

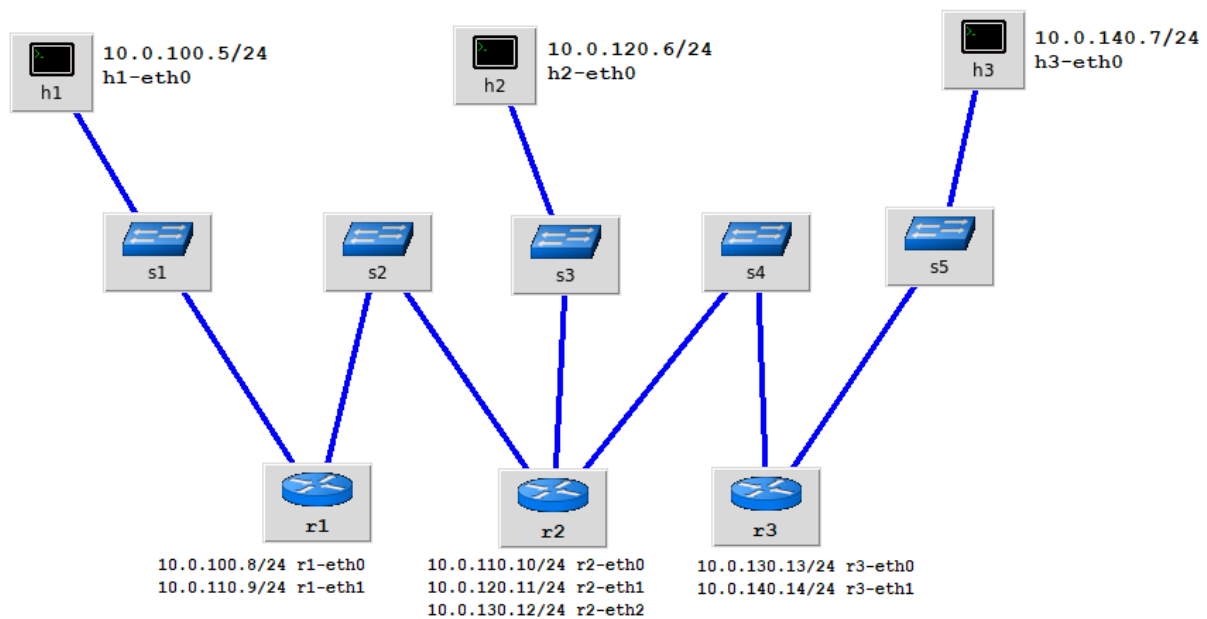
Planificación y administración de redes

ESCENARIO 4 MININET

1. Esquema gráfico de la configuración.
2. Comandos de configuración de cada nodo.
3. Verificación de conectividad (ping) entre nodos.
4. Captura de tráfico en el router r2 durante una comunicación entre h1 y h3.

Fernando Tirado Bulnes – 1º ASIR

1. Esquema gráfico de la configuración



2. Comandos de configuración de cada nodo

1- Empezamos abriendo mininet con Virtual Box poniendo usuario y contraseña (mininet,mininet) y viendo su ip con el comando: **"ip a"**

2- Nos descargamos en nuestra máquina Linux el escenario propuesto por el profesor "Escenario4.py".

3- En nuestra máquina Linux nos conectamos a mininet mediante el comando:

```
"ssh -X -v mininet@(ip de mininet)".
```

4- Para abrir nuestro escenario, ejecutamos

```
"sudo python (nombre escenario)".
```

5- Abrimos h1, h2, h3, r1, r2 y r3 (**"xterm h1"**, **"xterm h2"**, **"xterm h3"**, **"xterm r1"**, **"xterm r2"**, **"xterm r3"**)

Ahora, sólo habría que configurar cada uno:

Para h1:

```
ip a add 10.0.100.5/24 dev h1-eth0
```

Para h2:

```
ip a add 10.0.120.6/24 dev h2-eth0
```

Para h3:

```
ip a add 10.0.140.7/24 dev h3-eth0
```

Para r1:

```
ip a add 10.0.100.8/24 dev r1-eth0  
ip a add 10.0.110.9/24 dev r1-eth1
```

Para r2:

```
ip a add 10.0.110.10/24 dev r2-eth0  
ip a add 10.0.120.11/24 dev r2-eth1  
ip a add 10.0.130.12/24 dev r2-eth2
```

Para r3:

```
ip a add 10.0.130.13/24 dev r3-eth0  
ip a add 10.0.140.14/24 dev r3-eth1
```

6- Para que h1, h2, h3, r1, r2 y r3 puedan enviar y recibir datos mutuamente tendríamos que usar el siguiente comando:

- Escribir en h1:

```
ip r add default via 10.0.100.8
```

- Escribir en h2:

```
ip r add default via 10.0.120.11
```

- Escribir en h3:

```
ip r add default via 10.0.140.14
```

- Escribir en r1:

```
ip r add default via 10.0.110.10
```

- Escribir en r2:

```
ip r add default via 10.0.110.9  
ip r add 10.0.140.7 via 10.0.130.13  
ip r add 10.0.140.14 via 10.0.130.13
```

- Escribir en r3:

```
ip r add default via 10.0.130.12
```

De esta forma, todos estarían configurados de manera que son alcanzables entre ellos.

3. Verificación de conectividad (ping) entre nodos

Para comprobarlo habría que hacer un pingeo entre todos los nodos:

- Desde h1 hacia h2, h3, r1, r2 y r3, respectivamente:

```
ping 10.0.120.6  
ping 10.0.140.7  
ping 10.0.100.8  
ping 10.0.110.9  
ping 10.0.110.10  
ping 10.0.120.11  
ping 10.0.130.12  
ping 10.0.130.13  
ping 10.0.140.14
```

- Desde h2 hacia h1, h3, r1, r2 y r3, respectivamente:

```
ping 10.0.100.5  
ping 10.0.140.7  
ping 10.0.100.8  
ping 10.0.110.9  
ping 10.0.110.10  
ping 10.0.120.11  
ping 10.0.130.12  
ping 10.0.130.13  
ping 10.0.140.14
```

- Desde h3 hacia h1, h2, r1, r2 y r3, respectivamente:

```
ping 10.0.100.5
ping 10.0.120.6
ping 10.0.100.8
ping 10.0.110.9
ping 10.0.110.10
ping 10.0.120.11
ping 10.0.130.12
ping 10.0.130.13
ping 10.0.140.14
```

- Desde r1 hacia h1, h2, h3, r2 y r3, respectivamente:

```
ping 10.0.100.5
ping 10.0.120.6
ping 10.0.140.7
ping 10.0.110.10
ping 10.0.120.11
ping 10.0.130.12
ping 10.0.130.13
ping 10.0.140.14
```

- Desde r2 hacia h1, h2, h3, r1 y r3, respectivamente:

```
ping 10.0.100.5
ping 10.0.120.6
ping 10.0.140.7
ping 10.0.100.8
ping 10.0.110.9
ping 10.0.130.13
ping 10.0.140.14
```

- Desde r3 hacia h1, h2, h3, r1 y r2, respectivamente:

```
ping 10.0.100.5
ping 10.0.120.6
ping 10.0.140.7
ping 10.0.100.8
ping 10.0.110.9
ping 10.0.110.10
ping 10.0.120.11
ping 10.0.130.12
```

4. Captura de tráfico en el router r2 mostrando tráfico entre h1 y h3

```
"Node: h1" (on mininet-virtual-machine)
64 bytes from 10.0.140.7: icmp_seq=343 ttl=61 time=0.071 ms
64 bytes from 10.0.140.7: icmp_seq=344 ttl=61 time=0.082 ms
64 bytes from 10.0.140.7: icmp_seq=345 ttl=61 time=0.073 ms
64 bytes from 10.0.140.7: icmp_seq=346 ttl=61 time=0.195 ms
64 bytes from 10.0.140.7: icmp_seq=347 ttl=61 time=0.073 ms
64 bytes from 10.0.140.7: icmp_seq=348 ttl=61 time=0.083 ms
64 bytes from 10.0.140.7: icmp_seq=349 ttl=61 time=0.073 ms
64 bytes from 10.0.140.7: icmp_seq=350 ttl=61 time=0.070 ms
64 bytes from 10.0.140.7: icmp_seq=351 ttl=61 time=0.073 ms
64 bytes from 10.0.140.7: icmp_seq=352 ttl=61 time=0.070 ms
64 bytes from 10.0.140.7: icmp_seq=353 ttl=61 time=0.077 ms
64 bytes from 10.0.140.7: icmp_seq=354 ttl=61 time=0.069 ms
64 bytes from 10.0.140.7: icmp_seq=355 ttl=61 time=0.079 ms
64 bytes from 10.0.140.7: icmp_seq=356 ttl=61 time=0.072 ms
64 bytes from 10.0.140.7: icmp_seq=357 ttl=61 time=0.072 ms
64 bytes from 10.0.140.7: icmp_seq=358 ttl=61 time=0.077 ms
64 bytes from 10.0.140.7: icmp_seq=359 ttl=61 time=0.072 ms
64 bytes from 10.0.140.7: icmp_seq=360 ttl=61 time=0.113 ms
64 bytes from 10.0.140.7: icmp_seq=361 ttl=61 time=0.069 ms
64 bytes from 10.0.140.7: icmp_seq=362 ttl=61 time=0.070 ms
64 bytes from 10.0.140.7: icmp_seq=363 ttl=61 time=0.087 ms
64 bytes from 10.0.140.7: icmp_seq=364 ttl=61 time=0.118 ms
64 bytes from 10.0.140.7: icmp_seq=365 ttl=61 time=0.107 ms
^C

"Node: h3" (on mininet-virtual-machine)
root@mininet-virtual-machine:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
2: h3-eth0@if28: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 9e:c8:d8:bc:4d:d6 brd ff:ff:ff:ff:ff:ff
    inet 10.0.140.7/24 scope global h3-eth0
        valid_lft forever preferred_lft forever
root@mininet-virtual-machine:~#

"Node: r2" (on mininet-virtual-machine)
(1), length 84
10.0.100.5 > 10.0.140.7: ICMP echo request, id 3972, seq 362, length 64
10:14:43.241303 IP (tos 0x0, ttl 62, id 24838, offset 0, flags [none], proto ICMP)
P (1), length 84
10.0.140.7 > 10.0.100.5: ICMP echo reply, id 3972, seq 362, length 64
10:14:44.241303 IP (tos 0x0, ttl 63, id 19575, offset 0, flags [DF], proto ICMP)
P (1), length 84
10.0.100.5 > 10.0.140.7: ICMP echo request, id 3972, seq 363, length 64
10:14:44.241341 IP (tos 0x0, ttl 62, id 24982, offset 0, flags [none], proto ICMP)
P (1), length 84
10.0.140.7 > 10.0.100.5: ICMP echo reply, id 3972, seq 363, length 64
10:14:45.241349 IP (tos 0x0, ttl 63, id 19682, offset 0, flags [DF], proto ICMP)
P (1), length 84
10.0.100.5 > 10.0.140.7: ICMP echo request, id 3972, seq 364, length 64
10:14:45.241402 IP (tos 0x0, ttl 62, id 25041, offset 0, flags [none], proto ICMP)
P (1), length 84
10.0.140.7 > 10.0.100.5: ICMP echo reply, id 3972, seq 364, length 64
10:14:46.241326 IP (tos 0x0, ttl 63, id 19850, offset 0, flags [DF], proto ICMP)
P (1), length 84
10.0.100.5 > 10.0.140.7: ICMP echo request, id 3972, seq 365, length 64
10:14:46.241376 IP (tos 0x0, ttl 62, id 25079, offset 0, flags [none], proto ICMP)
P (1), length 84
10.0.140.7 > 10.0.100.5: ICMP echo reply, id 3972, seq 365, length 64
```