TRABALHO FINAL MININET
Parte_2

Nome : Guilherme Felipe Ribeiro e Pedro Consoli Bressan
Matricula: 2042 e 1764
Cruso : GEC

1 - Código Python da Topologia Customizada

https://github.com/Gpatinho/C115_Conceitos-e-Tecnologias-para-Dispositivos-Conectados/blob/main/Trabalho-mininet/topo_custom.ipynb

2 - Execução da Topologia

Passo 1: Limpar topologias antigas
sudo mn –c

```
pedro@CapDellG15:~$ sudo mn -c
[sudo] password for pedro:
*** Removing excess controllers/ofprotocols/ofdatapaths/pings/noxes
killall controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controller ovs-te
stcontroller udpbwtest mnexec ivs ryu-manager  2> /dev/null
killall -9 controller ofprotocol ofdatapath ping nox_core lt-nox_core ovs-openflowd ovs-controller ovs
-testcontroller udpbwtest mnexec ivs ryu-manager  2> /dev/null
pkill -9 -f "sudo mnexec"
*** Removing junk from /tmp
rm -f /tmp/vconn* /tmp/vlogs* /tmp/*.out /tmp/*.log
*** Removing old X11 tunnels
*** Removing excess kernel datapaths
ps ax | egrep -o 'dp[0-9]+' | sed 's/dp/nl:/'
***  Removing OVS datapaths
ovs-vsctl --timeout=1 list-br
ovs-vsctl --timeout=1 list-br
*** Removing all links of the pattern foo-ethX
ip link show | egrep -o '([-_.[:alnum:]]+-eth[[:digit:]]+)'
ip link show
*** Killing stale mininet node processes
pkill -9 -f mininet:
*** Shutting down stale tunnels
pkill -9 -f Tunnel=Ethernet
pkill -9 -f .ssh/mn
rm -f ~/.ssh/mn/*
*** Cleanup complete.
pedro@CapDellG15:~$
```

Passo 2: Executar o script
sudo python3 topo_custom.py

```
pedro@CapDellG15:~$ cd"C:\Users\pbres\OneDrive\Desktop\INATEL\C115\Codpedro@CapDellG15: $ cd"C:\Users\pbres\OneDrive\Desktop\INATEL\C115\Codigo_topo"~
cdC:\Users\pbres\OneDrive\Desktop\INATEL\C115\Codigo_topo~: command not found
pedro@CapDellG15:~$ cd~"C:\Users\pbres\OneDrive\Desktop\INATEL\C115\Codigo_topo"
cd~C:\Users\pbres\OneDrive\Desktop\INATEL\C115\Codigo_topo: command not found
pedro@CapDellG15:~$ cd /mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ ls
topo_custom.py
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ sudo apt update
[sudo] password for pedro:
Sorry, try again.
[sudo] password for pedro:
Hit:1 http://security.ubuntu.com/ubuntu focal-security InRelease
Hit:2 http://archive.ubuntu.com/ubuntu focal InRelease
Get:3 http://archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Hit:4 http://archive.ubuntu.com/ubuntu focal-backports InRelease
Fetched 128 kB in 1s (86.4 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
All packages are up to date.
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ sudo apt install -y dos2unix
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  dos2unix
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 374 kB of archives.
After this operation, 1342 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/universe amd64 dos2unix amd64 7.4.0-2 [374 kB]
Fetched 374 kB in 2s (242 kB/s)
Selecting previously unselected package dos2unix.
(Reading database ... 60662 files and directories currently installed.)
Preparing to unpack .../dos2unix_7.4.0-2_amd64.deb ...
Unpacking dos2unix (7.4.0-2) ...
Setting up dos2unix (7.4.0-2) ...
Processing triggers for man-db (2.9.1-1) ...
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ dos2unix topo_custom.py
dos2unix: converting file topo_custom.py to Unix format...
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ sudo apt install -y mininet
Reading package lists... Done
```

```
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ sudo apt install -y mininet
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  mininet
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 125 kB of archives.
After this operation, 886 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu focal/universe amd64 mininet amd64 2.2.2-5ubuntu1 [125 kB]
Fetched 125 kB in 1s (148 kB/s)
Selecting previously unselected package mininet.
(Reading database ... 60760 files and directories currently installed.)
Preparing to unpack .../mininet_2.2.2-5ubuntu1_amd64.deb ...
Unpacking mininet (2.2.2-5ubuntu1) ...
Setting up mininet (2.2.2-5ubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
pedro@CapDellG15:/mnt/c/Users/pbres/OneDrive/Desktop/INATEL/C115/Codigo_topo$ sudo python3 topo_custom.py
*** Creating network
*** 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

*** Starting 3 switches
s1 s2 s3 ...

### Rede inicializada ###
### Testando conectividade inicial com 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)
*** Starting CLI:
mininet> pingall
*** Ping: testing ping reachability
```

```
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)
mininet> h1 ping -c 3 h2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable

--- 10.0.0.2 ping statistics ---
3 packets transmitted, 0 received, +3 errors, 100% packet loss, time 2033ms
pipe 3
```

Obs. Estamos usando `controller=None` (controlador manual), conforme solicitado no item (a).
O parâmetro `autoSetMacs=True` gera **endereços MAC padronizados** automaticamente.

3 - Inspecionar informações da rede (b)

Dentro do prompt do Mininet (`mininet>`):
nodes

net

Dump

```
mininet> nodes
available nodes are:
h1 h2 h3 h4 h5 s1 s2 s3
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
h3 h3-eth0:s2-eth1
h4 h4-eth0:s3-eth1
h5 h5-eth0:s3-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0 s1-eth3:s2-eth2
s2 lo:  s2-eth1:h3-eth0 s2-eth2:s1-eth3 s2-eth3:s3-eth3
s3 lo:  s3-eth1:h4-eth0 s3-eth2:h5-eth0 s3-eth3:s2-eth3
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=71934>
<Host h2: h2-eth0:10.0.0.2 pid=71936>
<Host h3: h3-eth0:10.0.0.3 pid=71938>
<Host h4: h4-eth0:10.0.0.4 pid=71940>
<Host h5: h5-eth0:10.0.0.5 pid=71942>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=71947>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=71950>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=71953>
mininet>
```

Para um nó específico:

```
mininet> h1 ifconfig -a
h1-eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.0.1  netmask 255.0.0.0  broadcast 10.255.255.255
        inet6 fe80::200:ff:fe00:1  prefixlen 64  scopeid 0x20<link>
        ether 00:00:00:00:00:01  txqueuelen 1000  (Ethernet)
        RX packets 13  bytes 1026 (1.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 40  bytes 2200 (2.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 11  bytes 1232 (1.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 11  bytes 1232 (1.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

s1 ifconfig –a

```
mininet> s1 ifconfig -a
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1280
        inet 172.28.4.67  netmask 255.255.240.0  broadcast 172.28.15.255
        inet6 fe80::215:5dff:feea:2f8  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:ea:02:f8  txqueuelen 1000  (Ethernet)
        RX packets 144336  bytes 155077086 (155.0 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 19551  bytes 1359206 (1.3 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 3058279  bytes 338861567 (338.8 MB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 3058279  bytes 338861567 (338.8 MB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ovs-system: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether ba:8d:59:50:eb:fb  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 8e:02:e0:56:1c:45  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s2: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether 0e:88:ec:68:30:40  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s3: flags=4098<BROADCAST,MULTICAST>  mtu 1500
        ether a2:18:17:61:9a:47  txqueuelen 1000  (Ethernet)
```

```
s1-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::84b0:69ff:fe3e:213e  prefixlen 64  scopeid 0x20<link>
        ether 86:b0:69:3e:21:3e  txqueuelen 1000  (Ethernet)
        RX packets 40  bytes 2200 (2.2 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1026 (1.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::c002:dbff:fe69:3efd  prefixlen 64  scopeid 0x20<link>
        ether c2:02:db:69:3e:fd  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 2074 (2.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1026 (1.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s1-eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::e0da:ffff:fec0:952b  prefixlen 64  scopeid 0x20<link>
        ether e2:da:ff:c0:95:2b  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s2-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::c0c3:fdff:fed8:db4c  prefixlen 64  scopeid 0x20<link>
        ether c2:c3:fd:d8:db:4c  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 2074 (2.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1026 (1.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s2-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::400:8aff:febb:6edb  prefixlen 64  scopeid 0x20<link>
        ether 06:00:8a:bb:6e:db  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s2-eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::84c8:b7ff:fe56:7202  prefixlen 64  scopeid 0x20<link>
        ether 86:c8:b7:56:72:02  txqueuelen 1000  (Ethernet)
```

```
s3-eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::d87b:3cff:fe62:857  prefixlen 64  scopeid 0x20<link>
        ether da:7b:3c:62:08:57  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 2074 (2.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14  bytes 1116 (1.1 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s3-eth2: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::bc10:28ff:fef9:33a1  prefixlen 64  scopeid 0x20<link>
        ether be:10:28:f9:33:a1  txqueuelen 1000  (Ethernet)
        RX packets 37  bytes 2074 (2.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13  bytes 1026 (1.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

s3-eth3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::54b0:bbff:fee6:e9d7  prefixlen 64  scopeid 0x20<link>
        ether 56:b0:bb:e6:e9:d7  txqueuelen 1000  (Ethernet)
        RX packets 12  bytes 936 (936.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 12  bytes 936 (936.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```
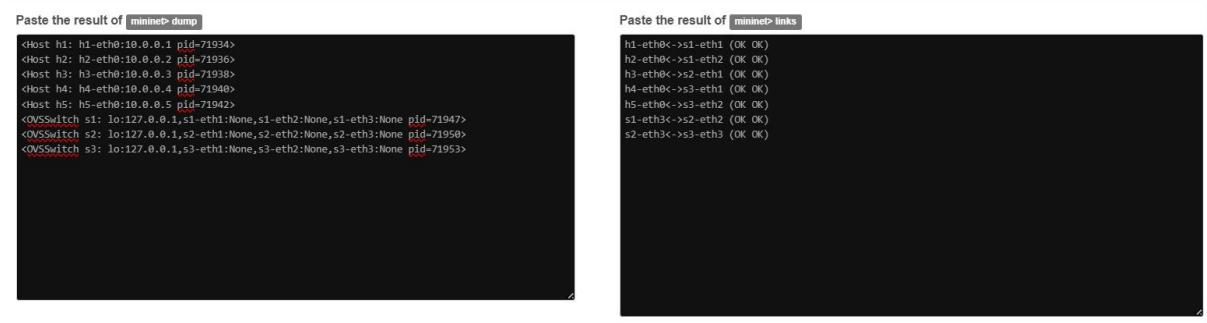
Obs. Esses comandos exibem interfaces, IPs, MACs e portas.
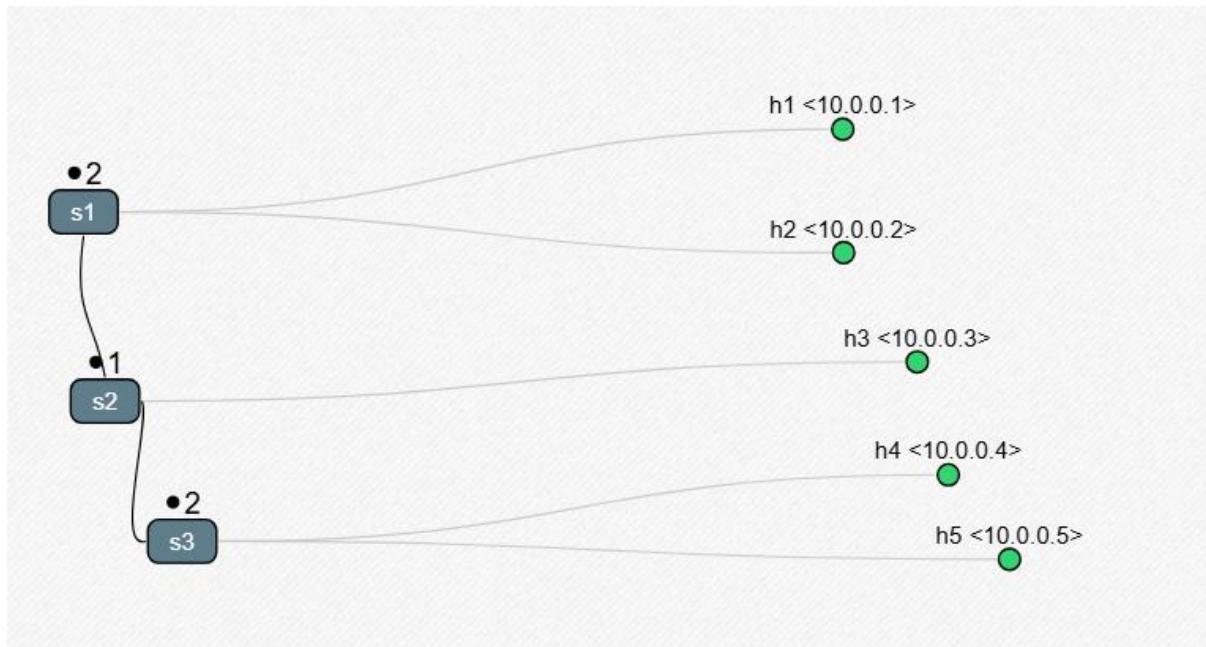
4 - Criar desenho ilustrativo (c)

Para desenhar a topologia graficamente:
sudo ~/mininet/examples/miniedit.py

Paste the result of `mininet> dump`

```
<Host h1: h1-eth0:10.0.0.1 pid=71934>
<Host h2: h2-eth0:10.0.0.2 pid=71936>
<Host h3: h3-eth0:10.0.0.3 pid=71938>
<Host h4: h4-eth0:10.0.0.4 pid=71940>
<Host h5: h5-eth0:10.0.0.5 pid=71942>
<OVSSwitch s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None,s1-eth3:None pid=71947>
<OVSSwitch s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=71950>
<OVSSwitch s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=71953>
```

Paste the result of `mininet> links`

```
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s1-eth2 (OK OK)
h3-eth0<->s2-eth1 (OK OK)
h4-eth0<->s3-eth1 (OK OK)
h5-eth0<->s3-eth2 (OK OK)
s1-eth3<->s2-eth2 (OK OK)
s2-eth3<->s3-eth3 (OK OK)
```

http://demo.spear.narmox.com/app/?apiurl=demo#!/mininet

Obs. Monte a topologia conforme o código e adicione manualmente os endereços IP e MAC obtidos via dump.
 Depois, tire um **print da tela** para o relatório.

5 - Testes de ping (d)

Ainda no CLI do Mininet:

Testar conectividade entre todos os nós:
pingall

```
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)
```

Testar hosts específicos:
h1 ping -c 4 h3

h4 ping -c 4 h5

```
mininet> h1 ping -c 4 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3051ms
pipe 4
mininet> h4 ping -c 4 h3
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
From 10.0.0.4 icmp_seq=1 Destination Host Unreachable
From 10.0.0.4 icmp_seq=2 Destination Host Unreachable
From 10.0.0.4 icmp_seq=3 Destination Host Unreachable
From 10.0.0.4 icmp_seq=4 Destination Host Unreachable

--- 10.0.0.3 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3067ms
pipe 4
```

6 - Apagar regras antigas e adicionar novas com base em MAC (e)

Primeiro, limpe os fluxos dos switches:

sh ovs-ofctl del-flows s1

sh ovs-ofctl del-flows s2

sh ovs-ofctl del-flows s3

```
mininet> sh ovs-ofctl del-flows s1
mininet> sh ovs-ofctl del-flows s2
mininet> sh osv-ofctl del-flows s3
/bin/sh: 1: osv-ofctl: not found
```

Obs. Agora adicione **regras de fluxo baseadas em endereços MAC** para permitir comunicação entre hosts de switches diferentes.

Exemplo: permitir h1 ↔ h4 e h2 ↔ h5

h1 → h4:

sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04,actions=output:3

sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01,actions=output:2

h2 → h5:

sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:05,actions=output: 3

sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:02,actions=output:2

```
mininet> sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04
,actions=output:3
mininet> sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01
,actions=output:2
mininet> sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:05
,actions=output: 3
ovs-ofctl: 'add-flow' command takes at most 2 arguments
mininet> sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:02
,actions=output:2
mininet>
```

Obs. Essas regras fazem com que **hosts de switches diferentes se comuniquem via MAC**.

7 - Teste de ping com novas regras (f)

Após adicionar as regras:
h1 ping -c 4 h4

h2 ping -c 4 h5

Se as regras estiverem corretas, os pacotes ICMP serão entregues com sucesso.

```
mininet> sh ovs-ofctl del-flows s1
mininet> sh ovs-ofctl del-flows s2
mininet> sh osv-ofctl del-flows s3
/bin/sh: 1: osv-ofctl: not found
mininet> sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04
,actions=output:3
mininet> sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01
,actions=output:2
mininet> sh ovs-ofctl add-flow s1 dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:05
,actions=output: 3
ovs-ofctl: 'add-flow' command takes at most 2 arguments
mininet> sh ovs-ofctl add-flow s3 dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:02
,actions=output:2
mininet> h1 ping -c 4 h4
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
From 10.0.0.1 icmp_seq=1 Destination Host Unreachable
From 10.0.0.1 icmp_seq=2 Destination Host Unreachable
From 10.0.0.1 icmp_seq=3 Destination Host Unreachable
From 10.0.0.1 icmp_seq=4 Destination Host Unreachable

--- 10.0.0.4 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3060ms
pipe 4
mininet> h2 ping -c 4 h5
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
From 10.0.0.2 icmp_seq=1 Destination Host Unreachable
From 10.0.0.2 icmp_seq=2 Destination Host Unreachable
From 10.0.0.2 icmp_seq=3 Destination Host Unreachable
From 10.0.0.2 icmp_seq=4 Destination Host Unreachable

--- 10.0.0.5 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 3074ms
pipe 4
mininet>
```

Resumo dos Comandos (para execução no terminal)

Etapa	Comando	Função
Limpar topologia	`sudo mn -c`	Remove redes anteriores
Executar script	`sudo python3 topo_custom.py`	Cria rede customizada
Ver nós	`nodes`	Lista hosts e switches
Ver conexões	`net`	Mostra ligações
Ver interfaces	`dump`	Mostra IP e MAC
Teste geral	`pingall`	Verifica conectividade
Limpar regras	`sh ovs-ofctl del-flows s1`	Apaga fluxos antigos
Criar regras MAC	`sh ovs-ofctl add-flow s1 ...`	Cria rotas manuais
Testar nova rota	`h1 ping -c 4 h4`	Valida regra criada
Encerrar rede	`exit + sudo mn -c`	Finaliza e limpa tudo