0:10.0.0.4 pid=16770>
<Host h5: h5-eth0:10.0.0.5 pid=16772>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=16777>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=16780>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=16783>
<RemoteController c0: 127.0.0.1:6653 pid=16756>
mininet>
```

c) Desenho ilustrativo da topologia



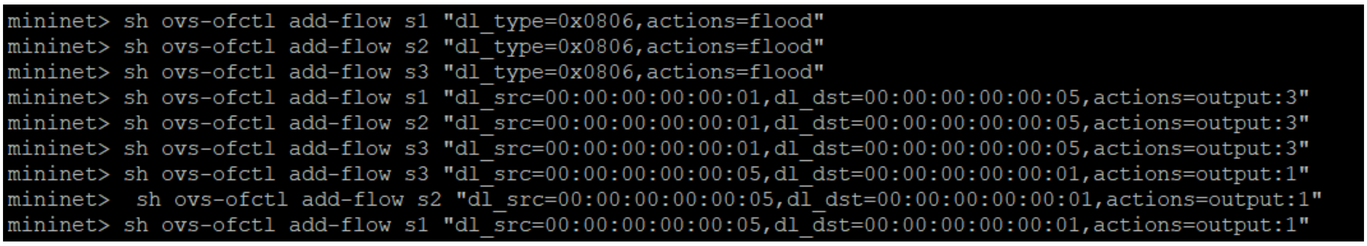
d) Testes de ping



mininet@mininet-vm: ~

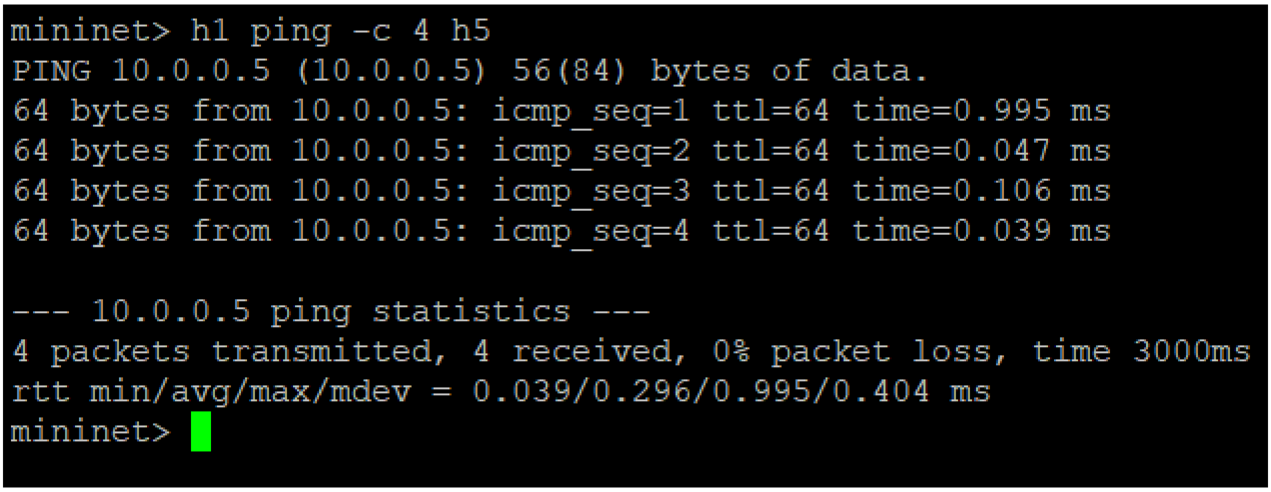
```
mininet> pingall
*** Ping: testing ping reachability
h1 -> X X X X
h2 -> X X X X
h3 -> X X X X
h4 -> X X X X
h5 -> X X X X
*** Results: 100% dropped (0/20 received)
```

e) Criação das novas regras para os nós



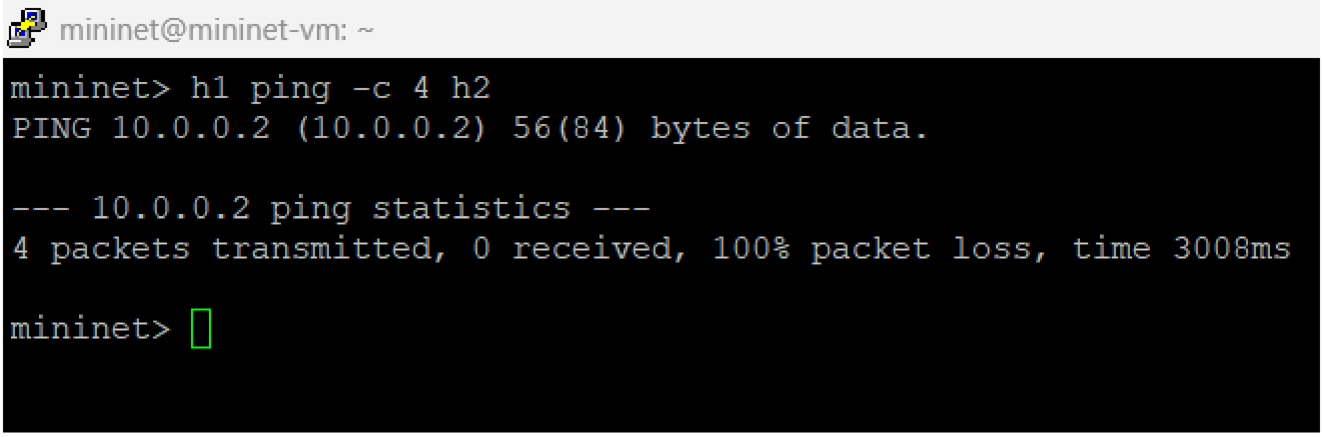
```
mininet> sh ovs-ofctl add-flow s1 "dl_type=0x0806,actions=flood"
mininet> sh ovs-ofctl add-flow s2 "dl_type=0x0806,actions=flood"
mininet> sh ovs-ofctl add-flow s3 "dl_type=0x0806,actions=flood"
mininet> sh ovs-ofctl add-flow s1 "dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:05,actions=output:3"
mininet> sh ovs-ofctl add-flow s2 "dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:05,actions=output:3"
mininet> sh ovs-ofctl add-flow s3 "dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:05,actions=output:3"
mininet> sh ovs-ofctl add-flow s3 "dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:01,actions=output:1"
mininet>  sh ovs-ofctl add-flow s2 "dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:01,actions=output:1"
mininet> sh ovs-ofctl add-flow s1 "dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:01,actions=output:1"
```

f) Testes de ping



```
mininet> h1 ping -c 4 h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
64 bytes from 10.0.0.5: icmp_seq=1 ttl=64 time=0.995 ms
64 bytes from 10.0.0.5: icmp_seq=2 ttl=64 time=0.047 ms
64 bytes from 10.0.0.5: icmp_seq=3 ttl=64 time=0.106 ms
64 bytes from 10.0.0.5: icmp_seq=4 ttl=64 time=0.039 ms

--- 10.0.0.5 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.039/0.296/0.995/0.404 ms
mininet> 
```



mininet@mininet-vm: ~

```
mininet> h1 ping -c 4 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3008ms

mininet> 
```