# Configuración avanzada de las redes

Este tema de examen vuelve a abordar, profundizando, diferentes elementos tratados en el estudio del tema anterior: gestión de las tarjetas de interfaz de red, tablas de enrutamiento y comandos básicos de test. Volveremos a estudiar algunos comandos, en un marco más técnico: integración en distintas redes, enrutamiento entre tarjetas de interfaz y conexión a redes seguras.

La segunda parte del tema concierne los comandos y herramientas de supervisión de la actividad de las redes, lo que permitirá hacer un diagnóstico de diferentes problemas de redes.

## 1. Conexión a una red inalámbrica

Por su propia naturaleza, las redes inalámbricas presentan riesgos importantes para la seguridad. Una persona malintencionada puede obtener fácilmente la lista de las redes activas alrededor de su sistema e intentar conectarse usando una interfaz inalámbrica.

Para evitar estas intrusiones, la mayoría de las redes inalámbricas están protegidas con mecanismos de autenticación y de cifrado según distintos protocolos. Para poder conectarse a la red, hay que enviar una clave cifrada al punto de acceso. Además, todos los mensajes que circulan en la red están cifrados.

Se pueden implementar distintos protocolos de protección, configurables a través de distintas herramientas en Linux.

Por defecto, en la mayoría de las distribuciones Linux, las interfaces de red inalámbricas son configuradas automáticamente usando el servicio `NetworkManager`. Si, en el marco de un test, desea gestionarlas dinámicamente desde la consola, deberá desactivar temporalmente este servicio, o configurarlo para que no administre las tarjetas que usted está usando.

### a. Redes inalámbricas no seguras

Algunas redes son de libre acceso y no cifran los intercambios. Se aconseja encarecidamente no conectarse a ellas, en especial en los lugares públicos.

Para conectarse en alguna de esas redes, puede usar los comandos `iw` o `iwconfig`, vistos anteriormente.

La conexión completa se hace en distintas etapas:

- Comprobar que la red está activa

  Para ello, se pueden escanear las redes accesibles a través de la tarjeta de interfaz de red: `iwlist` o `iw dev scan`.

- Conectar la tarjeta a punto de acceso de red

  Usando los comandos `iw` o `iwconfig`, especificando el ESSID (*Extended Service Set Identification*) de la red de destino.

- Comprobar la conexión

  Visualizando el estado de la tarjeta de interfaz de red, usando los comandos `iw dev` o `iwconfig`.

- Configurar la dirección IP de la tarjeta de interfaz de red

  La mayoría de las redes Wi-Fi disponen de un servidor DHCP, que puede distribuir dinámicamente una dirección IP a la tarjeta de interfaz de red, en respuesta al comando `dhclient`. Si no es el caso, se puede configurar dinámicamente la tarjeta usando los comandos `ip address` o `ifconfig`.

- Comprobar la conectividad IP

  Visualizando el estado IP de la tarjeta (dirección, máscara de subred y pasarela por defecto), con los comandos `ip address` e `ip route`, o `ifconfig` y `route`, y comprobando después la conexión hacia la pasarela por defecto con el comando `ping`.

Desconexión de la red:

El comando

`iw` usado con el subcomando `disconnect` permite desconectarse de la red actual.

*Ejemplo con el comando `iw`*

*Conexión en una red Wi-Fi no segura desde un sistema CentOS 8.*

*Se desactiva temporalmente el servicio `NetworkManager`, para evitar interferencias con los tests:*

```
systemctl stop NetworkManager
```

*Se muestra el estado de la tarjeta de interfaz de red inalámbrica:*

```
iw dev
phy#0
	Interface wlo1
		ifindex 3
		wdev 0x1
		addr 12:c4:77:51:b1:3c
		type managed
		txpower 15.00 dBm
```

*La tarjeta no tiene ningún ESSID asociado.*

*Se buscan redes inalámbricas accesibles, incluyendo `FreeWifi`:*

```
iw dev wlo1 scan
BSS e4:9e:12:11:18:9c(on wlp3s0)
	TSF: 0 usec (0d, 00:00:00)
	freq: 2427
	beacon interval: 96 TUs
	capability: ESS ShortSlotTime (0x0401)
	signal: -66.00 dBm
	last seen: 0 ms ago
	Information elements from Probe Response frame:
	SSID: FreeWifi
	Supported rates: 1.0* 2.0* 5.5* 11.0* 22.0 6.0 9.0 12.0
```

[...]

*La red de SSID `FreeWifi` se encuentra accesible, no es segura (no se menciona ningún protocolo de autenticación).*

*Se conecta la tarjeta de interfaz de red:*

```
iw wlo1 connect FreeWifi
```

*Se comprueba el estado de la tarjeta de interfaz de red:*

```
iw dev
phy#0
    Interfaz wlo1
        ifindex 3
        wdev 0x1
        addr 12:c4:77:51:b1:3c
        ssid FreeWifi
        type managed
        channel 11 (2462 MHz), width: 20 MHz, center1: 2462 MHz
        txpower 15.00 dBm
```

*La tarjeta está conectada en la capa enlace a la red `FreeWifi`, canal `11`.*

*Se muestra la configuración IP de las diferentes tarjetas de interfaz de red:*

```
ip -br a
lo            UNKNOWN      127.0.0.1/8::1/128
enp38s0       UP           192.168.0.4/24 fe80::5508:86a9:e923:75b3/64
wlo1          UP           fe80::10c4:77ff:fe51:b13c/64
```

*Podemos ver que la tarjeta de interfaz de red inalámbrica, `wlo1`, no tiene dirección IPv4, su dirección IPv6 es la del enlace local.*

*Se solicita una dirección IP dinámica al servidor DHCP de la red `FreeWifi`:*

```
dhclient wlo1
ip -br a
lo            UNKNOWN      127.0.0.1/8::1/128
enp38s0       UP           192.168.0.4/24 fe80::5508:86a9:e923:75b3/64
wlo1          UP           10.48.63.50/13 fe80::10c4:77ff:fe51:b13c/64
```

*La tarjeta tiene ahora una dirección IPv4.*

*Se muestran las pasarelas por defecto:*

```
ip route list default
default via 10.55.255.254 dev wlo1
default via 192.168.0.254 dev enp38s0 proto dhcp metric 100
```

*La pasarela por defecto de la red inalámbrica es* `10.55.255.254` .

*Comprobamos la comunicación con la pasarela por defecto:*

```
ping -c1 10.55.255.254
PING 10.55.255.254 (10.55.255.254) 56(84) bytes of data.
64 bytes from 10.55.255.254: icmp_seq=1 ttl=64 time=158 ms
--- 10.55.255.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 157.816/157.816/157.816/0.000 ms
```

*Nos desconectamos de la red inalámbrica:*

```
iw wlo1 disconnect
iw dev
phy#0
    Interfaz wlo1
        ifindex 3
        wdev 0x1
        addr 12:c4:77:51:b1:3c
        type managed
        txpower 15.00 dBm
```

*Ejemplo con el comando `iwconfig`*

*Conexión en la red Wi-Fi no segura usando un sistema Debian 10.*

*Desactivamos temporalmente el servicio `NetworkManager`, para evitar interferencias en nuestras comprobaciones:*

```
systemctl stop NetworkManager
```

*Se muestra el estado de la tarjeta de interfaz de red inalámbrica:*

```
iwconfig
wlp3s0    IEEE 802.11  ESSID:off/any
          Mode:Managed  Access Point: Not-Associated   Tx-Power=200 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Encryption key:off
          Power Management:off
enp0s10   no wireless extensions.
lo        no wireless extensions.
```

*La tarjeta no está asociada con ningún ESSID.*

*Buscamos las redes inalámbricas accesibles, incluyendo `FreeWifi`:*

```
iwlist scanning
[...]
          Cell 08 - Address: 00:07:CB:02:18:45
                    Channel:11
                    Frequency:2.462 GHz (Channel 11)
                    Quality=70/70  Signal level=-40 dBm
                    Encryption key:off
                    ESSID:"FreeWifi"
                    Bit Rates:1 Mb/s; 2 Mb/s; 5.5 Mb/s; 11 Mb/s; 22 Mb/s
                              6 Mb/s; 9 Mb/s; 12 Mb/s
                    Bit Rates:18 Mb/s; 24 Mb/s; 36 Mb/s; 48 Mb/s; 54 Mb/s
                    Mode:Master
[...]
```

*La red es accesible y no es segura (`Encryption key:off`).*

*Conectamos la tarjeta de interfaz de red:*

```
iwconfig wlp3s0 essid "FreeWifi"
```

*Se comprueba el estado de la tarjeta de interfaz de red:*

```
iwconfig
wlp3s0   IEEE 802.11  ESSID:"FreeWifi"
      Mode:Managed  Frequency:2.462 GHz  Access Point: 00:07:CB:02:18:45
      Bit Rate=54 Mb/s   Tx-Power=200 dBm
      Retry short limit:7   RTS thr:off   Fragment thr:off
      Encryption key:off
      Power Management:off
      Link Quality=70/70  Signal level=-40 dBm
      Rx invalid nwid:0  Rx invalid crypt:0  Rx invalid frag:0
      Tx excessive retries:0  Invalid misc:0   Missed beacon:0
```

*La tarjeta está conectada en la capa enlace de la red `FreeWifi`.*

*Mostramos la configuración IP de la tarjeta de interfaz de red:*

```
ifconfig wlp3s0
wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet6 fe80::21a:73ff:fe7e:147b  prefixlen 64  scopeid 0x20<link>
    ether 00:1a:73:7e:14:7b  txqueuelen 1000  (Ethernet)
    RX packets 0  bytes 0 (0.0 B)
    RX errors 0  dropped 0  overruns 0  frame 22777
    TX packets 0  bytes 0 (0.0 B)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
    device interrupt 19
```

*La tarjeta de interfaz de red inalámbrica no tiene dirección IPv4, su dirección IPv6 es la del enlace local.*

*Se solicita una dirección IP dinámica al servidor DHCP de la red `FreeWifi`:*

```
dhclient wlp3s0
ifconfig wlp3s0
wlp3s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.59.43.115  netmask 255.248.0.0  broadcast 10.63.255.255
    inet6 fe80::21a:73ff:fe7e:147b  prefixlen 64  scopeid 0x20<link>
    ether 00:1a:73:7e:14:7b  txqueuelen 1000  (Ethernet)
    RX packets 2  bytes 684 (684.0 B)
    RX errors 0  dropped 0  overruns 0  frame 32404
    TX packets 10  bytes 2306 (2.2 KiB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
    device interrupt 19
```

*La tarjeta presenta ahora una dirección IPv4,* `10.59.43.115` .

*Mostramos la tabla de enrutamiento:*

```
route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.63.255.254   0.0.0.0         UG    0      0        0 wlp3s0
0.0.0.0         192.168.0.254   0.0.0.0         UG    100    0        0 enp0s10
10.56.0.0       0.0.0.0         255.248.0.0     U     0      0        0 wlp3s0
169.254.0.0     0.0.0.0         255.255.0.0     U     1000   0        0 enp0s10
192.168.0.0     0.0.0.0         255.255.255.0   U     100    0        0 enp0s10
```

*La pasarela por defecto de la red inalámbrica es* `10.63.255.254` .

*Comprobamos la comunicación con la pasarela por defecto:*

```
ping -c1 10.63.255.254
PING 10.63.255.254 (10.63.255.254) 56(84) bytes of data.
64 bytes from 10.63.255.254: icmp_seq=1 ttl=64 time=160 ms
--- 10.63.255.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 160.470/160.470/160.470/0.000 ms
```

*Nos desconectamos de la red inalámbrica:*

```
iw wlp3s0 disconnect
iwconfig wlp3s0
wlp3s0   IEEE 802.11  ESSID:off/any
      Mode:Managed  Access Point: Not-Associated   Tx-Power=200 dBm
      Retry short limit:7   RTS thr:off   Fragment thr:off
      Encryption key:off
      Power Management:off
```

## b. Redes inalámbricas seguras

Las redes inalámbricas seguras utilizan diferentes protocolos de autenticación y de cifrado de las comunicaciones:

### WEP (Wired Equivalent Privacy)

Aunque forme parte de la norma IEEE 802.11, este protocolo presenta distintos fallos que comprometen gravemente su seguridad. Su uso no está aconsejado.

### WPA (Wi-Fi Protected Access)

Ha sido concebido para reemplazar al protocolo WEP, este protocolo existe en dos versiones, WPA et WPA2.

WPA usa el protocolo de cifrado TKIP. Aunque es más seguro que WEP, puede presentar algunos puntos débiles.

WPA2 es la versión «definitiva» del protocolo. Esta versión usa el protocolo de cifrado CCMP, más seguro que TKIP.

Las dos versiones del protocolo pueden funcionar a través de un servidor de autenticación y el protocolo de autenticación EAP (*Extensible Authentication Protocol*) o, de manera menos segura, con una clave compartida (modo PSK, *Pre Shared Key*).

La versión WAP2 es la más segura y la más usada hoy en día.

## c. Configuración de una conexión segura WPA/WPA2

Los comandos `wpa_passphrase` y `wpa_supplicant` permiten configurar y activar una conexión a una red segura WPA o WPA2.

Estos comandos forman parte del paquete de software `wpa_supplicant`.

Los comandos se usan en dos fases diferentes: primero generar la clave de autenticación con una `wpa_passphrase`, y después usar `wpa_supplicant` para conectar una interfaz de red a la red inalámbrica WPA o WPA2.

### Generación de la clave de autenticación

*Sintaxis*

```
wpa_passphrase Essid [ FraseContraseña ] [ > ArchivoClave ]
```

*Parámetros principales*

| | |
|---|---|
| `Essid` | Identificador de la red segura. |
| `[FraseContraseæa]` | Frase de contraseña sin cifrar. |
| `[ > ArchivoClave ]` | Redirección hacia el archivo de texto ArchivoClave. |

*Descripción*

Por defecto el comando muestra en la salida estándar la configuración necesaria para conectarse a la red segura cuyo ESSID (*Extended Service Set Identification*) se ha especificado, a partir de la frase de contraseña que se ha dado como argumento. Si ésta

no ha sido facilitada como argumento, se tiene que teclear.

La configuración generada contiene, como comentario, la frase de contraseña sin cifrar. Una vez que la conexión se haya realizado, por prudencia, es preferible suprimir ese comentario.

*Ejemplo*

*Definición de la configuración de conexión hacia una red WPA, ESSID* `SecWIFI`*, con la frase de contraseña "*`Un4 fr42e c0ntr4_23nn4`*":*

```
wpa_passphrase SecWIFI "Un4 fr42e c0ntr4_23nn4" > ./wpa_SecWIFI.conf
cat ./wpa_SecWIFI.conf
network={
    ssid="SecWIFI"
    #psk= "Un4 fr42e c0ntr4_23nn4"
    psk= adfdfed11bc345acb8c70f86202df4484fabfd0e0456261a06cffb2d589d6a4f
}
```

### Uso de la clave de autenticación

*Sintaxis*

```
wpa_supplicant [ -B ] -i Interfaz -c ArchConf
```

*Parámetros principales*

| | |
|---|---|
| `[ -B ]` | Proceso en segundo plano. |
| `-i Interfaz` | Interfaz que se querrá conectar. |
| `-c ArchConf` | Archivo de configuración para la interfaz. |

*Descripción*

El comando se ejecuta lanzando un proceso cargado para administrar la fase de autenticación para la conexión de la interfaz especificada a la red cuyo ESSID está configurado en el archivo indicado. El comando gestiona la autentificación WPA y WPA2, en modo de servidor de autenticación o en modo PSK.

*Ejemplo*

*Conexión de la tarjeta de interfaz de red `wlp3s0` en la red WPA con ESSID `SecWIFI`, con la clave configurada en el ejemplo anterior:*

```
iw dev
phy#0
    Interface wlp3s0
        ifindex 3
        wdev 0x1
        addr 00:1a:73:7e:14:7b
        type managed
        txpower 200.00 dBm
```

*La tarjeta de interfaz de red no está conectada a ninguna red inalámbrica.*

*Se solicita la conexión, con el archivo de configuración creado en el ejemplo anterior:*

```
cat ./wpa_SecWIFI.conf
network={
    ssid="SecWIFI"
```

```
    #psk= "Un4 fr42e c0ntr4_23nn4"
    psk= adfdfed11bc345acb8c70f86202df4484fabfd0e0456261a06cffb2d589d6a4f
}
 wpa_supplicant -B -i wlp3s0 -c ./wpa_SecWIFI.conf
Successfully initialized wpa_supplicant
ps -ef | grep wpa
root      486     1  0 abril01 ?      00:00:41 /sbin/wpa_supplicant
-u -s -O /run/wpa_supplicant
root     3374     1  0 09:43 ?        00:00:00 wpa_supplicant
-B -i wlp3s0 -c ./wpa_SecWIFI.conf
root     3378  3177  0 09:43 pts/0    00:00:00 grep wpa
```

*El comando ha ejecutado un proceso `/sbin/wpa_supplicant` en segundo plano.*

```
iw dev
phy#0
    Interface wlp3s0
        ifindex 3
        wdev 0x1
        addr 00:1a:73:7e:14:7b
        ssid SecWIFI
        type managed
        txpower 200.00 dBm
```

*La tarjeta de interfaz de red está conectada a la red `SecWIFI`.*

*Mostramos la configuración IP:*

```
ip a show dev wlp3s0
3: wlp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
  link/ether 00:1a:73:7e:14:7b brd ff:ff:ff:ff:ff:ff
```

*La tarjeta no tiene configurada una dirección IPv4.*

*Usamos el servidor DHCP de la red inalámbrica:*

```
dhclient wlp3s0
ip a show dev wlp3s0
3: wlp3s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast
state UP group default qlen 1000
  link/ether 00:1a:73:7e:14:7b brd ff:ff:ff:ff:ff:ff
  inet 192.168.0.11/24 brd 192.168.0.255 scope global dynamic wlp3s0
    valid_lft 43198sec preferred_lft 43198sec
```

*La tarjeta ahora está configurada en IP, con la dirección de red* `192.168.0.11` .

*Se muestra la pasarela por defecto:*

```
ip route
default via 192.168.0.254 dev wlp3s0
[...]
```

*La pasarela por defecto es* `192.168.0.254` .

*Se comprueba la comunicación con la pasarela por defecto:*

```
ping -c1 192.168.0.254 -I wlp3s0
PING 192.168.0.254 (192.168.0.254) from 192.168.0.11 wlp3s0: 56(84) bytes of data.
64 bytes from 192.168.0.254: icmp_seq=1 ttl=64 time=1.32 ms
--- 192.168.0.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 1.319/1.319/1.319/0.000 ms
```

*Desde una máquina remota:*

```
ssh pba@192.168.0.11
The authenticity of host '192.168.0.11 (192.168.0.11)' can't be established.
ECDSA key fingerprint is SHA256:Cbob+4M9v2AjgHYHBK/bSV3E7ZsJ+TCep/4lBkdHzGM.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.11' (ECDSA) to the list of known hosts.
pba@192.168.0.11's password:
```

```
Linux alpha 4.19.0-8-amd64 #1 SMP Debian 4.19.98-1 (2020-01-26) x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Thu Apr  2 08:58:41 2020 from 192.168.0.24
```

*Se ha establecido la comunicación.*

# 2. Conexión en distintas redes

Un sistema Linux se puede conectar a distintas redes, del mismo tipo o de tipos diferentes, como un simple host o con la función de un router, a través de distintas tarjetas de interfaz de red.

Una tarjeta de interfaz de red también puede ser configurada para estar integrada en distintas redes/subredes, y opcionalmente, realizar el enrutamiento entre ellas.

## a. Configuración multiredes (multihomed)

Un sistema Linux puede encontrarse en distintas redes a través de distintas tarjetas de interfaz de red. En ese caso, basta con configurar cada tarjeta de interfaz de red con los parámetros adaptados a la red IP en la que se encuentra dicha tarjeta: dirección IP y máscara de subred. Se configurará una pasarela por defecto y, opcionalmente, rutas específicas hacia algunas redes/subredes. Esto permite que el sistema pueda comunicarse con diferentes redes.

*Ejemplo*

*Configuración de un sistema con dos redes IP, una en Ethernet y la otra en Wi-Fi.*

*Las dos redes son privadas, una en `10.1.0.0/16`, la otra en `192.168.0.0/24`. Las máquinas de una red no se comunicarán con las de la otra red.*

*El sistema Linux tendrá dos direcciones:* `10.1.0.1` *en la red Ethernet,* `192.168.0.5` *en la red Wi-Fi.*

*La tarjeta de interfaz de red Wi-Fi ya está configurada.*

*Se muestran las interfaces de red disponibles:*

```
ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN
                              mode DEFAULT group default qlen 1000
  link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp38s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
                              mode DEFAULT group default qlen 1000
  link/ether e4:11:5b:50:13:32 brd ff:ff:ff:ff:ff:ff
3: wlo1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
                              mode DORMANT group default qlen 1000
  link/ether 9c:b7:0d:bb:2b:67 brd ff:ff:ff:ff:ff:ff
```

*El sistema dispone de dos interfaces de red físicas,* `enp38s0` *(Ethernet) y* `wlo1` *(Wi-Fi).*

*Se muestran las direcciones IP configuradas:*

```
ip -br a
lo            UNKNOWN       127.0.0.1/8::1/128
enp38s0       UP
wlo1          UP            192.168.0.5/24
2a01:e35:2439:1510:4635:482d:bc79:3f71/64 fe80::a8a:b708:d2b2:6bc5/64
```

*La tarjeta de interfaz de red Wi-Fi ya está configurada en IP.*

*Hay que configurar la tarjeta de interfaz de red Ethernet:*

```
ip a add 10.1.0.1/16 dev enp38s0
ip -br a
lo            UNKNOWN       127.0.0.1/8::1/128
enp38s0       UP            10.1.0.1/16
```

```
wlo1          UP          192.168.0.5/24
2a01:e35:2439:1510:4635:482d:bc79:3f71/64 fe80::a8a:b708:d2b2:6bc5/64
```

*La tarjeta de interfaz de red posee ahora una dirección IP.*

*Se muestra la tabla de enrutamiento:*

```
ip r show
default via 192.168.0.254 dev wlo1 proto dhcp metric 600
10.1.0.0/16 dev enp38s0 proto kernel scope link src 10.1.0.1
192.168.0.0/24 dev wlo1 proto kernel scope link src 192.168.0.5 metric 600
```

*La pasarela por defecto se encuentra en la red Wi-Fi `192.168.0.0/24` .*

*Se comprueba la comunicación en las dos redes:*

```
ping -c1 192.168.0.254
PING 192.168.0.254 (192.168.0.254) 56(84) bytes of data.
64 bytes from 192.168.0.254: icmp_seq=1 ttl=64 time=4.03 ms
--- 192.168.0.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 4.028/4.028/4.028/0.000 ms
```

*Se puede acceder a la pasarela por defecto.*

```
ping -c1 10.1.0.2
PING 10.1.0.2 (10.1.0.2) 56(84) bytes of data.
64 bytes from 10.1.0.2: icmp_seq=1 ttl=64 time=0.239 ms
--- 10.1.0.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.239/0.239/0.239/0.000 ms
```

*Se puede acceder a la máquina `10.1.0.2` .*

*On se connecte sur la machine `10.1.0.2` :*

```
ssh pba@10.1.0.2
pba@10.1.0.2's password: XXXX
Linux alpha 4.19.0-8-amd64 #1 SMP Debian 4.19.98-1 (2020-01-26) x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Fri Apr  3 12:57:53 2020 from 10.1.0.1
```

*El sistema remoto es una distribución Debian 10.*

*Se muestra la tabla de enrutamiento:*

```
ip route show
10.1.0.0/16 dev enp0s10 proto kernel scope link src 10.1.0.2
```

*El sistema no tiene pasarela por defecto, por lo tanto no puede comunicar fuera de su red/subred:*

```
ping -c1 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=1 ttl=64 time=0.156 ms
--- 10.1.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.156/0.156/0.156/0.000 ms
ping -c1 192.168.0.5
connect: La red es inaccesible
```

## b. Configuración de una interfaz multiredes

Un sistema Linux también puede pertenecer a distintas redes/subredes IP a través de una sola tarjeta de interfaz de red. Las diferentes redes/subredes comparten una misma red física, Ethernet por ejemplo. El sistema multired (*multihomed*) estará integrado en diferentes redes y podrá realizar o no el enrutamiento entre ellas, según su configuración y

la de los otros elementos en las redes/subredes.

Para ello, hay que configurar la tarjeta de interfaz de red con los parámetros adaptados a las diferentes redes IP en las que está integrado: dirección IP y máscara de subred. Configuraremos una pasarela por defecto y, opcionalmente, rutas específicas hacia algunas redes/subredes. Esto permitirá que el sistema pueda comunicarse con las diferentes redes.

*Ejemplo*

*Configuración de un sistema con dos redes IP a través de una tarjeta de interfaz de red Ethernet.*

*Se usará la red Ethernet para las dos redes IP:* `192.168.0.0/24` *y* `10.1.0.0/16` *.*

*Hay tres máquinas en la red Ethernet:*

- Un router con una dirección IP `192.168.0.254` , que no tiene ninguna ruta hacia la red `10.1.0.0/16` .
- Un servidor `CentOS 8` con acceso a las dos redes, a través de la misma tarjeta de interfaz de red:
  - `192.168.0.4/24`
  - `10.1.0.1/16`

    Pasarela por defecto: `192.168.0.254`
- Un servidor `Debian 10` que solamente tiene acceso a la red `10.1.0.0/16` :

  `10.1.0.2/16`

  Pasarela por defecto: `192.168.0.4`

*Configuración multired del servidor* `CentOS 8` *:*

*Se muestran las tarjetas de interfaz existentes:*

**ip -br a**

```
lo            UNKNOWN      127.0.0.1/8::1/128
enp38s0       UP           192.168.0.4/24 fe80::5508:86a9:e923:75b3/64
wlo1          DOWN
```

*La tarjeta Wi-Fi no está activa (`DOWN`), la tarjeta Ethernet `enp38s0` tiene una dirección IP de la red `192.168.0.0/24` .*

*Se muestra la ruta por defecto:*

```
ip route show default
default via 192.168.0.254 dev enp38s0 proto dhcp metric 100
```

*Añadimos la dirección de red `10.1.0.1/16` :*

```
ip a add 10.1.0.1/16 dev enp38s0
ip -br a
lo            UNKNOWN      127.0.0.1/8::1/128
enp38s0       UP           192.168.0.4/24 10.1.0.1/16 fe80::5508:86a9:e923:75b3/64
wlo1          DOWN
```

*El sistema tiene acceso a las dos redes IP a través de la misma tarjeta de interfaz de red Ethernet.*

*Se comprueba la comunicación con el router IP `192.168.0.254/24` :*

```
ping -c 1 192.168.0.254
PING 192.168.0.254 (192.168.0.254) 56(84) bytes of data.
64 bytes from 192.168.0.254: icmp_seq=1 ttl=64 time=0.328 ms
--- 192.168.0.254 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.328/0.328/0.328/0.000 ms
```

*Configuración del servidor `Debian 10` :*

*Se muestran las tarjetas de interfaz de red y sus configuraciones:*

```
ip -br a
lo               UNKNOWN      127.0.0.1/8::1/128
enp0s10          UP           10.1.0.2/16 2a01:e35:2439:1510:4d0:e3ed:e3d8:5f48/64
2a01:e35:2439:1510:21b:24ff:fe6a:7814/64 fe80::21b:24ff:fe6a:7814/64
wlp3s0           DOWN
```

*El sistema tiene una tarjeta de interfaz de red Ethernet activa, cuya dirección IP es* `10.1.0.2/16` .

*Se muestra su pasarela por defecto:*

```
ip route show default
```

*El sistema no tiene pasarela por defecto, por lo tanto no puede comunicar fuera de su red* `10.1.0.0/16` .

*Se comprueba la comunicación con el servidor CentOS 8 en la red IP 10.1.0.0/16:*

```
ping -c 1 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=1 ttl=64 time=0.225 ms
--- 10.1.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.225/0.225/0.225/0.000 ms
```

*Los dos sistemas pueden comunicar en la red* `10.1.0.0/16` .

*Se comprueba la comunicación con la otra dirección IP del servidor CentOS 8,* `192.168.0.4/16` *:*

```
ping -c 1 192.168.0.4
connect: La red es inaccesible
```

*La comunicación es imposible porque las dos direcciones no pertenecen a la misma red IP y el servidor Debian 10 no tiene pasarela por defecto.*

### c. Enrutamiento estático a través de un servidor multired

Un sistema Linux que está integrado en distintas redes/subredes IP puede realizar o no el enrutamiento entre ellas, según su configuración y la de los otros elementos presentes en las redes/subredes.

Si el enrutamiento (*forwarding*) entre las diferentes redes en las que participa está activo, cuando el sistema recibe un paquete IP destinado a una dirección IP que no forme parte de las suyas, el núcleo consultará su tabla de enrutamiento y, si encuentra una ruta, enviará el paquete a través de la interfaz correspondiente a dicha ruta.

Para que dos sistemas con redes diferentes puedan comunicar a través de un sistema multired, es necesario que sus tablas de enrutamiento respectivas estén configuradas correctamente.

La activación/desactivación de este enrutamiento estático depende de un parámetro dinámico del núcleo: `ip_forward`. Si este parámetro tiene un valor de `1` o `0`, el enrutamiento estará activado o desactivado.

Este parámetro se puede modificar dinámicamente escribiendo `0` o `1` en el pseudoarchivo `/proc/sys/net/ipv4/ip_forward`.

Se dice que el enrutamiento es estático porque el sistema multiredes no intercambia su información de enrutamiento con otros routers de su red. Existen diferentes programas en Linux que gestionan un protocolo de enrutamiento dinámico.

Otra condición para que el sistema pueda enrutar los paquetes entre las redes/subredes es configurar el servicio de cortafuegos para autorizar este enrutamiento (reglas de *forwarding*).

*Ejemplo*

*En el ejemplo anterior, se configuró un sistema CentOS 8 en multired. Este pertenecía a dos redes,* `10.1.0.0/16` *y* `192.168.0.0/24`*, y vamos a utilizarlo para enrutar paquetes entre esas dos redes.*

*Sistema CentOS 8:*

*Para evitar posibles bloqueos durante nuestros tests, pararemos el servicio* `firewalld`*:*

```
systemctl stop firewalld
```

*¿Está activado el enrutamiento estático?*

```
cat /proc/sys/net/ipv4/ip_forward
0
```

*El enrutamiento no está activo. Procedemos a activarlo dinámicamente:*

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

*Se muestra la tabla de enrutamiento:*

```
ip -br route show
default via 192.168.0.254 dev enp38s0 proto dhcp metric 100
10.1.0.0/16 dev enp38s0 proto kernel scope link src 10.1.0.1
192.168.0.0/24 dev enp38s0 proto kernel scope link src 192.168.0.4 metric 100
```

*El sistema tiene una pasarela por defecto, y una ruta para cada red en la que participa.*

*En el servidor Debian 10 del ejemplo anterior:*

*Para evitar posibles bloqueos durante nuestros tests, pararemos el servicio* `firewalld` *:*

```
systemctl stop firewalld
```

*Mostramos las tarjetas de interfaz de red y su configuración:*

```
ip -br a
lo               UNKNOWN        127.0.0.1/8::1/128
enp0s10          UP             10.1.0.2/16
2a01:e35:2439:1510:4d0:e3ed:e3d8:5f48/64
2a01:e35:2439:1510:21b:24ff:fe6a:7814/64 fe80::21b:24ff:fe6a:7814/64
wlp3s0           DOWN
```

*El sistema tiene una tarjeta de interfaz de red Ethernet activa, con la dirección IP* `10.1.0.2/16` *.*

*Comprobamos la comunicación con el servidor CentOS 8 en la red IP* `10.1.0.0/16` *:*

```
ping -c 1 10.1.0.1
PING 10.1.0.1 (10.1.0.1) 56(84) bytes of data.
64 bytes from 10.1.0.1: icmp_seq=1 ttl=64 time=0.225 ms
--- 10.1.0.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.225/0.225/0.225/0.000 ms
```

*Los dos sistemas pueden comunicar en la red* `10.1.0.0/16` *.*

*Se comprueba la comunicación con la otra dirección IP del servidor CentOS 8,* `192.168.0.4/16` *:*

```
ping -c 1 192.168.0.4
connect: La red es inaccesible
```

*La comunicación no se puede realizar, porque las dos máquinas no se encuentran en la misma red IP y el servidor Debian 10 no tiene pasarela por defecto:*

```
ip route show
10.1.0.0/16 dev enp0s10 proto kernel scope link src 10.1.0.2
```

*Añadimos una pasarela por defecto, la dirección de red del servidor CentOS 8 en la red* `10.1.0.0/16` *:*

```
ip route add default via 10.1.0.1
ip route show
default via 10.1.0.1 dev enp0s10
10.1.0.0/16 dev enp0s10 proto kernel scope link src 10.1.0.2
```

*Se comprueba de nuevo la comunicación:*

```
ping -c 1 192.168.0.4
PING 192.168.0.4 (192.168.0.4) 56(84) bytes of data.
64 bytes from 192.168.0.4: icmp_seq=1 ttl=64 time=0.195 ms
--- 192.168.0.4 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.195/0.195/0.195/0.000 ms
```

*En una tercera maquina (Windows) de la red* `192.168.0.0/16` *, añadimos en la tabla de enrutamiento una entrada hacia la red* `10.1.0.0/16` *, a través de la dirección de red* `192.168.0.4` *.*

*Se comprueba la comunicación con el servidor Debian 10 de la red* `10.1.0.0/16` *:*

```
ping 10.1.0.2
Haciendo ping a 10.1.0.2 con 32 bytes de datos:
Respuesta de 10.1.0.2: bytes=32 tiempo=1ms TTL=63
Respuesta de 10.1.0.2: bytes=32 tiempo=17ms TTL=64
Respuesta de 10.1.0.2: bytes=32 tiempo=3ms TTL=64
Respuesta de 10.1.0.2: bytes=32 tiempo=3ms TTL=64
```

```
Estadísticas de ping para 10.1.0.2:
  Paquetes: enviados = 4, recibidos = 4, perdidos = 0 (0% perdidos),
Tiempos aproximados de ida y vuelta en milisegundos:
  Mínimo = 1ms, Máximo = 17ms, Media = 6ms
```

*Nos conectamos desde la máquina Windows, en SSH, en el servidor Debian 10:*

```
ssh pba@10.1.0.2
pba@10.1.0.2's password: XXXX
Linux alpha 4.19.0-8-amd64 #1 SMP Debian 4.19.98-1 (2020-01-26) x86_64
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
No mail.
Last login: Sat Apr  4 16:09:51 2020 from 10.1.0.1
pba@alpha:~$
```

*En el servidor CentOS 8, se desactiva dinámicamente el enrutamiento:*

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

*Se comprueba de nuevo la comunicación entre la máquina Windows de la red `192.1.0.0/24` y el servidor Debian 10 de la red `10.1.0.0/16` :*

```
ping 10.1.0.2
Haciendo ping a 10.1.0.2 con 32 bytes de datos:
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.
Tiempo de espera agotado para esta solicitud.
Estadísticas de ping para 10.1.0.2:
  Paquetes: enviados = 4, recibidos = 0, perdidos = 4 (100% perdidos),
```

*El servidor CentOS 8 ya no enruta los paquetes entre las dos redes.*

# 3. Herramientas de diagnóstico y de supervisión de la actividad de las redes

Un cierto número de comandos y de herramientas permiten supervisar la actividad de las redes de un sistema Linux, comprobar su buen funcionamiento o establecer un diagnóstico en caso de problemas.

Ya se han estudiado los comandos `arp` para la supervisión de la tabla ARP y `ping` para comprobar la comunicación con hosts remotos usando mensajes ICMP.

- Se puede supervisar la actividad de red con los comandos `netstat`, `ss` y `lsof` en distintos niveles de protocolos.
- Distintas herramientas permiten capturar el tráfico de red de la máquina local y de las máquinas remotas (`tcpdump`, `wireshark`) y comprobar las capacidades de comunicación con una máquina remota (`nc`, `nmap`).

Algunos de estos comandos (*`tcpdump`*, *`nmap`*…) permiten registrar intercambios de información de otras máquinas en las redes y someter a tests a máquinas remotas que podrían ser considerados potencialmente como intrusivos. Por lo tanto, tienen que ser utilizados con prudencia y respetando estrictamente las leyes.

### a. Supervisión de la actividad con netstat

El comando `netstat` permite visualizar el estado y la actividad de la red del sistema local en diferentes niveles, gracias a sus numerosas opciones.

Este comando está considerado como en vía de obsolescencia y ya no se encuentra instalado por defecto en las versiones recientes de la mayoría de las distribuciones. Forma parte del paquete `net-tools` del que ya se ha hablado. Este comando ha sido reemplazado esencialmente por el comando `ss`, que será visto más adelante.

### *Sintaxis*

```
netstat [ -Opciones ]
```

### *Parámetros principales*

| | |
|---|---|
| `--inet[6]` | Solamente muestra la información IPv4 (v6), y no la relativa al tipo Unix. |
| `-n` | Muestra los valores numéricos en lugar de los nombres. |
| `-i` | Actividad de las interfaces de red. |
| `-r` | Tabla de enrutamiento. |
| `-s` | Estadísticas por protocolo. |
| `-a` | Conexiones y sockets. |
| `-p` | Procesos vinculados a los sockets. |
| `-l` | Sockets con un proceso en escucha. |
| `-c` | Vista en continuo, actualizada cada segundo. |

*Descripción*

Sin argumento, el comando muestra todos los sockets activos, de tipo red o de tipo Unix.

Las opciones principales son:

-n

No hace ninguna resolución de nombres, el comando solo mostrará las direcciones IP. Esto limita el tráfico de red y hace más eficiente la vista en pantalla.

--inet[6]

Solamente mostrará la información relativa al tipo red, sin mostrar las vinculadas a los sockets de modo Unix, usados para la comunicación entre procesos internos.

-i

Muestra la información y las estadísticas de actividad de las diferentes interfaces de red.

-r

Muestra la tabla de enrutamiento del núcleo, con el estado de las diferentes rutas.

-s

Muestra las estadísticas de uso por protocolo.

-a

Muestra todos los sockets activos, así como los sockets en los que se encuentra un proceso en espera, o en escucha (proceso servidor en espera de una conexión TCP, o mensaje UDP).

-p

Muestra todos los procesos vinculados a los sockets.

*Ejemplos*

*Se usa el comando `netstat` con, si fuera necesario, las opciones `-n` (sin resolución de nombres) e --`inet` (solamente los sockets de red), en una distribución CentOS 8.*

*Vista de la información en las diferentes interfaces:*

```
netstat -in
Kernel Interface table
Iface       MTU    RX-OK RX-ERR RX-DRP RX-OVR    TX-OK TX-ERR TX-DRP TX-OVR Flg
enp38s0    1500    88163      0     16 0          33589      0      0      0 BMRU
lo        65536     5716      0      0 0           5716      0      0      0 LRU
wlo1       1500    65391      0      0 0            550      0      0      0 BMRU
```

*El sistema dispone de dos interfaces de red, además de la de loopback `lo`. Se puede ver la actividad en la red, mensajes transmitidos (`TX`) y recibidos (`RX`). Los indicadores (flags) muestran el estado y la configuración de las interfaces: `U` activa (Up), `L` loopback, `B` broadcast activo, `M` multicast activo y `R` enrutamiento activo.*

*Vista de la información de enrutamiento:*

```
netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         192.168.0.254   0.0.0.0         UG        0 0          0 enp38s0
0.0.0.0         192.168.0.254   0.0.0.0         UG        0 0          0 wlo1
10.1.0.0        0.0.0.0         255.255.0.0     U         0 0          0 enp38s0
192.168.0.0     0.0.0.0         255.255.255.0   U         0 0          0 enp38s0
192.168.0.0     0.0.0.0         255.255.255.0   U         0 0          0 wlo1
```

*El sistema está presente en dos redes IP, `192.168.0.0/24` y `10.1.0.0/16`. La pasarela por defecto (`Destination = 0.0.0.0`) es la dirección de red `192.168.0.254`. Los indicadores indican el estado de la ruta: `U` activa (Up), `G` mediante una pasarela (Gateway).*

*Vista de la información correspondiente a los sockets de red, incluyendo los que tienen procesos en espera, con los procesos asociados a dichos sockets:*

```
netstat -anput
```

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:111             0.0.0.0:*               LISTEN      1/systemd
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      1247/sshd
tcp        0      0 127.0.0.1:631           0.0.0.0:*               LISTEN      1244/cupsd
tcp        0     64 192.168.0.4:22          192.168.0.24:64586      ESTABLISHED 7725/sshd: root [pr
tcp        0      0 192.168.0.4:22          192.168.0.24:64241      ESTABLISHED 6870/sshd: root [pr
tcp        0      0 192.168.0.5:22          192.168.0.24:64017      ESTABLISHED 6705/sshd: root [pr
tcp        0      0 10.1.0.1:34868          10.1.0.2:22             ESTABLISHED 6922/ssh
tcp        0      0 192.168.0.4:35210       192.168.0.39:3260       ESTABLISHED 1245/iscsid
udp        0      0 0.0.0.0:5353            0.0.0.0:*                           1039/avahi-daemon:
udp        0      0 0.0.0.0:36305           0.0.0.0:*                           1039/avahi-daemon:
udp        0      0 192.168.0.5:68          0.0.0.0:*                           1229/NetworkManager
udp        0      0 192.168.0.4:68          0.0.0.0:*                           1229/NetworkManager
udp        0      0 0.0.0.0:111             0.0.0.0:*                           1/systemd
```

*Observamos conexiones SSH activas. Si la dirección de red local tiene un puerto `22`, se trata de una conexión entrante hacia el servidor SSH; si el puerto es un número elevado, se trata de un puerto atribuido dinámicamente a un cliente SSH, y es una conexión entrante. Esto se puede confirmar gracias al proceso vinculado al socket, `sshd` en el caso del servidor y `ssh` en el caso del cliente. Se puede observar también una conexión iSCSI.*

*Varios servidores se encuentran a la espera de solicitudes (dirección remota con * en el número del puerto) en distintos puertos, en TCP: `sshd`, `systemd` y `cupsd`, en UDP: `avahi-daemon`, `NetworkManager` y `systemd`.*

### b. Supervisión de la actividad con ss

Este comando (*socket statistics*) permite visualizar el estado de la actividad de la red del sistema local, en lo que respecta a los sockets.

Este comando reemplaza en gran medida a `netstat` y proporciona funcionalidades suplementarias. Las opciones de `netstat` que no son relativas a los sockets (`-i`, `-r`…) han sido reemplazadas por el comando `ip` y los objetos `link`, `address` y `route`.

### Sintaxis

```
ss [ -Options ]
```

Parámetros principales

| | |
|---|---|
| `-f FamSoc` | Solamente muestra los datos para el tipo de socket `FamSoc` (`inet`, `inet6`, unix). |
| `-n` | Muestra los valores numéricos en lugar de los nombres. |
| `-s` | Estadísticas por protocolo. |
| `-a` | Conexiones y sockets. |
| `-l` | Sockets con un proceso en escucha. |
| `-p` | Procesos vinculados a los sockets. |
| `-i` | Información detallada de la capa TCP. |

### Descripción

Sin ningún argumento, el comando muestra todos los sockets activos, de red o de tipo

Unix.

Las opciones principales son:

-n

Permite no hacer resoluciones de nombres; el comando solo muestra las direcciones, lo que limita el tráfico de red y mejora el rendimiento de la visualización.

-f FamSoc

Limita la información a `FamSoc`: `-f inet` y/o `-f inet6`, no muestra la información relativa a los sockets en modo Unix, utilizada para la comunicación de los procesos locales.

-s

Muestra las estadísticas de uso por protocolo.

-a

Muestra todos los sockets activos, así como los sockets en los que un proceso está en espera (proceso en modo servidor que espera una conexión TCP, o un mensaje UDP).

-l

Muestra los sockets en los que un proceso está en espera (proceso en modo servidor que espera una conexión TCP, o un mensaje UDP).

-p

Muestra los procesos vinculados a los sockets.

-i

Muestra información detallada, en la capa TCP, para las conexiones activas.

*Ejemplos*

*(Este ejemplo está basado en el anterior, usando los nuevos comandos en lugar de `netstat`.)*

*Se usa el comando `ss`, con las opciones `-n` (sin resolución de nombres) y `-f inet` (solamente los sockets IPv4), en una distribución CentOS 8.*

*Se comprueba primero la configuración de las interfaces y la tabla de enrutamiento, usando para ello el comando `ip`:*

*Vista de la información en las diferentes interfaces:*

```
ip -br link
lo            UNKNOWN        00:00:00:00:00:00 <LOOPBACK,UP,LOWER_UP>
enp38s0       UP             e4:11:5b:50:13:32 <BROADCAST,MULTICAST,UP,LOWER_UP>
wlo1          UP             9c:b7:0d:bb:2b:67 <BROADCAST,MULTICAST,UP,LOWER_UP>
```

*El sistema dispone de dos interfaces de red, además de la de loopback `lo`.*

*Vista de la información de enrutamiento:*

```
ip route
default via 192.168.0.254 dev enp38s0 proto dhcp metric 100
default via 192.168.0.254 dev wlo1 proto dhcp metric 600
10.1.0.0/16 dev enp38s0 proto kernel scope link src 10.1.0.1
192.168.0.0/24 dev enp38s0 proto kernel scope link src 192.168.0.4 metric 100
192.168.0.0/24 dev wlo1 proto kernel scope link src 192.168.0.5 metric 600
```

*El sistema está presente en dos redes IP diferentes, `192.168.0.0/24` y*

`10.1.0.0/16`. *La pasarela por defecto (Destino =* `0.0.0.0`*) es la dirección IP* `192.168.0.254`.

*Vista de la información de los sockets de red, incluyendo aquellos donde los procesos se encuentran en espera, con los procesos asociados:*

```
ss -f inet -nap
NetidState  Recv-Q  Send-Q        Local Address:Port
Peer Address:Port
udp  UNCONN 0       0                 0.0.0.0:5353
0.0.0.0:*     users:(("avahi-daemon",pid=1039,fd=15))
udp  UNCONN 0       0                 0.0.0.0:36305
0.0.0.0:*     users:(("avahi-daemon",pid=1039,fd=17))
udp  UNCONN 0       0           192.168.0.5%wlo1:68
0.0.0.0:*     users:(("NetworkManager",pid=1229,fd=24))
udp  UNCONN 0       0        192.168.0.4%enp38s0:68
0.0.0.0:*     users:(("NetworkManager",pid=1229,fd=22))
udp  UNCONN 0       0                 0.0.0.0:111
0.0.0.0:*     users:(("rpcbind",pid=987,fd=5),("systemd",pid=1,fd=37))
tcp  LISTEN 0       128               0.0.0.0:111
0.0.0.0:*     users:(("rpcbind",pid=987,fd=4),("systemd",pid=1,fd=36))
tcp  LISTEN 0       128               0.0.0.0:22
0.0.0.0:*     users:(("sshd",pid=1247,fd=5))
tcp  LISTEN 0       5                 127.0.0.1:631
0.0.0.0:*     users:(("cupsd",pid=1244,fd=10))
tcp  ESTAB  0       0             192.168.0.4:22
192.168.0.24:61397  users:(("sshd",pid=20008,fd=5),("sshd",pid=20004,fd=5))
tcp  ESTAB  0       0             192.168.0.4:35330
192.168.0.39:3260   users:(("iscsid",pid=1245,fd=12))
tcp  ESTAB  0       0             192.168.0.4:22
192.168.0.24:61335  users:(("sshd",pid=19603,fd=5),("sshd",pid=19579,fd=5))
tcp  ESTAB  0       0              10.1.0.1:35170
10.1.0.2:22     users:(("ssh",pid=20003,fd=5))
```

*Observamos conexiones SSH activas. Si la dirección de red local tiene un puerto* `22`*, se trata de una conexión entrante hacia el servidor SSH; si el puerto es un número elevado, se trata de un puerto atribuido dinámicamente a un cliente SSH, y es una conexión saliente. Esto se puede confirmar gracias al proceso ligado al socket,* `sshd` *en el caso del servidor,* `ssh` *en*

*el caso del cliente. Se puede observar también una conexión iSCSI.*

*Varios servidores se encuentran a la espera de solicitudes (dirección remota con * en el número del puerto) en distintos puertos, en TCP:* `sshd`, `systemd` *y* `cupsd`, *en UDP:* `avahi-daemon`, `NetworkManager` *y* `systemd`.

### c. Supervisión de los sockets abiertos usando lsof -i

El comando `lsof` permite obtener información sobre los archivos abiertos por el sistema.

Entre sus numerosas opciones, la opción `-i` permite seleccionar solamente los archivos abiertos de tipo socket de red. Se puede completar la opción con diferentes criterios de selección de los sockets (versión IP, dirección asociada, etc.).

La opción `-n` permite desactivar la resolución de nombres, lo que acelera la vista.

*Ejemplo*

*Archivos de tipo sockets abiertos en un sistema Debian 10:*

```
lsof -ni
COMMAND    PID       USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
avahi-dae  476      avahi   12u  IPv4  19295      0t0  UDP *:mdns
avahi-dae  476      avahi   13u  IPv6  19296      0t0  UDP *:mdns
avahi-dae  476      avahi   14u  IPv4  19297      0t0  UDP *:59786
avahi-dae  476      avahi   15u  IPv6  19298      0t0  UDP *:38283
tgtd       547       root    6u  IPv4  19876      0t0  TCP *:iscsi-target (LISTEN)
tgtd       547       root    7u  IPv6  19877      0t0  TCP *:iscsi-target (LISTEN)
tgtd       547       root   11u  IPv4  51729      0t0  TCP 192.168.0.39:
iscsi-target->192.168.0.4:35330 (ESTABLISHED)
sshd       562       root    3u  IPv4  21096      0t0  TCP *:ssh (LISTEN)
sshd       562       root    4u  IPv6  21098      0t0  TCP *:ssh (LISTEN)
dhclient   566       root    6u  IPv4  19966      0t0  UDP *:bootpc
exim4      919 Debian-exim    3u  IPv4  23787      0t0  TCP 127.0.0.1:smtp (LISTEN)
exim4      919 Debian-exim    4u  IPv6  23788      0t0  TCP [::1]:smtp (LISTEN)
sshd      3669       root    3u  IPv4  51755      0t0  TCP 10.1.0.2:
ssh->10.1.0.1:35170 (ESTABLISHED)
sshd      3687        pba    3u  IPv4  51755      0t0  TCP 10.1.0.2:
ssh->10.1.0.1:35170 (ESTABLISHED)
```

### d. Pruebas de comunicación con ncat (nc)

El comando `ncat` o `nc` (*netcat*) es una herramienta multiusos que permite establecer comunicaciones a través de sockets, locales o de red en TCP y en UDP, en IPv4 o IPv6, en modo cliente o servidor, especificando el puerto que se tiene que usar.

Con sus numerosas opciones, este comando permite comprobar diferentes tipos de comunicación, en particular a través de las redes.

Existen diferentes programas que implementan el comando `nc`, con opciones diferentes. El que se describe aquí es `ncat`, que se encuentra en el paquete de software `nmap-ncat`.

*Sintaxis*

```
ncat|nc [ -Opciones ] [ Host ] [ Puerto ]
```

*Parámetros principales*

| | |
|---|---|
| `Host` | Dirección o nombre de máquina. |
| `Puerto` | Número de puerto (por defecto `31337`). |
| `-n` | Muestra los valores numéricos en lugar de los nombres. |
| `-4\|6` | IPv4 o IPv6. |
| `-u` | UDP en lugar de TCP. |
| `-l` | Modo escucha (servidor). |
| `-c Cmd` | Ejecuta el comando `Cmd`. |
| `-p Puerto` | Número de puerto origen. |
| `-s Dir` | Dirección origen. |

*Descripción*

Como argumento, según el caso de uso, el comando espera una dirección IP o un nombre de host y/o un número de puerto. Por defecto, usa el protocolo de transporte TCP. El número de puerto por defecto es 31337.

En modo cliente, el comando envía al servidor cada línea tecleada. En modo servidor, muestra cada línea recibida del cliente.

Las principales opciones del comando son:

-n

Sin resolución de nombres (*noDNS*), el comando solamente usa direcciones, lo que limita el tráfico de red y mejora el rendimiento de la visualización.

-4|6

Especifica la versión de IP que se usará.

-u

Usa el protocolo de transporte UDP.

-l

El comando se pone en espera en el número de puerto especificado o 31337 por defecto.

-c Cmd

Ejecuta el comando `Cmd` cuando la conexión se ha establecido.

-p

Especifica el número de puerto de origen que se usará.

-s

Especifica la dirección de origen que se usará.

*Ejemplos*

*Test de comunicación en TCP, en el puerto por defecto de `nc`, entre dos hosts.*

*(Para evitar cualquier riesgo de interferencias, se ha desactivado el firewall en las máquinas*

*de test.)*

*En la máquina CentOS 8, de dirección* `10.1.0.1`*, se usa* `nc` *en modo servidor, en el puerto por defecto:*

```
nc -l
```

*En la máquina Debian 10, nos conectamos hacia la máquina CentOS 8, en el puerto por defecto:*

```
nc 10.1.0.1
Prueba de comunicación
```

*En el lado CentOS 8, se ve lo siguiente:*

```
Prueba de comunicación
```

*Tecleamos una línea en el servidor:*

```
aaaaa
```

*En el lado del cliente, el comando muestra:*

```
aaaaa
```

*Los dos sistemas están conectados en TCP, en el puerto del servidor por defecto. Se puede comprobar desde otra ventana de terminal en la máquina CentOS 8:*

```
ss -inet -p | grep nc
ESTAB  0       0              10.1.0.1:31337
10.1.0.2:35018    users:(("nc",pid=30557,fd=5)) ino:164944 sk:264 <->
```

*Vemos que la conexión está activa (`ESTAB`) entre las direcciones IP `10.1.0.1` y `10.1.0.2`, el puerto del servidor es 31337 y el puerto del cliente (atribuido dinámicamente) es 35018.*

*Para terminar la conexión, tecleamos [Ctrl] D, en el client.*

*Uso de la opción `-c`:*

*Esta opción permite ejecutar un comando dentro de la conexión. Desde el cliente, el comando se ejecuta escribiendo su salida estándar en el socket de la conexión. Desde el servidor, el comando se ejecuta utilizando el socket conexión como entrada estándar.*

*Utilizamos esta opción, desde el cliente, para enviar al servidor la lista de procesos activos:*

*En el servidor:*

```
nc -l
```

*En el cliente:*

```
nc -c "ps -ef" 10.1.0.1
```

*Vista desde el servidor:*

```
UID        PID  PPID  C STIME TTY          TIME CMD
root         1     0  0 abril04 ?      00:00:10 /sbin/init
root         2     0  0 abril04 ?      00:00:00 [kthreadd]
root         3     2  0 abril04 ?      00:00:00 [rcu_gp]
root         4     2  0 abril04 ?      00:00:00 [rcu_par_gp]
root         6     2  0 abril04 ?      00:00:00 [kworker/0:0H-kblockd]
root         8     2  0 abril04 ?      00:00:00 [mm_percpu_wq]
root         9     2  0 abril04 ?      00:00:00 [ksoftirqd/0]
[...]
root      4984  4658  0 10:42 pts/0    00:00:00 nc -c ps -ef 10.1.0.1
root      4985     2  0 10:42 ?        00:00:00 [kworker/0:2-events]
root      4986  4984  0 10:42 pts/0    00:00:00 sh -c ps -ef
root      4987  4986  0 10:42 pts/0    00:00:00 ps -ef
```

*Se ha obtenido el resultado correctamente del comando `ps -ef` ejecutado desde el cliente.*

*Uso del comando para comprobar la accesibilidad a un servidor HTTP:*

*Nos conectamos en el puerto 80 de uno de los servidores HTTP de Google y solicitamos la página de inicio:*

```
nc www.google.com 80
GET /
HTTP/1.0 200 OK
Date: Tue, 07 Apr 2020 08:46:34 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=ISO-8859-1
P3P: CP="This is not a P3P policy! See g.co/p3phelp for more info."
Server: gws
X-XSS-Protection: 0
X-Frame-Options: SAMEORIGIN
Set-Cookie: 1P_JAR=2020-04-07-08; expires=Thu, 07-May-2020 08:46:34 GMT;
path=/; domain=.google.com; Secure
Set-Cookie: NID=201=tHy-5jJ104pD35b8rFBxwTXY62gAUgVDMEEZ-
iI4FX5c6Ge5Rj7qhKsSD-ngfu6142nyOOE3NGdnP8Zr5CVasXeq4LQQsthN-
aq6mtTD2BWU2POyNaLigejSN3rh7HKhrZlykA8STSdwn-
FAwbCXt4dH12CLeQ8WRD1Vn2fVdNY; expires=Wed, 07-Oct-2020 08:46:34 GMT;
path=/; domain=.google.com; HttpOnly
Accept-Ranges: none
Vary: Accept-Encoding
[...]
```

### e. Comprobar los puertos abiertos remotos: el comando nmap

El comando `nmap` (*Network Mapper*), integrado en el paquete `nmap`, es una herramienta muy potente de exploración de una red y de auditoría de seguridad. Permite determinar las máquinas activas en la red y los servicios de redes disponibles en esas máquinas.

El comando permite obtener, en particular, una tabla de los puertos de cada máquina objetivo, con su estado (abierto, cerrado o filtrado por un firewall). También puede opcionalmente buscar y mostrar información más detallada: sistema operativo, programa

en escucha en cada puerto abierto, incluyendo los números de versión.

El uso de este comando tiene que hacerse con prudencia y respetando las obligaciones legales, porque comprobar el estado de los puertos de una máquina sin autorización puede ser considerado como un intento de acceso ilícito y ser objeto de procedimientos penales.

### *Sintaxis*

```
nmap [ Typescan ] [ -Opciones ] [ Objetivos ]
```

### *Parámetros principales*

| | |
|---|---|
| `TypeScan` | Tipo de test. |
| `Objetivos` | Dirección(es) o nombre(s) de máquina(s). |
| `-Opciones` | Opciones del escaneo. |

### *Descripción*

El comando puede explorar máquinas activas en una red y/o comprobar el estado de los puertos en esas máquinas (*scan*).

Los objetivos puede ser direcciones o nombres de máquinas, identificadores de red o subred y rangos de direcciones.

La exploración puede durar bastante tiempo. En modo interactivo, teclear CTRL/D provoca la visualización de la información del estado del escaneo y los resultados obtenidos.

Existe un gran número de opciones que permiten especificar el tipo de exploración o de escaneo y configurarlos.

Algunas opciones habituales:

-T[0-5]

Grado de velocidad/agresividad/eficacia de la exploración. Por defecto, el nivel es `3`. Los niveles `4` y `5` presuponen una red muy eficiente. Los niveles `0` y `1` son muy lentos.

-A

Detección del sistema operativo y de las versiones de los programas.

-sL

Da la lista de las máquinas objetivo.

-sP

Comprueba las máquinas activas.

-sV

Comprueba los puertos abiertos, el programa que está en escucha y la versión de este.

-n

No hace resolución de los nombres de hosts.

-p NúmPuertos

Solamente hará un escáner de la lista de puertos o del rango especificado `NœmPuertos`.

-exclude Hosts

Excluye las máquinas de la lista `Hosts`.

*Ejemplo*

*Exploración de un rango de direcciones:*

**nmap -A 192.168.0.1-40**
Starting Nmap 7.70 ( https://nmap.org ) at 2020-04-07 10:24 BST

Nmap scan report for 192.168.0.24
Host is up (0.013s latency).
Not shown: 999 filtered ports
PORT STATE SERVICE VERSION
5357/tcp open http Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-server-header: Microsoft-HTTPAPI/2.0
|_http-title: Service Unavailable
MAC Address: DC:A2:66:67:1B:1F (Unknown)
Warning: OSScan results may be unreliable because we could not find at least
1 open and 1 closed port
Device type: specialized|general purpose
Running (JUST GUESSING): AVtech embedded (87%), Microsoft Windows XP (87%),
FreeBSD 6.X|10.X (86%)
OS CPE: cpe:/o:microsoft:windows_xp::sp2 cpe:/o:freebsd:freebsd:6.2
cpe:/o:freebsd:freebsd:10.3
Aggressive OS guesses: AVtech Room Alert 26W environmental monitor (87%),
Microsoft Windows XP SP2 (87%), FreeBSD 6.2-RELEASE (86%), FreeBSD
10.3-STABLE (85%)
No exact OS matches for host (test conditions non-ideal).
Network Distance: 1 hop
Service Info: OS: Windows; CPE: cpe:/o:microsoft:windows

TRACEROUTE
HOP RTT ADDRESS
1 13.44 ms 192.168.0.24

```
Nmap scan report for 192.168.0.39
Host is up (0.00019s latency).
Not shown: 998 closed ports
PORT     STATE SERVICE VERSION
22/tcp   open  ssh     OpenSSH 7.9p1 Debian 10+deb10u2 (protocol 2.0)
| ssh-hostkey:
|   2048 9e:0a:2e:ce:8b:ba:44:f8:3b:07:67:74:46:2f:60:02 (RSA)
|   256 5c:4d:59:8c:b8:8c:4f:1f:f8:52:63:5f:96:e5:8e:94 (ECDSA)
|_  256 30:c8:29:f0:9c:97:f4:8b:2f:37:64:b8:20:ae:04:44 (ED25519)
3260/tcp open  iscsi?
| iscsi-info:
|   iqn.2008-09.com.example:server.target1:
|     Address: 192.168.0.39:3260,1
|_    Authentication: NOT required
MAC Address: 00:1B:24:6A:78:14 (Quanta Computer)
No exact OS matches for host (If you know what OS is running on it,
see https://nmap.org/submit/ ).
TCP/IP fingerprint:
OS:SCAN(V=7.70%E=4%D=4/7%OT=22%CT=1%CU=37904%PV=Y%DS=1%DC=D%G=Y%M=001B24%TM
OS:=5E8C484F%P=x86_64-redhat-linux-gnu)SEQ(SP=FF%GCD=1%ISR=109%TI=Z%CI=Z%TS
OS:=A)SEQ(SP=FF%GCD=1%ISR=109%TI=Z%CI=Z%II=I%TS=A)OPS(O1=M5B4ST11NW7%O2=M5B
OS:4ST11NW7%O3=M5B4NNT11NW7%O4=M5B4ST11NW7%O5=M5B4ST11NW7%O6=M5B4ST11)WIN(W
OS:1=FE88%W2=FE88%W3=FE88%W4=FE88%W5=FE88%W6=FE88)ECN(R=Y%DF=Y%T=40%W=FAF0%
OS:O=M5B4NNSNW7%CC=Y%Q=)T1(R=Y%DF=Y%T=40%S=O%A=S+%F=AS%RD=0%Q=)T2(R=N)T3(R=
OS:N)T4(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T5(R=Y%DF=Y%T=40%W=0%S=Z%A
OS:=S+%F=AR%O=%RD=0%Q=)T6(R=Y%DF=Y%T=40%W=0%S=A%A=Z%F=R%O=%RD=0%Q=)T7(R=Y%D
OS:F=Y%T=40%W=0%S=Z%A=S+%F=AR%O=%RD=0%Q=)U1(R=Y%DF=N%T=40%IPL=164%UN=0%RIPL
OS:=G%RID=G%RIPCK=G%RUCK=G%RUD=G)IE(R=Y%DFI=N%T=40%CD=S)

Network Distance: 1 hop
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

TRACEROUTE
HOP RTT     ADDRESS
1   0.19 ms 192.168.0.39

Nmap scan report for 192.168.0.4
Host is up (0.000010s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE VERSION
```

```
22/tcp  open  ssh     OpenSSH 8.0 (protocol 2.0)
| ssh-hostkey:
|   3072 fb:6e:b5:75:e5:e5:8e:6a:61:f2:f6:36:19:32:1a:7b (RSA)
|   256 76:22:2e:35:cd:b1:95:09:df:20:c3:42:56:3c:5a:6d (ECDSA)
|_  256 25:d9:86:e5:10:dc:81:85:92:8c:8e:84:ec:3b:3e:db (ED25519)
111/tcp open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4       111/tcp  rpcbind
|_  100000  2,3,4       111/udp  rpcbind
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.10
Network Distance: 0 hops

Nmap scan report for 192.168.0.5
Host is up (0.000010s latency).
Not shown: 998 closed ports
PORT    STATE SERVICE VERSION
22/tcp  open  ssh     OpenSSH 8.0 (protocol 2.0)
| ssh-hostkey:
|   3072 fb:6e:b5:75:e5:e5:8e:6a:61:f2:f6:36:19:32:1a:7b (RSA)
|   256 76:22:2e:35:cd:b1:95:09:df:20:c3:42:56:3c:5a:6d (ECDSA)
|_  256 25:d9:86:e5:10:dc:81:85:92:8c:8e:84:ec:3b:3e:db (ED25519)
111/tcp open  rpcbind 2-4 (RPC #100000)
| rpcinfo:
|   program version   port/proto  service
|   100000  2,3,4       111/tcp  rpcbind
|_  100000  2,3,4       111/udp  rpcbind
Device type: general purpose
Running: Linux 3.X
OS CPE: cpe:/o:linux:linux_kernel:3
OS details: Linux 3.7 - 3.10
Network Distance: 0 hops

Post-scan script results:
| ssh-hostkey: Possible duplicate hosts
| Key 3072 fb:6e:b5:75:e5:e5:8e:6a:61:f2:f6:36:19:32:1a:7b (RSA) used by:
|   192.168.0.4
```

```
|   192.168.0.5
| Key 256 76:22:2e:35:cd:b1:95:09:df:20:c3:42:56:3c:5a:6d (ECDSA) used by:
|   192.168.0.4
|   192.168.0.5
| Key 256 25:d9:86:e5:10:dc:81:85:92:8c:8e:84:ec:3b:3e:db (ED25519) used by:
|   192.168.0.4
|_  192.168.0.5
OS and Service detection performed. Please report any incorrect results at
https://nmap.org/submit/ .
Nmap done: 40 IP addresses (4 hosts up) scanned in 406.29 seconds
```

La exploración ha detectado cuatro direcciones en el rango especificado, éstas pertenecen a tres sistemas: un Windows y dos Linux. Existen diferentes puertos abiertos en esos sistemas (HTTP, SSH, iSCSI, RPC…).

### f. Captura de tráfico de red: tcpdump

Existen diferentes herramientas en Linux, con interfaz gráfica o en modo de línea de comandos, que permiten capturar el tráfico de red, de la máquina local y/o de las máquinas remotas. La mayoría de estas herramientas se basan en una biblioteca especializada, `libpcap`.

El comando `tcpdump` se usa en modo de línea de comandos.

*Sintaxis*

```
tcpdump [ -Opciones ] [ Selección ]
```

*Parámetros principales*

| | |
|---|---|
| `Selección` | Criterios de selección de los paquetes. |
| `-Opciones` | Opciones. |

*Descripción*

Por defecto, el comando muestra en la salida estándar un informe por cada paquete capturado, con datos sobre la captura (fecha, emisor y receptor). Para interrumpir la captura, hay que teclear CTRL/C o enviar al proceso una señal INT o TERM.

También se puede guardar el tráfico en un archivo, y después se puede interpretar dicho archivo con el mismo comando.

La cadena de caracteres `Selección` permite definir los criterios de selección de los paquetes que se quieran capturar: emisor, receptor, protocolo, red…, dichos criterios pueden ser combinados con operadores lógicos (`or`, `and`), con valores numéricos o umbrales.

Algunas opciones habituales:

-n

Sin resolución de nombres.

-v|vv|vvv

Vista detallada, muy detallada, realmente detallada.

-c nb

Número máximo de paquetes que se capturarán.

-i Interf

Tarjeta de interfaz de red que se utilizará para la captura.

-w Archivo

Almacenamiento de los paquetes capturados en el archivo `Archivo`, para un posterior análisis.

-r Archivo

Análisis de los paquetes guardados en el archivo `Archivo`.

-A

Vista del contenido del paquete en ASCII.

-e

Vista del encabezado del nivel de enlace.

*Ejemplo*

*Captura de tráfico de red entre dos hosts que están comunicando a través del comando `nc` visto anteriormente.*

*En el sistema de captura:*

**tcpdump -nvA 'ip src 10.1.0.1 or src 10.1.0.2**

*En el servidor:*

**nc -l**
Mensaje a través de nc de Debian a CentOS
**Respuesta a través de nc de CentOS a Debian**

*En el sistema cliente:*

**nc 10.1.0.1**
**Mensaje a través de nc de Debian a CentOS**
Respuesta a través de nc de CentOS a Debian

*Interpretación (parcial) de la vista resultante en el sistema de captura:*

```
11:37:07.399325 IP (tos 0x0, ttl 64, id 7414, offset 0, flags [DF], proto TCP (6), length 60)
  10.1.0.2.35032 > 10.1.0.1.31337: Flags [S], cksum 0xf3a2 (correct), seq 927634099, win
64240, options [mss 1460,sackOK,TS val 3637026655 ecr 0,nop,wscale 7], length 0
E..<..@.@.      .
...
.....zi7J.....................
..._........
11:37:07.399434 IP (tos 0x0, ttl 64, id 0, offset 0, flags [DF], proto TCP (6), length 60)
  10.1.0.1.31337 > 10.1.0.2.35032: Flags [S.], cksum 0xb161 (correct), seq 2541485370, ack
927634100, win 28960, options [mss 1460,sackOK,TS val 3894824739 ecr 3637026655,nop,wscale
7], length 0
E..<..@.@.&.
...
...zi...{.:7J....q .a.........
.&O#..._....
11:37:07.399532 IP (tos 0x0, ttl 64, id 7415, offset 0, flags [DF], proto TCP (6), length 52)
  10.1.0.2.35032 > 10.1.0.1.31337: Flags [.], cksum 0x4f58 (correct), ack 1, win 502,
options [nop,nop,TS val 3637026655 ecr 3894824739], length 0
E..4..@.@.      .
...
.....zi7J...{.;....OX.....
..._.&O#
```

*Estos tres paquetes corresponden al inicio de una comunicación TCP, un primer paquete del cliente `10.1.0.2`, puerto de origen dinámico 35032, hacia el servidor `10.1.0.1` y puerto por defecto de `nc` 31337.*

```
11:37:20.938909 IP (tos 0x0, ttl 64, id 54036, offset 0, flags [DF], proto TCP (6),
length 52)    10.1.0.2.35032 > 10.1.0.1.31337: Flags [P.], cksum 0xf328 (correct), seq 1:36,
ack 1, win 502, options [nop,nop,TS val 3637040195 ecr 3894824739], length 35
```

```
E..W..@.@.      .
...
.....zi7J...{.;.....(.....
...C.&O#Mensaje a través de nc de Debian a CentOS
Paquete TCP de datos, del cliente hacia el servidor.



[...]

11:37:36.922238 IP (tos 0x0, ttl 64, id 54037, offset 0, flags [DF], proto TCP (6),
length 88)
   10.1.0.1.31337 > 10.1.0.2.35032: Flags [P.], cksum 0xe93c (correct), seq 1:37, ack 36,
win 227, options [nop,nop,TS val 3894854262 ecr 3637040195], length 36
E..X..@.@.S.
...
...zi...{.;7J.......<.....
.&.v...CRespuesta a través de nc de CentOS a Debian
```

*Paquete TCP de datos, del servidor al cliente.*

```
[...]
^C
12 packets captured
12 packets received by filter
0 packets dropped by kernel
```

Distintas herramientas gráficas permiten capturar el tráfico de red. Wireshark es una de las más utilizadas en Linux.