TCP/IP

1. Fundamentos

El origen de **TCP/IP** se sitúa en las investigaciones de la **DARPA** (*Defense Advanced Research Project Agency*), que empezaron en 1970 y desembocaron en **ARPANET**. En realidad, la DARPA financió a la universidad de Berkeley, que integró los protocolos básicos de TCP/IP dentro de su sistema **UNIX BSD 4**.

TCP/IP se popularizó gracias a su interfaz genérica de programación de intercambios de datos entre las máquinas de una red, las primitivas **sockets**, y la integración de protocolos de aplicación. El **IAB** (*Internet Architecture Board*) supervisa los protocolos de TCP/IP y a otros dos organismos:

- La **IRTF** (*Internet Reseach Task Force*), responsable del desarrollo de los protocolos.
- La **IETF** (*Internet Engineering Task Force*), responsable de la red Internet.

El **NIC** (*Network Information Center*) y en España el **ESNIC** distribuyen las direcciones red. Se describe el conjunto de los protocolos de TCP/IP en los documentos **RFC** (*Request For Comments*) (ver el RFC 793).

- La capa inferior es **IP** (*Internet Protocol*).
- La capa de transporte es **TCP** (*Transmission Control Protocol*) o **UDP** (*User Datagram Protocol*).
- Las capas superiores son las capas de los protocolos de aplicación; por ejemplo:
  - **NFS** (*Network File System*): comparte los archivos a distancia.
  - **DNS** (*Domain Name System*): asociación anfitrión<->IP.
  - **FTP** (*File Transfer Protocol*): transferencia de archivos.
  - **TELNET**: emulación de un terminal de tipo texto...

La versión del protocolo IP más utilizada es la V4. El protocolo IPV6 deberá remplazarla

con el tiempo. Compatible con IPV4, propone un sistema de direcciones en 128 bits (16 bytes) que permite extender las capacidades de la red, en particular en cuestión de tamaño y direccionamiento.

2. Direccionamiento

a. Clases

Es importante saber antes de la instalación a qué tipo de red se debe integrar el nuevo servidor, TCP/IP por supuesto, pero hay que reservarle una dirección IP, un hostname (nombre de máquina de red), conocer las diversas pasarelas, el nombre de dominio, la clase utilizada y la máscara de subred o netmask.

Hagamos un breve repaso de las clases de IP. Se define una dirección IP en 32 bits y se representa con cuatro nombres separados por puntos: **n1.n2.n3.n4**. Esta dirección está formada por dos partes que definen la dirección de red y el anfitrión en la red, respectivamente.

Según los casos, se distinguen cuatro o cinco clases de direcciones: A, B, C, D y E, pero sólo las tres primeras nos interesan. Esas clases ya no existen, teóricamente, desde 1992 (RFC 1338), pero se acostumbra a utilizarlas para las redes privadas.

Leyenda: N y h son bits; N es el identificador de la red h, que es el identificador de la máquina.

> **Clase A**: 0NNNNNNN hhhhhhhh hhhhhhhh hhhhhhhh o sea 1.x.x.x a
> 126.x.x.x.
> n1 está comprendido entre 1 y 126.
> 16777214 anfitriones, 127 redes.
>
> **Clase B**: 10NNNNNN NNNNNNNN hhhhhhhh hhhhhhhh o sea de 128.0.x.x a
> 191.255.x.x.
> n1 está comprendido entre 128 y 191.
> 65534 anfitriones, 16382 redes.
>
> **Clase C**: 110NNNNN NNNNNNNN NNNNNNNN hhhhhhhh o sea de 192.0.0.x a

> 223.255.255.x.
> n1 está comprendido entre 192 y 223.
> 254 anfitriones, 2097150 redes.
>
> **Clase D**: Empieza por 1110, para la multidifusión IP.
> **Clase E**: Empieza por 1111 para experimentación.

Existen direcciones de anfitriones que no pueden utilizarse. Por ejemplo, en la clase C sólo se pueden tener 254 anfitriones, mientras que el identificador de la máquina está codificado en 8 bits (por lo tanto, 256 valores). La dirección 0 representa la dirección de la red, y la dirección 255, la del **broadcast** (multidifusión).

Observe que las direcciones siguientes no se deben enrutar en Internet y se reservan a redes locales.

- 10.0.0.0 - 10.255.255.255 (10/8)
- 172.16.0.0 - 172.31.255.255 (172.16/12)
- 192.168.0.0 - 192.168.255.255 (192.168/16)

La dirección 127.0.0.1 es la dirección de loopback o bucle: representa a la propia máquina, así como la subred 127.0.0.0/8.

b. Subredes

Es posible dividir estas redes en subredes usando máscaras que permiten una división más pequeña de las direcciones. Un **netmask** es una máscara binaria que permite separar de manera inmediata la dirección de la red y de la subred, de la dirección del anfitrión en la dirección IP global. Las máscaras predefinidas son:

- **Clase A**: 255.0.0.0
- **Clase B**: 255.255.0.0
- **Clase C**: 255.255.255.0

Para comunicarse directamente entre ellos, los anfitriones deben pertenecer a una misma red o subred. Calcular una subred es bastante sencillo. Veamos un ejemplo para una red de clase C.

- Red: 192.168.1.0
- Dirección de red: 192.168.1.255
- Máscara de red: 255.255.255.0

Calcular una máscara de subred:

- Para calcular la máscara de subred, primero debe determinar cuántas máquinas quiere integrar en ella. Una red de clase C permite integrar 254 máquinas (0 y 255 están reservados). Desea crear redes que contienen 60 máquinas. Añada 2 a este valor para las direcciones reservadas (dirección de la subred y dirección de broadcast), lo que da **62**.
- Una vez determinado el número de máquinas, encuentre la potencia de dos, exacta o superior al número encontrado. 2 elevado a 6 da **64**.
- Escriba la máscara en binario, coloque todos los bits de la máscara de red de clase C a 1, y coloque a 0 los 6 primeros bits de la máscara correspondiente a la parte máquina: **11111111 11111111 11111111 11000000**
- Convierta esta máscara en decimal: **255.255.255.192**, y calcule el conjunto de las subredes posibles. Como está en una red de clase C, todavía puede hacer variar los dos últimos bits de la parte máquina:
  - 00xxxxxx: 255.255.255.0
  - 01xxxxxx: 255.255.255.64
  - 10xxxxxx: 255.255.255.128
  - 11xxxxxx: 255.255.255.192
- Al final, obtiene cuatro subredes de 62 máquinas, o sea, 248 máquinas. En efecto: obtiene 256 si añade las cuatro direcciones de broadcast y las cuatro direcciones de red.

c. Encaminamiento

La máscara de red permite determinar si una máquina destinataria está en la misma red que usted o no. Hay que indicar la ruta que deben tomar los paquetes IP para alcanzar su

destino. Si su máquina es un terminal cliente que dispone de una sola tarjeta de red y esta red tiene un único enrutador (caso clásico de una conexión hacia Internet), entonces debe crear dos rutas. La primera es la que indica qué tarjeta de red deben emplear los paquetes para acceder al resto de la red (a la subred); la segunda, qué ruta deben utilizar los paquetes para salir de la red. En general, se habla de ruta por defecto cuando sólo hay un enrutador.

- Hacia red1 -> utilizar interfaz de red izquierda.
- Hacia red2 -> utilizar interfaz de red derecha.
- Hacia otras -> utilizar interfaz de red derecha hacia enrutador1.

Ejemplo:

Red1 de clase C 192.168.1.0 en eth0, Red2 de clase B 172.16.0.0 en eth1, dirección de enrutador 192.168.1.254.

```
Red          Máscara       Interfaz    Pasarela
192.168.1.0    255.255.255.0  eth0       eth0
172.16.0.0     255.255.0.0    eth1       eth1
0.0.0.0        0.0.0.0        eth0       192.168.1.254
```

Todos los paquetes de red hacia 192.168.1.0 transitarán por eth0. Todos los paquetes cuyo destino sea 172.16.0.0 transitarán por eth1. Por defecto, los demás paquetes para las redes no especificadas transitarán por eth0 y serán tratados por la pasarela 192.168.1.254, que enrutará los paquetes.

d. IPv6

Saturación de IPv4

Las últimas direcciones IPv4 públicas se saturaron el 3 de febrero de 2011, aunque no se usen todas debido al modo de reparto por clases y subredes. Por ejemplo, las únicas subredes de clase A representan cerca de dos mil millones de direcciones IP, probablemente sin usarse todas o en manos de organismos que no usan todas. Incluso sucede lo mismo con las subredes de clase B, con mil millones de direcciones.

Un gran número de direcciones IP se reservan para usos concretos, como por ejemplo las

IP multicast, que no pueden asignarse a un equipo.

Una mala gestión inicial, la multiplicación de los terminales, especialmente de terminales móviles de tipología diversa y el gran crecimiento del número de conexiones a Internet han incrementado la demanda de forma espectacular, lo que nos ha llevado a su extinción.

Se han aplicado varios métodos para intentar retrasar la fecha de saturación, especialmente haciendo desaparecer el concepto de clases en las IP públicas o el uso de NAT, que permite mediante la traducción de direcciones ofrecer servicios accesibles al público alojados en máquinas pertenecientes a una subred privada. De este modo puede instalar docenas de servidores, sitios web u otros servicios en sus propios PC de casa, que tienen una dirección IP de una red privada, pero son accesibles desde el exterior mediante la IP pública y las funciones NAT de su router de Internet.

Las direcciones IPv4 no usadas que todavía están en manos de sus propietarios originales ahora son objeto de un negocio lucrativo.

En Europa, se ha atribuido una última subred de /22 en noviembre de 2019. Según el RIPE NCC ya no queda ninguna disponible.

Esto no significa la inmediata desaparición de IPv4: estas direcciones IP se seguirán usando durante muchos años, lo que tarden las IP no utilizadas en dejar de estarlo. Las direcciones IPv4 públicas están llamadas a desaparecer o más bien a cohabitar con otro mecanismo.

Puede seguir el estado de las asignaciones de direcciones IPv4 en el sitio siguiente:
http://ipv4.potaroo.net/

Direccionamiento IPv6

Las direcciones IPv6 reemplazarán a las direcciones IPv4. Se codifican con 128 bits y hay tantas disponibles que cada ser humano podría disponer de un trillón de direcciones sin llegar a saturarse. De este modo, NAT se convierte en una tecnología inútil y el encaminamiento se vuelve más sencillo.

La dirección se divide en 8 grupos de dos bytes escritos en hexadecimal y separados por ":", es decir 39 caracteres:

2001:0e36:2ed9:d4f0:021b:0000:0000:f81d

Pueden omitirse de uno a tres ceros no significativos:

2001:e36:2ed9:d4f0:21b:0:0:f81d

Uno o varios bloques de 16 bytes (2 bytes) nulos pueden omitirse conservando los ":" de cada lado. En el ejemplo usado anteriormente, sólo se puede aplicar una solución.

2001:e36:2ed9:df40:21b:0:0:f81d

El prefijo predeterminado para las direcciones es /64. Lo que significa que su proveedor de Internet le proporcionará una dirección con el formato 2a01:e36:2ed9:d4f0::/64, es decir de forma desarrollada:

2001:0e36:2ed9:d4f0:0000:0000:0000:0000/64

Esta subred es suya y se podrá direccionar automáticamente por su proveedor de Internet. No se puede (todavía) subdividirla en otras subredes. Podrá crear tantas direcciones IPv6 como desee en esta subred, alrededor de 2^{64} huéspedes: 18.446.744.073.709.551.616 direcciones IP. Se podrá acceder directamente a estas IP desde el exterior, sin NAT, pero evidentemente podrá proteger sus equipos con un firewall.

Uso en Linux

Este libro se basa en la configuración de las interfaces y de los servicios para un uso en IPv4. Sin embargo, el principio es el mismo que en IPv6. Linux es completamente compatible con IPv6, de forma nativa. Los comandos ifconfig, route, ping, etc. aceptan IPv6, así como los archivos de configuración de las interfaces propias de cada distribución. A continuación se muestran algunos ejemplos adaptados.

3. Casos particulares

a. NetworkManager

NetworkManager es un servicio que permite la configuración y administración dinámica de interfaces de red y de los protocolos asociados. Se acompaña en particular de un conjunto de comandos (**nmcli**, **nmtui**…) que permiten gestionar de forma centralizada la red de su equipo.

Cuando se trata de un servidor, muchos administradores prefieren usar el método clásico de gestión de la red y, por lo tanto, desactivan NetworkManager. Es cierto que la configuración de red en un servidor cambia poco en el tiempo. Sin embargo, NetworkManager ha ido remplazando los sistemas clásicos en la mayoría de las distribuciones y algunos productos lo necesitan para funcionar correctamente. En ese caso, no podrá desactivarlo.

Si desea desactivarlo en una distribución Red Hat, y volver a usar el servicio de red clásico, siga los pasos a continuación:

```
systemctl stop NetworkManager
systemctl mask NetworkManager
systemctl disable NetworkManager
systemctl enable network
systemctl unmask network
systemctl start network
```

Para Ubuntu y Debian, sustituya network por **networking**.

b. Nomenclatura de las interfaces

Parece que muchas distribuciones han modificado las reglas de nomenclatura de las interfaces de red. La nomenclatura genérica ha sido remplazada por un nombre que representa la numeración de las interfaces por el bios o por su enumeración en el bus de hardware. Así, podemos encontrar nombres de interfaces como p2p1, enp0s2… Los nombres de tipo ethX han sido de este modo reemplazados por otros.

Si esto le molesta y quiere volver a la antigua regla de nomenclatura con eth, añada las opciones `net.ifnames=0` y `biosdevname=0` al arranque del núcleo para evitar la nomenclatura a través de BIOS. Deberá, para esto, modificar los parámetros por defecto de GRUB y regenerar su configuración, como se explica en el capítulo Inicio de Linux, servicios, núcleo y periféricos.

4. Configuración

a. Caso general e histórico

La configuración básica de una interfaz de red se hace con la ayuda del comando **ifconfig**. Sin embargo, la mayoría de las distribuciones utilizan scripts de administración y archivos de configuración que simplifican mucho las cosas, ya que configuran al mismo tiempo la interfaz de red y las rutas.

Configuración de eth0 para la dirección de clase C 192.168.1.2

```
ifconfig eth0 inet 192.168.1.2 netmask 255.255.255.0
```

o

```
ifconfig eth0 192.168.1.2
```

Activación de la interfaz de red eth0:

```
ifconfig eth0 up
```

Parada de la interfaz de red eth0:

```
ifconfig eth0 down
```

Visualización de información de eth0:

```
# ifconfig eth0
eth0      Vínculo encap:Ethernet  HWaddr 00:XX:XX:XX:XX:XX
          inet adr:192.168.1.60  Bcast:192.168.1.255  Máscara:255.255.255.0
          adr inet6: fe80::21b:fcff:fec9:f81d/64 Scope:Vínculo
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:16522 errors:0 dropped:0 overruns:0 frame:0
```

```
TX packets:13631 errors:0 dropped:0 overruns:0 carrier:2
collisions:0 lg file transmission:1000
RX bytes:17732221 (16.9 MB)  TX bytes:1648879 (1.5 MB)
```

Observe que la dirección IPv6 aparece en el resultado en la línea inet6.

Configuración de una dirección IPv6

```
# ifconfig eth0 inet6 add fe80::21b:fcff:fec9:f81d/64
```

Visualización de todas las interfaces de red activadas:

```
ifconfig
```

Visualización de todas las interfaces de red, activadas o no:

```
ifconfig -a
```

b. Caso de las distribuciones de tipo Red Hat

Red Hat propuso, antes de la llegada de NetworkManager, herramientas para configurar la red básica sin pasar por la manipulación de los archivos de configuración. Estas herramientas desaparecieron después de la versión 7. Como información, encontrará más adelante los comandos para las versiones anteriores.

El comando **netconfig** funciona en modo texto y permite la configuración básica de TCP/IP (IP estática o dinámica, router, nombre de host, servidor DNS).

```
# netconfig --device eth0
```

El comando gráfico **system-config-network** inicia una interfaz muy completa.

La distribución SuSE (o SLES) toma el mismo principio expuesto anteriormente, pero la sintaxis y la ubicación de los archivos pueden variar. Debe consultar la documentación de su distribución para más detalles.
Lo mejor sigue siendo la utilización de los scripts **ifup** e **ifdown**. Se basan en los archivos presentes en `/etc/sysconfig/network-scripts/` . Estos archivos de configuración de interfaz se llaman ifcfg-xxx, donde xxx es el nombre de la interfaz de red, como eth0:

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
NETWORK=192.168.1.0
BROADCAST=192.168.1.255
ONBOOT=yes
BOOTPROTO=static
```

Los parámetros hablan por sí mismos. Los valores **NETWORK** y **BROADCAST** son opcionales si se notifican **IPADDR** y **NETMASK** (en este caso, el cálculo es automático) o si se utiliza **DHCP**. **BOOTPROTO** indica cómo montar la interfaz, o bien **static**, o bien **dhcp**. Se puede utilizar el valor **bootp**. En el caso de DHCP, el archivo se puede parecer a esto:

```
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
```

En el caso de una configuración estática, **IPADDR** y **NETMASK** son obligatorios:

```
DEVICE=eth0
IPADDR=192.168.1.2
NETMASK=255.255.255.0
ONBOOT=yes
BOOTPROTO=static
```

ONBOOT determina si se debe activar automáticamente la interfaz en

el arranque de la máquina.
Para IPv6, debe utilizar las siguientes variables:

```
IPV6INIT=yes
IPV6ADDR=<IPv6-IP-Address>
IPV6_DEFAULTGW=<IPv6-IP-Gateway-Address>
```

IPV6INIT indica si IPv6 está activo o no en esta interfaz. Las dos variables siguientes indican la dirección IP y su puerta de enlace.
Una vez notificado correctamente el archivo, se utilizan los comandos **ifup/ifdown**:
Activación de la interfaz eth0:

```
ifup eth0
```

Parada de la interfaz eth0:

```
ifdown eth0
```

Parámetros generales

El archivo `/etc/sysconfig/network` contiene los parámetros generales de la red.

```
NETWORKING=yes
HOSTNAME=puesto1.mired.org # nombre completo
GATEWAY=0.0.0.0 # pasarela por defecto
NISDOMAIN= # nombre del dominio NIS
NETWORKING_IPV6=yes
```

- **NETWORKING**: activación o no de la red.
- **HOSTNAME**: nombre de dominio completo FQDN.
- **GATEWAY**: dirección IP de la pasarela.
- **GATEWAYDEV**: interfaz de red que permite acceder a la pasarela.
- **NISDOMAIN**: en caso de un dominio NIS.
- **NETWORKING_IPV6**: activación o no del soporte de IPv6.

c. Máquinas de tipo Debian y Ubuntu

El archivo de configuración de las interfaces de redes en Debian (y Ubuntu, antes de la

versión 17.10) se sitúa en **`/etc/network/interfaces`** . No tiene el mismo formato que en Red Hat.

```
# cat interfaces
auto lo eth0 eth1
iface lo inet loopback

iface eth0 inet static
    address 192.161.1.60
    netmask 255.255.255.0
    broadcast 192.168.1.255
    gateway 192.168.1.1

iface eth1 inet dhcp
```

Este ejemplo muestra tres tipos de interfaces:

- la interfaz lo de loopback,
- la interfaz eth1 en dhcp, que no necesita una configuración más avanzada,
- la interfaz eth0 configurada de manera estática.

La sintaxis general de una declaración es la siguiente:

```
interfaz nombre tipo modo
```

Con una configuración estática, especifique los diferentes parámetros con las palabras claves siguientes:

- **address**: la dirección IP.
- **netmask**: la máscara de subred.
- **broadcast**: la dirección de broadcast.
- **gateway**: la pasarela por defecto.

La línea **auto** indica las interfaces que se activarán automáticamente en el arranque.

Para IPv6 la configuración es idéntica: basta con reemplazar inet con inet6.

A partir de 2012 podemos emplear una directiva **include** que permite incluir un archivo externo a la configuración.

El archivo `/etc/hostname` contiene el nombre de la máquina:

```
# cat /etc/hostname
slyserver
```

d. Encaminamiento

Con la utilización de los archivos y comandos anteriores, no hace falta crear un encaminamiento específico, puesto que la pasarela por defecto está ya presente (vea Parámetros generales) e **ifup** implementa automáticamente las rutas para las interfaces. Sin embargo, se puede utilizar el comando **route**.

Visualiza las rutas actuales:

```
route
netstat -nr
```

En el ejemplo siguiente, las interfaces vmnet1 y vmnet8 son interfaces virtuales procedentes de la configuración de red de VMWare.

```
# netstat -rn
Tabla de encaminamiento IP del núcleo
Destino      Pasarela     Genmask       Indic   MSS Ventana irtt   Iface
192.168.211.0  0.0.0.0     255.255.255.0  U        0 0        0  vmnet8
192.168.1.0    0.0.0.0     255.255.255.0  U        0 0        0    eth0
172.16.248.0   0.0.0.0     255.255.255.0  U        0 0        0  vmnet1
169.254.0.0    0.0.0.0     255.255.0.0    U        0 0        0    eth0
127.0.0.0      0.0.0.0     255.0.0.0      U        0 0        0      lo
0.0.0.0        192.168.1.1  0.0.0.0       UG       0 0        0    eth0
```

Creación de la entrada loopback:

```
route add -net 127.0.0.0
```

Crea la ruta hacia la red 192.168.1.0, que pasa por eth0. Se puede omitir netmask.

```
route add -net 192.168.1.0 netmask 255.255.255.0 eth0
```

Crea la pasarela por defecto hacia el enrutador:

```
route add default gw 192.168.1.254
```

Suprime la ruta hacia la red 172.16.0.0:

```
route del -net 172.16.0.0 eth0
```

La gestión de rutas IPv6 es idéntica, pero indicando el tipo de dirección usado en el comando con la opción -A:

```
route -A inet6 add <ipv6> gw <ipv6>
```

e. iproute2

Los comandos **ifconfig** y **route** no son los únicos que permiten la asignación, la modificación o la eliminación de las direcciones IP y las rutas. Los comandos **iproute2** hacen eso y mucho más, en particular aquellos que ofrecen opciones de más bajo nivel y permiten operaciones más complejas. El comando principal se llama **ip**. Solo abordaremos las nociones básicas.

Los tres objetos, entre otros, que ip puede manipular son:

- **link**, gestiona los atributos de la interfaz de red.
- **addr**, gestiona la dirección IP de la interfaz de red.
- **route**, gestiona las rutas a las redes y hosts.

Para mostrar el estado de las interfaces de red a nivel de hardware (o virtual), utilice **link show**, y si es preciso la interfaz. Si aparece **NO CARRIER**, es que la interfaz no recibe nada. Si se muestra **state DOWN**, la interfaz no está conectada:

```
$ ip link show eth0
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN mode DEFAULT qlen 1000
  link/ether 08:00:27:6b:d5:77 brd ff:ff:ff:ff:ff:ff
```

Para mostrar la configuración IP de una interfaz, utilice **addr show**; obtendrá la información de IPv4 e IPv6:

```
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000
  link/ether 08:00:27:a7:26:48 brd ff:ff:ff:ff:ff:ff
  inet 192.168.99.13/24 brd 192.168.99.255 scope global eth1
    valid_lft forever preferred_lft forever
  inet6 fe80::a00:27ff:fea7:2648/64 scope link
    valid_lft forever preferred_lft forever
```

Para cambiar la IP, utilice **addr add** según sigue;

```
# ip addr add 192.168.0.100/24 dev eth0
# ip addr show dev eth0
2: eth0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN qlen 1000
  link/ether 08:00:27:6b:d5:77 brd ff:ff:ff:ff:ff:ff
  inet 192.168.0.100/24 scope global eth0
    valid_lft forever preferred_lft forever
```

Para listar las rutas, utilice **route show**:

```
# ip route show
default via 192.168.0.254 dev eth0
169.254.0.0/16 dev eth0  scope link  metric 1002
169.254.0.0/16 dev eth1  scope link  metric 1003
```

```
192.168.0.0/24 dev eth0  proto kernel  scope link  src 192.168.0.19
192.168.99.0/24 dev eth1  proto kernel  scope link  src 192.168.99.13
```

Por último, para modificar la puerta de enlace por defecto, utilice **route change**:

```
# ip route change default via 192.168.99.1
# ip route show
default via 192.168.99.1 dev eth1
```

f. Network Manager

Network Manager se ha impuesto en un gran número de distribuciones. Se trata de un gestor de configuración de redes dinámico, constituido por un servicio de comandos (cli) para poder comunicarse con él. La presencia de ese servicio, la compatibilidad con DBUS y la presencia también de una API han contribuido a simplificar la administración de las redes, especialmente en los entornos gráficos, o en la configuración de las redes Wi-Fi. Network Manager puede administrar todo tipo de periféricos de red: Ethernet, Wi-Fi, 3/4G, módems, aunque también puede gestionar las conexiones, las rutas, los servidores DNS, etc.

Para saber si Network Manager está siendo usado, compruebe la presencia del servicio asociado:

```
# systemctl status NetworkManager
  NetworkManager.service - Network Manager
    Loaded: loaded (/lib/systemd/system/NetworkManager.service; enabled; vendo>
    Active: active (running) since Sat 2021-05-15 04:07:38 CEST; 7min ago
```

El comando **nmcli** gestiona integralmente la configuración. Ejecutado sin argumentos, le mostrará el detalle de la configuración actual:

```
# nmcli
enp0s3: conectado to enp0s3
```

```
        "Intel 82540EM"
        ethernet (e1000), 08:00:27:5A:AF:88, hw, mtu 1500
        ip4 par défaut
        inet4 10.0.2.15/24
        route4 0.0.0.0/0
        route4 10.0.2.0/24
        inet6 fe80::df5b:46ad:b91b:6e17/64
        route6 fe80::/64
        route6 ff00::/8
...

DNS configuration:
        servers: 208.67.222.222 208.67.220.220
        interface: enp0s3
```

Para conocer los detalles de una interfaz en particular, use **device show**:

```
# nmcli device show enp0s3
GENERAL.DEVICE:                         enp0s3
GENERAL.TYPE:                           ethernet
GENERAL.HWADDR:                         08:00:27:5A:AF:88
GENERAL.MTU:                            1500
GENERAL.STATE:                          100 (conectado)
GENERAL.CONNECTION:                     enp0s3
GENERAL.CON-PATH:                       /org/freedesktop/NetworkManager/ActiveConnection/1
WIRED-PROPERTIES.CARRIER:               activado
IP4.ADDRESS[1]:                         10.0.2.15/24
IP4.GATEWAY:                            10.0.2.2
IP4.ROUTE[1]:                           dst = 0.0.0.0/0, nh = 10.0.2.2, mt = 100
IP4.ROUTE[2]:                           dst = 10.0.2.0/24, nh = 0.0.0.0, mt = 100
IP4.DNS[1]:                             208.67.222.222
IP4.DNS[2]:                             208.67.220.220
IP6.ADDRESS[1]:                         fe80::df5b:46ad:b91b:6e17/64
IP6.GATEWAY:                            --
IP6.ROUTE[1]:                           dst = fe80::/64, nh = ::, mt = 100
IP6.ROUTE[2]:                           dst = ff00::/8, nh = ::, mt = 256, table=255
```

El argumento `connection show` le muestra el estado actual de las conexiones:

```
# nmcli connection show
NAME      UUID                                  TYPE      DEVICE
enp0s3    7df028a8-050f-3519-98d5-11fd8e0b3495  ethernet  enp0s3
```

Para iniciar o parar una interfaz de red:

```
# nmcli con up enp0s3
Conexión activada con éxito (ruta activa D-Bus: /org/freedesktop/NetworkManager/
ActiveConnection/5)
# nmcli con down enp0s3
La conexión « enp0s3 » se desactivó correctamente (ruta activa D-Bus: /org/freedesktop/
NetworkManager/ActiveConnection/7)
```

Para comprobar el estado de las interfaces, donde se puede ver que la interfaz enp0s3 está desconectada:

```
# nmcli dev status
DEVICE     TYPE     STATE        CONNECTION
enp0s8     ethernet  conectado     Conexión cableada 1
enp0s3     ethernet  desconectado  --
```

Para configurar una interfaz en modo DHCP (modo por defecto), la sintaxis es la siguiente:

```
# mcli con add type ethernet con-name "Red virtual" ifname enp0s3

Conexión «Red virtual» (c828a07e-4841-472b-a551-7c415042f633) añadida con éxito.
```

Una dirección IP estática también se configura de manera simple:

```
# nmcli con add type ethernet con-name "Red virtual" ifname enp0s3 ip4
10.0.2.15/24 gw4 10.0.2.2
```

Si el servidor DHCP configura generalmente los servidores DNS por defecto, esto no es lo que ocurre en el caso de una configuración estática. Añada, a la conexión anterior, los servidores DNS de Google:

```
# nmcli con mod "Red virtual" ipv4.dns "8.8.8.8 8.8.4.4"
```

Active la conexión como se vio anteriormente con el comando `nmcli con up`.

Network Manager administra también las rutas estáticas. Añada una ruta en la interfaz enp0s3:

```
# nmcli connection modify enp0s3 +ipv4.routes "192.168.122.0/24 10.10.10.1"
```

g. netplan

Adoptado por Ubuntu, **netplan** es un sistema de gestión de configuración de redes que hace pantalla a **networkd** (systemd) o a Network Manager. Ha sido desarrollado por Canonical para Ubuntu, pero se puede utilizar en cualquier distribución.

El principio es simple:

- La configuración de red está descrita en un archivo de formato YAML.
- Netplan configura el backend (Network Manager, networkd) en función de esta configuración.

La configuración se encuentra en el directorio **`/etc/netplan/`**, en forma de archivos numerados, por ejemplo 50-cloud-init.yaml, que precisan el orden de tratamiento.

He aquí el ejemplo más simple de contenido, que será utilizado por Network Manager para configurar todas las interfaces en DHCP:

```
network:
  version: 2
  renderer: NetworkManager
```

El ejemplo siguiente muestra una configuración automática en DHCP pero usando **networkd** (systemd) :

```
network:
  ethernets:
    enp0s3:
      dhcp4: true
    enp0s8:
      dhcp4: true
  version: 2
```

El comando **networkctl** controla **networkd** (systemd) y le proporciona la información:

```
$ networkctl
IDX LINK         TYPE            OPERATIONAL     SETUP
 1 lo            loopback        carrier         unmanaged
 2 enp0s3        ether           routable        configured
 3 enp0s8        ether           routable        configured
```

La sintaxis no depende del backend usado. Si quisiera atribuir una dirección IP estática a enp0s3 usando los servidores DNS de Google, con una ruta adicional, he aquí el contenido del archivo:

```
network:
 version: 2
 renderer: networkd # o NetworkManager
 ethernets:
  enp3s0:
    addresses:
     - 10.0.2.15/24
    gateway4: 10.0.2.2
    nameservers:
      addresses: [8.8.8.8, 8.8.4.4]
    routes:
     - to: 192.168.0.0/24
       via: 10.0.2.4
       metric: 100
```

Ya solo quedaría aplicar la configuración:

```
# netplan apply
```

h. Los puertos

Los protocolos TCP y UDP trabajan con la ayuda de los puertos. Un puerto es un identificador numérico que permite a los interlocutores distinguirse y dialogar de forma conjunta. En la práctica, un servicio escucha en un puerto y responde a los clientes en su propio puerto. El caso más conocido es el uso de un servicio que hospeda sitios web cuyo puerto por defecto es el 80. Los clientes (navegadores de Internet) intentarán siempre conectarse a este si el protocolo usado es HTTP.

Un puerto se define con 16 bits. Los puertos desde el 0 al 1023 están reservados a los servicios, se deben ejecutar como root y se llaman «*Well Known Services*». Más allá de estos valores, el uso es libre; sin embargo, observe: cuando un cliente se conecta a un servicio, también utiliza un puerto, y dos puertos en ejecución no pueden escuchar en el mismo puerto.

Un puerto UDP y un puerto TCP son dos puertos distintos. De la misma forma, un puerto puede escuchar en todas las direcciones IP de un servidor o una en particular. De esta forma, si un servidor tiene dos direcciones IP, un primer servicio puede escuchar en el puerto 80 de la primera dirección IP, y otro puede escuchar en el puerto 80 de la segunda dirección IP: los puertos no son los mismos.

Aquí tenemos una lista de puertos destacables:

Puerto	Servicio
20	Datos FTP
21	Protocolo FTP
22	ssh

23	telnet
25	smtp (envío de mail)
53	DNS
80	HTTP
110	pop3 (recepción de mail)
123	NTP
139	netbios (servicio windows)
143	imap
161	snmp
162	traps snmp
389	ldap (no seguro)
443	https
465	smtps (smtp con SSL)
514	syslog
636	ldaps (ldap con SSL)

993	imaps (imap con SSL)
995	pop3s (pop3 con SSL)

5. Herramientas de red

a. Ping

El comando **ping** es un comando central, incluso ineludible. Para saber si una máquina está accesible o no, lo primero que se suele hacer es intentar “hacer ping” (con la condición de que la configuración del firewall autorice las peticiones ICMP). El comando **ping6** hace lo mismo para una dirección Ipv6. Si se omiten los paquetes, es señal de un error en la red. Si se omiten todos los paquetes, será que la conexión el equipo remoto está cortada, o bien el protocolo ICMP está restringido por un firewall.

Ping emite un "eco" de red, como un sónar, y espera una respuesta, el retorno del eco. Para ello, utiliza el protocolo ICMP. Interrumpa el comando **ping** con [Ctrl] **C**.

```
$ ping www.kde.org
PING www.kde.org (62.70.27.118) 56(84) bytes of data.
64 bytes from 62.70.27.118: icmp_seq=1 ttl=57 time=10.5 ms
64 bytes from 62.70.27.118: icmp_seq=2 ttl=57 time=11.3 ms
64 bytes from 62.70.27.118: icmp_seq=3 ttl=57 time=10.4 ms
64 bytes from 62.70.27.118: icmp_seq=4 ttl=57 time=11.5 ms
...
```

Tres parámetros deben llamarle la atención:

- `-c` permite especificar el número de ecos que se deben emitir.
- `-b` permite emitir un eco en una dirección de broadcast.
- `-I` permite especificar la interfaz de red.

En el primer caso, el parámetro puede ser útil en un script para probar que un servidor responde:

```
# ping -c 1 10.9.238.170 >/dev/null 2>&1 && echo "El servidor contesta"
El servidor contesta
```

En el segundo caso, todas las direcciones de la subred correspondientes a la dirección de broadcast deben responder.

```
# ping -b 192.168.1.255
WARNING: pinging broadcast address
PING 192.168.1.255 (192.168.1.255) 56(84) bytes of data.
64 bytes from 192.168.1.10: icmp_seq=1 ttl=64 time=0.232 ms
64 bytes from 192.168.1.60: icmp_seq=1 ttl=64 time=0.240 ms
64 bytes from 192.168.1.130: icmp_seq=1 ttl=255 time=0.285 ms
64 bytes from 192.168.1.139: icmp_seq=1 ttl=255 time=0.292 ms
...
```

En el último caso, puede especificar una tarjeta de salida. Esta opción es muy útil para comprobar una resolución DNS o una ruta.

```
# ping -I eth0 192.168.1.60
PING 192.168.1.60 (192.168.1.60) from 192.168.1.10:eth0: 56(84) bytes
of data.
64 bytes from 192.168.1.60: icmp_seq=1 ttl=62 time=0.478 ms
64 bytes from 192.168.1.60: icmp_seq=2 ttl=62 time=0.408 ms
...
```

b. Traceroute

Cuando intenta acceder a un anfitrión remoto desde su máquina, los paquetes IP pasan a menudo por muchas rutas, a veces diferentes según el punto de partida y de destino, los cuellos de botella, etc. El trayecto pasa por numerosas pasarelas (gateways), que dependen de las rutas por defecto o predefinidas de cada una de ellas.

El comando **traceroute** permite visualizar cada uno de los puntos de paso de sus paquetes

IP a su destino en un anfitrión dado. En el ejemplo siguiente, el anfitrión ubicado en la región parisina en la red del proveedor Free intenta determinar la ruta recorrida para ir al servidor www.kde.org. Se enmascarará la dirección IP fuente (fuera de la red local). El comando **traceroute6** es idéntico para IPv6.

```
$ traceroute www.kde.org
traceroute to www.kde.org (62.70.27.118), 30 hops max, 40 byte packets
 1 DD-WRT (192.168.1.1)  0.558 ms   0.533 ms   0.585 ms
 2 82.xxx.yyy.zzz (82.xxx.yyy.zzz)  6.339 ms   6.404 ms   6.901 ms
 3 * * *
 4 * * *
 5 212.73.205.5 (212.73.205.5)  39.267 ms   35.499 ms   31.736 ms
 6 ae-12-55.car2.Paris1.Level3.net (4.68.109.144)  6.485 ms ae-22-
52.car2.Paris1.Level3.net (4.68.109.48)  6.401 ms   6.338 ms
 7 UUnet-Level3.Level3.net (212.73.240.206)  6.113 ms   6.152 ms
5.866 ms
 8 so-3-2-0.TL2.PAR2.ALTER.NET (146.188.8.121)  6.107 ms   6.410 ms
6.365 ms
 9 so-2-2-0.TL2.STK2.ALTER.NET (146.188.7.33)  87.323 ms   86.840 ms
87.010 ms
10 so-7-1-0.XR2.OSL2.ALTER.NET (146.188.15.62)  96.491 ms   97.148 ms
96.488 ms
11 ge-0-1-0.GW6.OSL2.ALTER.NET (146.188.3.242)  95.972 ms   95.934 ms
96.108 ms
12 213.203.63.74 (213.203.63.74)  95.320 ms   94.321 ms   96.188 ms
13 leeloo.troll.no (62.70.27.10)  94.064 ms   94.052 ms   92.374 ms
14 jamaica.kde.org (62.70.27.118)  97.064 ms   96.182 ms   97.853 ms
```

c. tracepath

Los comandos **tracepath** y **tracepath6** son casi equivalentes a traceroute con la excepción de que ellos no emplean el mismo método. Traceroute es muy preciso y trabaja con tramas sin procesar (raw), y requiere de permisos para que algunas opciones funcionen mientras que tracepath trabaja con sockets y no necesita de permisos particulares. Se emplea igual que traceroute.

```
$ tracepath -n www.kde.org
```

```
 1?: [LOCALHOST]                      pmtu 1500
 1:  192.168.0.254                     0.930ms
 1:  192.168.0.254                     0.927ms
... (información privada)
 5:  78.254.255.157                    7.643ms
 6:  78.254.249.1                     13.584ms
 7:  194.149.162.22                    8.794ms
 8:  212.73.243.5                      8.284ms
 9:  4.69.166.69                      15.213ms asymm 13
10:  212.113.8.154                    14.798ms asymm 13
11:  91.189.88.10                     67.234ms asymm 16
12:  91.189.93.5                      16.868ms reached
   Resume: pmtu 1500 hops 12 back 17
```

d. Whois

¿Sabía que puede obtener toda la información que desee sobre un dominio (pepito.es) usando el comando **whois**? Por ejemplo, para obtener toda la información sobre el dominio kde.org:

```
> whois kde.org
...
Domain Name:KDE.ORG
Domain ID: D1479623-LROR
Creation Date: 1996-12-14T05:00:00Z
Updated Date: 2013-12-14T12:11:24Z
Registry Expiry Date: 2014-12-13T05:00:00Z
Sponsoring Registrar:GoDaddy.com, LLC (R91-LROR)
Sponsoring Registrar IANA ID: 146
WHOIS Server:
Referral URL:
Domain Status: clientDeleteProhibited
Domain Status: clientRenewProhibited
Domain Status: clientTransferProhibited
Domain Status: clientUpdateProhibited
Registrant ID:CR88351429
Registrant Name:K Desktop Environment EV
```

```
Registrant Organization:KDE e.V.
Registrant Street: Schoenhauser Allee 6
Registrant City:Berlin
Registrant State/Province:
Registrant Postal Code:10119
Registrant Country:DE
Registrant Phone:+49.30202373050
Registrant Phone Ext:
Registrant Fax:
Registrant Fax Ext:
Registrant Email:domain@kde.org
Admin ID:CR88351433
...
```

e. Netstat

El comando **netstat** permite obtener gran cantidad de información sobre la red y los protocolos.

El parámetro `-i` permite obtener el estado de las tarjetas de redes para determinar una posible avería o un problema de cable:

```
# netstat -i
Tabla de interfaces del núcleo
Iface  MTU Met   RX-OK RX-ERR RX-DRP RX-OVR   TX-OK TX-ERR TX-DRP TX-OVR Flg
eth0  1500   0 2332007      0      0      0  677842      0      0  0 BMRU
lo   16436   0    1109      0      0      0    1109      0      0  0 LRU
```

Si añade el parámetro `-e`, obtiene el mismo resultado que con ifconfig `-a`.

```
# netstat -ei
Tabla de interfaces del núcleo
eth0   Vínculo encap:Ethernet  HWaddr 00:XX:D3:XX:AA:XX
       inet adr:12.168.1.60  Bcast:192.168.1.255  Máscara:255.255.255.0
       UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
```

```
RX packets:2335314 errors:0 dropped:0 overruns:0 frame:0
      TX packets:678095 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 lg file transmission:1000
      RX bytes:1055212145 (1006.3 MB)  TX bytes:61264196 (58.4 MB)
      Interruption:20 Dirección basica:0x8c00

lo    Vínculo encap:Bucle local
      inet adr:127.0.0.1  Máscara:255.0.0.0
      UP LOOPBACK RUNNING  MTU:16436  Metric:1
      RX packets:1109 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1109 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 lg file transmission:0
      RX bytes:60423 (59.0 Kb)  TX bytes:60423 (59.0 KB)
```

El parámetro `-r` permite obtener las tablas de encaminamiento como ruta. Añada el parámetro `-n` para indicar las IP en lugar de los nombres.

```
# netstat -rn
Tabla de encaminamiento IP del núcleo
Destino       Pasarela     Genmask       Indic   MSS Ventana irtt   Iface
192.168.211.0   0.0.0.0      255.255.255.0   U        0 0          0   vmnet8
192.168.1.0     0.0.0.0      255.255.255.0   U        0 0          0   eth0
172.16.248.0    0.0.0.0      255.255.255.0   U        0 0          0   vmnet1
169.254.0.0     0.0.0.0      255.255.0.0     U        0 0          0   eth0
127.0.0.0       0.0.0.0      255.0.0.0       U        0 0          0   lo
0.0.0.0         192.168.1.1   0.0.0.0       UG       0 0          0   eth0
```

El parámetro `-a` permite visualizar todas las conexiones, para todos los protocolos, incluido los puertos en escucha de la máquina. La salida es demasiado larga para reproducirla en estas páginas.

```
# netstat -a | wc -l
495
```

El parámetro `-A` permite especificar el protocolo que es preciso consultar: inet, inet6

(ipv6), unix, ipx, ax25, netrom y ddp.

```
# netstat -a -A inet
Conexiones Internet activas (servidores y establecidas)
Proto Recv-Q Send-Q Dirección local       Dirección remota       Estado
tcp        0      0 localhost:716           *:*                    LISTEN
tcp        0      0 *:sunrpc                *:*                    LISTEN
tcp        0      0 localhost:ipp           *:*                    LISTEN
tcp        0      0 localhost:smtp          *:*                    LISTEN
tcp        0      0 localhost:hpssd         *:*                    LISTEN
tcp        0      0 slyserver:41851         imap.tele2.es:imap     ESTABLISHED
tcp        0      0 slyserver:41850         imap.tele2.es:imap     ESTABLISHED
tcp        0      0 slyserver:54220         by1msg4176111.gate:msnp ESTABLISHED
tcp        0      0 slyserver:34267         by2msg2105007.phx.:msnp ESTABLISHED
tcp        0      0 slyserver:47990         by1msg4082314.phx.:msnp ESTABLISHED
udp        0      0 *:filenet-tms           *:*
udp        0      0 *:mdns                  *:*
udp        0      0 *:sunrpc                *:*
udp        0      0 *:ipp                   *:*
udp        0      0 172.16.248.1:ntp        *:*
udp        0      0 192.168.211.1:ntp       *:*
udp        0      0 slyserver:ntp           *:*
udp        0      0 localhost:ntp           *:*
udp        0      0 *:ntp                   *:*
raw        0      0 *:icmp                  *:*                    7
```

Por último, el parámetro `-p` permite indicar, cuando es posible, el PID y el nombre del proceso.

```
# netstat -A inet -p
Conexiones a Internet activas (sin servidores)
Proto Recv-Q Send-Q Dirección local       Dirección remota
Estado      PID/Program name
...
tcp        0      0 slyserver:54220         by1msg4176111.gate:msnp
ESTABLISHED   4041/kopete
tcp        0      0 slyserver:34267         by2msg2105007.phx.:msnp
ESTABLISHED   4041/kopete
```

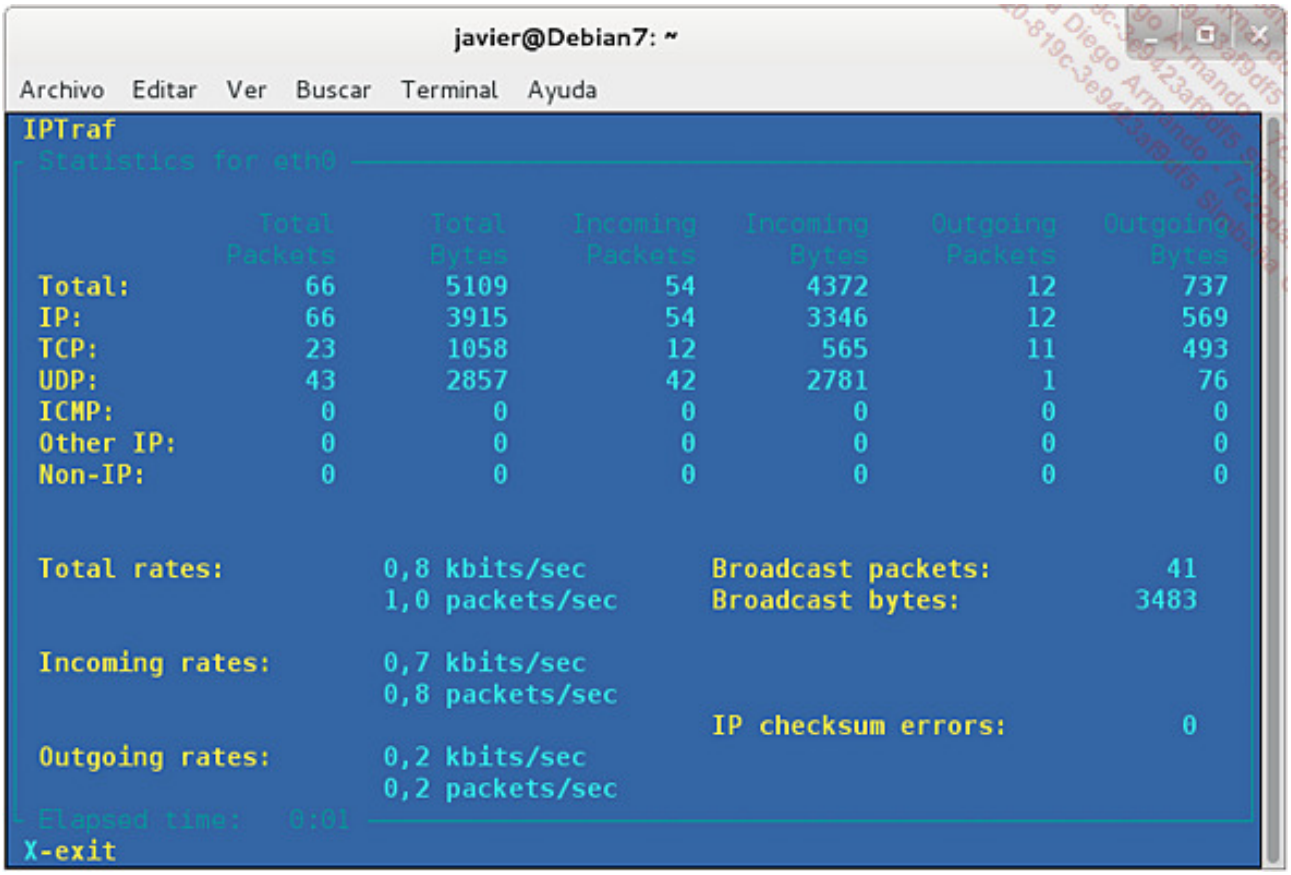
```
tcp        0      0 slyserver:47990         by1msg4082314.phx.:msnp
ESTABLISHED  4041/kopete
```

f. IPTraf

El comando **iptraf**, o **iptraf-ng**, permite visualizar en tiempo real la actividad de la red mediante una herramienta de texto, opcionalmente interactiva (línea de comandos). Los menús son claros. Puede moverse por ellos con las teclas de dirección y los diferentes atajos especificados.

La captura siguiente muestra la visualización detallada de las estadísticas de la tarjeta `eth0`. A esta pantalla se accede desde la línea de comandos con:

```
# iptraf -d eth0
```



IPTraff analiza el tráfico de eth0

6. Archivos generales

a. /etc/resolv.conf

Se utiliza el archivo **`/etc/resolv.conf`** para indicar al sistema qué servidores de nombres y qué dominios hay que consultar para resolver las peticiones DNS clientes. Estas API, al igual que las API estándares de Linux, están incluidas en la librería (no hace falta añadir herramientas adicionales). Esta librería se llama **`resolver`** .

Al configurar DHCP, en principio se actualiza automáticamente este archivo y no se debería modificar, salvo que haya prohibido la configuración DNS en el cliente.

```
$ cat /etc/resolv.conf
domain midominio.org
search midominio.org
nameserver 192.168.1.1
nameserver 192.168.1.2
```

- **domain**: nombre del dominio local. Las peticiones se suelen reducir a unos atajos relativos al dominio local. Si no está creado, el nombre del dominio se debe determinar a partir del nombre completo del anfitrión: corresponde a la parte ubicada después del primer «.».
- **search**: lista de los dominios de búsqueda. Por defecto, durante la utilización de atajos (nombres de anfitriones cortos) el resolver inicia una búsqueda sobre el dominio definido por la línea domain, pero se puede especificar aquí una lista de dominios separados por espacios o comas.
- **nameserver**: dirección IP del servidor de nombres (el servidor DNS). Se puede

colocar un máximo de tres. El resolver intenta utilizar el primero. Si fracasa (timeout), pasa al segundo, y así sucesivamente.

- **opciones**: se pueden especificar opciones. Por ejemplo, **timeout:n**, donde n (en segundos) indica el tiempo de espera de respuesta de un servidor de nombres antes de pasar al siguiente.

Observe que systemd puede mantener este archivo con la ayuda del servicio **systemd-resolved**. En ese caso, /etc/resolv.conf es un enlace simbólico hacia **/run/systemd/resolve/stub-resolv.conf**.
Systemd actuará como un servidor DNS local, escuchando en la dirección 127.0.0.53, en el puerto 53. Su configuración se encuentra en **/etc/systemd/resolved.conf**, pero como systemd controla también la red, se puede configurar cada interfaz en **/etc/systemd/network/*.** En el caso de una configuración con un servidor DHCP con servidores atribuidos automáticamente, esos archivos se generan vacíos. Para obtener el estado actual, use el comando **resolvectl** o **systemd-resolv --status**:

```
$ resolvectl status enp0s8
Link 3 (enp0s8)
    Current Scopes: DNS
DefaultRoute setting: yes
     LLMNR setting: yes
MulticastDNS setting: no
  DNSOverTLS setting: no
     DNSSEC setting: no
   DNSSEC supported: no
 Current DNS Server: 208.67.220.220
        DNS Servers: fd0f:ee:b0::1
                     208.67.222.222
                     208.67.220.220
         DNS Domain: ~.
```

b. /etc/hosts y /etc/networks

Sin siquiera utilizar un servidor de nombres, se puede establecer una correspondencia entre las direcciones IP y los nombres de las máquinas dentro del archivo `/etc/hosts` .

```
192.168.1.1  server1 www1 ftp
```

```
192.168.1.11 puesto1
192.168.1.12 puesto2
```

Puede hacer lo mismo para nombrar las redes (lo que puede ser útil para los **tcp_wrappers** o el comando **route**) en el archivo `/etc/networks` .

```
loopnet  127.0.0.0
localnet 192.168.1.0
```

c. /etc/nsswitch.conf

El archivo `/etc/nsswitch.conf` permite determinar el orden en el cual el resolver (u otros servicios) recupera su información. Las dos líneas en negrita en el ejemplo indican que, durante una petición de resolución de nombre (o red), los archivos son prioritarios. Primero se lee el archivo `/etc/hosts` ; luego, si el resolver no encuentra la información, busca mediante una resolución DNS.

```
passwd: compat
group:  compat
hosts:      files dns
networks:   files dns

services:   files
protocols:  files
rpc:        files
ethers:     files
netmasks:   files
netgroup:   files nis
publickey:  files

bootparams: files
automount:  files nis
aliases:    files
```

Puede ocurrir que determinados programas no utilicen el resolver sino directamente el DNS o el archivo **`etc/hosts`** , o inviertan el orden establecido en **`/etc/nsswitch.conf`** . En este caso, no es posible prever (ni predecir) el funcionamiento correcto de este tipo de programas...

d. /etc/services

El archivo **`/etc/services`** contiene la lista de los servicios de red conocidos de Unix, así como los puertos y protocolos asociados. Muchos servicios (entre los cuales se halla **xinetd**) y subsistemas, como el firewall de Linux, lo utilizan.

Este archivo es indicativo: se trata de un archivo de descripción y definición: no todos los servicios de este archivo se ejecutan obligatoriamente en su máquina (por fortuna... ¡visto el número!). Y se puede configurar un servicio para escuchar otro puerto. En cambio, se aconseja ponerlo al final cuando se añade un servicio que no está presente en este archivo.

```
tcpmux          1/tcp    # TCP Port Service Multiplexer
tcpmux          1/udp    # TCP Port Service Multiplexer
compressnet     2/tcp    # Management Utility
compressnet     2/udp    # Management Utility
compressnet     3/tcp    # Compression Process
compressnet     3/udp    # Compression Process
rje             5/tcp    # Remote Job Entry
rje             5/udp    # Remote Job Entry
echo            7/tcp    Echo
echo            7/udp    Echo
discard         9/tcp    # Discard
discard         9/udp    # Discard
systat         11/tcp    users       # Active Users
systat         11/udp    users       # Active Users
daytime        13/tcp    # Daytime (RFC 867)
daytime        13/udp    # Daytime (RFC 867)
netstat        15/tcp    # Unassigned [was netstat]
qotd           17/tcp    quote       # Quote of the Day
```

```
qotd            17/udp    quote       # Quote of the Day
msp             18/tcp    # Message Send Protocol
msp             18/udp    # Message Send Protocol
chargen         19/tcp    # Character Generator
chargen         19/udp    # Character Generator
ftp-data        20/tcp    # File Transfer [Default Data]
ftp-data        20/udp    # File Transfer [Default Data]
ftp             21/tcp    # File Transfer [Control]
fsp             21/udp    # File Transfer [Control]
ssh             22/tcp    # SSH Remote Login Protocol
ssh             22/udp    # SSH Remote Login Protocol
telnet          23/tcp    # Telnet
telnet          23/udp    # Telnet
...
```

e. /etc/protocols

El archivo **`/etc/protocols`** contiene la lista de los protocolos conocidos por Unix.

```
# Assigned Internet Protocol Numbers
#
# Decimal   Keyword     Protocol                    References
# -------  -------    --------                ----------
# protocol  num aliases     # comments
hopopt       0 HOPOPT      # IPv6 Hop-by-Hop Option        [RFC1883]
icmp         1 ICMP        # Internet Control Message      [RFC792]
igmp         2 IGMP        # Internet Group Management     [RFC1112]
ggp          3 GGP         # Gateway-to-Gateway            [RFC823]
ip           4 IP          # IP in IP (encapsulation)      [RFC2003]
st           5 ST          # Stream                [RFC1190,RFC1819]
tcp          6 TCP         # Transmission Control          [RFC793]
cbt          7 CBT         # CBT                       [Ballardie]
egp          8 EGP         # Exterior Gateway Protocol  [RFC888,DLM1]
igp          9 IGP         # any private interior gateway     [IANA]
bbn-rcc-mon  10 BBN-RCC-MON  # BBN RCC Monitoring            [SGC]
nvp-ii       11 NVP-II     # Network Voice Protocol     [RFC741,SC3]
pup          12 PUP        # PUP                     [PUP,XEROX]
```

```
argus       13 ARGUS       # ARGUS                    [RWS4]
emcon       14 EMCON       # EMCON                     [BN7]
xnet        15 XNET        # Cross Net Debugger        [IEN158,JFH2]
chaos       16 CHAOS       # Chaos                     [NC3]
udp         17 UDP         # User Datagram             [RFC768,JBP]
...
```