Universidad Nacional del Comahue
Facultad de Informática
2025

Laboratorio Switches

Redes II, Tecnicatura en administración de sistemas y Software Libre

1.

```
pc1:~#ping -c 3 pc2
PING pc2 (192.168.0.2) 56(84) bytes of data.
64 bytes from pc2 (192.168.0.2): icmp_seq=1 ttl=63 time=15.2 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=2 ttl=63 time=2.37 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=3 ttl=63 time=1.29 ms

--- pc2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 1.291/6.311/15.267/6.348 ms

pc1:~#ping -c 3 pc3
PING pc3 (192.168.0.3) 56(84) bytes of data.
64 bytes from pc3 (192.168.0.3): icmp_seq=1 ttl=63 time=15.6 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=2 ttl=63 time=0.397 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=3 ttl=63 time=0.364 ms

--- pc3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.364/5.458/15.614/7.181 ms

pc1:~#ping -c 3 pc4
PING pc4 (10.0.0.3) 56(84) bytes of data.
64 bytes from pc4 (10.0.0.3): icmp_seq=1 ttl=64 time=10.3 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=2 ttl=64 time=0.419 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=3 ttl=64 time=0.602 ms

--- pc4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.419/3.781/10.324/4.627 ms
```

```
pc2:~#ping -c 3 pc1
PING pc1 (10.0.0.2) 56(84) bytes of data.
64 bytes from pc1 (10.0.0.2): icmp_seq=1 ttl=63 time=0.301 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=2 ttl=63 time=1.02 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=3 ttl=63 time=0.386 ms

--- pc1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.301/0.570/1.025/0.324 ms

pc2:~#ping -c 3 pc3
```

```
PING pc3 (192.168.0.3) 56(84) bytes of data.
64 bytes from pc3 (192.168.0.3): icmp_seq=1 ttl=64 time=10.4 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=2 ttl=64 time=1.15 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=3 ttl=64 time=0.811 ms

--- pc3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 0.811/4.131/10.425/4.452 ms

pc2:~#ping -c 3 pc4
PING pc4 (10.0.0.3) 56(84) bytes of data.
64 bytes from pc4 (10.0.0.3): icmp_seq=1 ttl=63 time=6.35 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=2 ttl=63 time=0.463 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=3 ttl=63 time=1.36 ms

--- pc4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2010ms
rtt min/avg/max/mdev = 0.463/2.728/6.355/2.591 ms
```

```
pc3:~#ping -c 3 pc1
PING pc1 (10.0.0.2) 56(84) bytes of data.
64 bytes from pc1 (10.0.0.2): icmp_seq=1 ttl=63 time=12.5 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=2 ttl=63 time=0.295 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=3 ttl=63 time=1.12 ms

--- pc1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.295/4.665/12.572/5.601 ms

pc3:~#ping -c 3 pc2
PING pc2 (192.168.0.2) 56(84) bytes of data.
64 bytes from pc2 (192.168.0.2): icmp_seq=1 ttl=64 time=0.234 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=2 ttl=64 time=0.213 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=3 ttl=64 time=0.908 ms

--- pc2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2004ms
rtt min/avg/max/mdev = 0.213/0.451/0.908/0.323 ms

pc3:~#ping -c 3 pc4
PING pc4 (10.0.0.3) 56(84) bytes of data.
64 bytes from pc4 (10.0.0.3): icmp_seq=1 ttl=63 time=0.322 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=2 ttl=63 time=1.59 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=3 ttl=63 time=0.708 ms

--- pc4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2008ms
```

```
rtt min/avg/max/mdev = 0.322/0.876/1.599/0.535 ms
```

```
pc4:~#ping -c 3 pc1
PING pc1 (10.0.0.2) 56(84) bytes of data.
64 bytes from pc1 (10.0.0.2): icmp_seq=1 ttl=64 time=0.295 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=2 ttl=64 time=0.203 ms
64 bytes from pc1 (10.0.0.2): icmp_seq=3 ttl=64 time=0.286 ms

--- pc1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.203/0.261/0.295/0.043 ms

pc4:~#ping -c 3 pc2
PING pc2 (192.168.0.2) 56(84) bytes of data.
64 bytes from pc2 (192.168.0.2): icmp_seq=1 ttl=63 time=15.4 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=2 ttl=63 time=0.902 ms
64 bytes from pc2 (192.168.0.2): icmp_seq=3 ttl=63 time=0.535 ms

--- pc2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2014ms
rtt min/avg/max/mdev = 0.535/5.640/15.485/6.963 ms

pc4:~#ping -c 3 pc3
PING pc3 (192.168.0.3) 56(84) bytes of data.
64 bytes from pc3 (192.168.0.3): icmp_seq=1 ttl=63 time=6.27 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=2 ttl=63 time=1.56 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=3 ttl=63 time=1.28 ms

--- pc3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 1.282/3.039/6.275/2.291 ms
```

2. luego limpiamos la caché arp de pc1, y pc4

```
pc1:~#ip -s -s neigh flush all
pc1:~#
```

```
pc4:~#ping -c 3 pc3
pc4:~#
```

ahora hacemos ping de pc1 a pc4

```
pc1:~#ping -c 3 pc4
PING pc4 (10.0.0.3) 56(84) bytes of data.
64 bytes from pc4 (10.0.0.3): icmp_seq=1 ttl=64 time=4.05 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=2 ttl=64 time=0.251 ms
64 bytes from pc4 (10.0.0.3): icmp_seq=3 ttl=64 time=0.962 ms

--- pc4 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
```

```
rtt min/avg/max/mdev = 0.251/1.754/4.051/1.650 ms
```

r1 captura

```
r1:~# tcpdump -i eth0 -v -n
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
22:11:29.758348 arp who-has 10.0.0.3 tell 10.0.0.2
```

pc4 captura

```
pc4:~# tcpdump -i eth0 -v -n
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
22:11:29.758348 arp who-has 10.0.0.3 tell 10.0.0.2
22:11:29.758725 arp reply 10.0.0.3 is-at f2:1a:fd:1d:74:3a
22:11:29.758424 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 10.0.0.3: ICMP echo request, id 5635, seq 1, length 64
22:11:29.758436 IP (tos 0x0, ttl 64, id 27376, offset 0, flags [none], proto ICMP (1), length
84) 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 5635, seq 1, length 64
22:11:30.754422 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 10.0.0.3: ICMP echo request, id 5635, seq 2, length 64
22:11:30.754454 IP (tos 0x0, ttl 64, id 27377, offset 0, flags [none], proto ICMP (1), length
84) 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 5635, seq 2, length 64
22:11:31.755433 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 10.0.0.3: ICMP echo request, id 5635, seq 3, length 64
22:11:31.755474 IP (tos 0x0, ttl 64, id 27378, offset 0, flags [none], proto ICMP (1), length
84) 10.0.0.3 > 10.0.0.2: ICMP echo reply, id 5635, seq 3, length 64
22:11:34.748744 arp who-has 10.0.0.2 tell 10.0.0.3
22:11:34.749700 arp reply 10.0.0.2 is-at 82:70:16:b2:be:ed
```

r1 solo captura el mensaje arp request preguntando por 10.0.0.3 (pc4) pero no la respuesta de pc4 ni tampoco los mensajes de ping. En cambio pc4 registra este primer mensaje arp request y los demás ping que llegan a él.

3. hacemos un ping desde pc1 a pc3

```
pc1:~# ping -c 3 pc3
PING pc3 (192.168.0.3) 56(84) bytes of data.
64 bytes from pc3 (192.168.0.3): icmp_seq=1 ttl=63 time=22.7 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=2 ttl=63 time=1.65 ms
64 bytes from pc3 (192.168.0.3): icmp_seq=3 ttl=63 time=0.684 ms

--- pc3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1998ms
rtt min/avg/max/mdev = 0.684/8.360/22.740/10.176 ms
pc1:~#
```

el monitoreo de r1 nos da

```
r1:~# tcpdump -i eth0 -v -n
tcpdump: listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
22:30:29.573236 arp who-has 10.0.0.1 tell 10.0.0.2
```

```
22:30:29.574693 arp reply 10.0.0.1 is-at 46:d1:71:c3:27:14
22:30:29.573312 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 192.168.0.3: ICMP echo request, id 7682, seq 1, length 64
22:30:29.585471 IP (tos 0x0, ttl 63, id 47709, offset 0, flags [none], proto ICMP (1), length
84) 192.168.0.3 > 10.0.0.2: ICMP echo reply, id 7682, seq 1, length 64
22:30:30.562456 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 192.168.0.3: ICMP echo request, id 7682, seq 2, length 64
22:30:30.563108 IP (tos 0x0, ttl 63, id 47710, offset 0, flags [none], proto ICMP (1), length
84) 192.168.0.3 > 10.0.0.2: ICMP echo reply, id 7682, seq 2, length 64
22:30:31.561133 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto ICMP (1), length 84)
10.0.0.2 > 192.168.0.3: ICMP echo request, id 7682, seq 3, length 64
22:30:31.561419 IP (tos 0x0, ttl 63, id 47711, offset 0, flags [none], proto ICMP (1), length
84) 192.168.0.3 > 10.0.0.2: ICMP echo reply, id 7682, seq 3, length 64
22:30:34.587221 arp who-has 10.0.0.2 tell 10.0.0.1
22:30:34.587658 arp reply 10.0.0.2 is-at ea:d2:7a:ba:ba:58

10 packets captured
10 packets received by filter
0 packets dropped by kernel
r1:~#
```

como pc1 y pc3 no están en la misma red, entonces tienen que hacer uso del router para poder enviar esos mensajes. Entonces pc1 tiene que averiguar la mac de su gateway (r1), larga el mensaje arp, r1 contesta, entonces comienza a enviar los mensajes de ping (con destino pc3). Como estamos monitorizando eth0 no podemos ver el trafico que sale por eth1 (192.189.0.0/24) y no podemos ver lo que pasa del lado de la red entre pc3 y r1.

4. hacemos traceroute de pc1 a pc4

```
pc1:~# traceroute pc4
traceroute to pc4 (10.0.0.3), 64 hops max, 40 byte packets
 1  pc4 (10.0.0.3)  10 ms  0 ms  0 ms
pc1:~#
```

hacemos un traceroute de pc1 a pc3

```
pc1:~# traceroute pc3
traceroute to pc3 (192.168.0.3), 64 hops max, 40 byte packets
 1  r1 (10.0.0.1)  11 ms  0 ms  0 ms
 2  pc3 (192.168.0.3)  6 ms  0 ms  0 ms
pc1:~#
```

como en la segunda las redes están separadas tienen que ir necesariamente el router y de ahí a pc3. en la primera no porque están en la misma red.

5.

```
#pc1.startup
ifconfig eth0 10.0.0.2/24
route add default gw 10.0.0.1
```

```
#pc2.startup
ifconfig eth0 192.168.0.2/24
route add default gw 192.168.0.1
```

```
#pc3.startup
ifconfig eth0 192.168.0.3/24
route add default gw 192.168.0.1
```

```
#pc4.startup
ifconfig eth0 10.0.0.3/24
route add default gw 10.0.0.1
```

```
#r1.startup
ifconfig eth0 10.0.0.1/24
ifconfig eth1 192.168.0.1/24
```

```
#s1.startup
ifconfig eth0 up
ifconfig eth0 hw ether 00:00:00:00:01:00

ifconfig eth1 up
ifconfig eth1 hw ether 00:00:00:00:01:01

ifconfig eth2 up
ifconfig eth2 hw ether 00:00:00:00:01:02

ifconfig eth3 up
ifconfig eth3 hw ether 00:00:00:00:01:03

ifconfig eth4 up
ifconfig eth4 hw ether 00:00:00:00:01:04

ifconfig eth5 up
ifconfig eth5 hw ether 00:00:00:00:01:05

brctl addbr br0
brctl addif br0 eth0
brctl addif br0 eth1
brctl addif br0 eth4
ifconfig br0 up

brctl addbr br1
brctl addif br1 eth5
brctl addif br1 eth2
brctl addif br1 eth3
ifconfig br1 up
```

Los host se configuran de forma normal (dirección ip, y gateway), y el router tiene la configuración para las redes de 10.0.0.0/24 y 192.168.0.0/24 . El switch tiene dos puentes uno

para la red 10.0.0.0/24 con eth0, eth1, y eth4, las computadoras de esa red tienen que estar conectadas pc1 -> eth1, pc4 -> eth4, r1 (eth0) -> eth0, y otra para la red 192.168.0.0/24 con eth2, eth3, eth5, las computadoras de esa red tienen que estar conectadas a esas interfaces, pc2-> eth2, pc3 -> eth3 y r1(eth1) -> eth5 (por medio de un hub).
Investigando el comando brctl addbr br1 crea dos switches separados con las interfaces indicadas . Esto genera dos switches separados, por lo que es importante que los hosts de la misma red estén conectados a las interfaces configuradas. en cuyo caso no podremos conectar a host en la misma red. sirve para que los host de redes separadas no reciban los mensajes de broadcast de una de las redes (por ejemplo cuando hacemos arp request por broadcast).

6. La separación es lógica pero da la apariencia de ser física. ya que solo tenemos un switch conectado a 4 máquinas, un hub y un router, pero da la apariencia de que son dos switches separados (por que tienen una configuración que hace que funcione así). Esto evita que mensajes de broadcast de una red pasen a la otra.
   Entonces tenemos una separación lógica pero que tiene el comportamiento de una separación física.