# Comandos para administrar la conexión a la red en Linux

Las distribuciones modernas de Linux (Ubuntu, Fedora, Suse, etc) vienen configuradas de fábrica para que, una vez instaladas, éstas puedan incorporarse a la red de forma automática, ya sea cableado o usando Wifi. El programa más común usado en estos sistemas es “Network Manager” (el ícono con antenita o flechita que aparece en la parte de arriba en Ubuntu, por ejemplo). Esta aplicación provee una interfaz gráfica intuitiva para administrar distintas redes. Por ejemplo, según la ubicación donde se encuentre el computador, Network Manager elegirá automáticamente cuál de todas las redes configuradas es la que se ajusta mejor a la que está conectada. Network Manager se encuentra presente en sistemas orientadas a escritorio, a usuarios normales. Otros sistemas, orientados a servidores, o a administradores de sistemas, ofrece otros mecanismos para administrar las conexiones; por ejemplo, en Ubuntu y Debian, se ofrece el archivo “/etc/network/interfaces” del paquete “ifupdown” (comandos ifup, ifdown), y en Centos y Redhat está “/etc/sysconfig/network” y “/etc/sysconfig/network-scripts” del paquete initscripts, para configurar las interfaces de red, orientado a una conexión fija usando bajos recursos.

Por debajo de estos dos sistemas se encuentran los comandos “ip”, “ss”, “ifconfig”, “route” y “netstat” (entre otros) para configurar directamente los parámetros de red en el sistema operativo

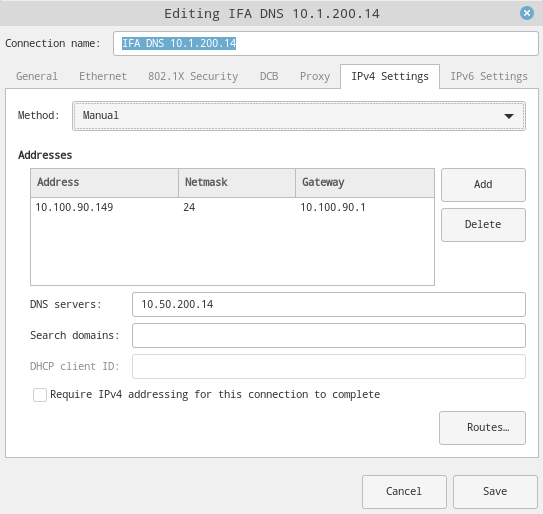


Ilustración : Ejemplo de ventana de Network Manager

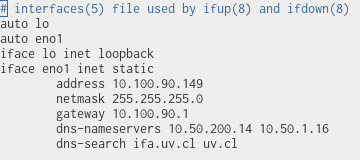


Ilustración : Ejemplo de /etc/network/interfaces

### Configurando la interfaz de red

Si comenzamos con un sistema de cero, tenemos la opción de utilizar los comandos “ip” o “ifconfig” para obtener información sobre nuestras interfaces de red. OJO: “ifconfig” está obsoleto en sistemas Linux, no así en otros Unix, como macOS o FreeBSD.

#### ip link show para ver información sobre las interfaces de red.

root@debian-prueba:~# **ip link show**

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: **ens3:** <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo\_fast state DOWN mode DEFAULT group default qlen 1000

link/ether **52:54:00:d3:6c:52** brd ff:ff:ff:ff:ff:ff

root@debian-prueba:~# **ip link show dev ens3**

2: ens3: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo\_fast state DOWN mode DEFAULT group default qlen 1000

link/ether 52:54:00:d3:6c:52 brd ff:ff:ff:ff:ff:ff

#### ifconfig -a para ver información sobre las interfaces de red.

root@debian-prueba:~# **ifconfig -a**

**ens3:** flags=4098<BROADCAST,MULTICAST> mtu 1500

**ether 52:54:00:d3:6c:5**2 txqueuelen 1000 (Ethernet)

RX packets 4408 bytes 3667508 (3.4 MiB)

RX errors 0 dropped 5 overruns 0 frame 0

TX packets 680 bytes 46899 (45.7 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

**lo:** flags=73<UP,LOOPBACK,RUNNING> mtu 65536

inet 127.0.0.1 netmask 255.0.0.0

inet6 ::1 prefixlen 128 scopeid 0x10<host>

loop txqueuelen 1000 (Local Loopback)

RX packets 0 bytes 0 (0.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 0 bytes 0 (0.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

root@debian-prueba:~# **ifconfig ens3**

ens3: flags=4098<BROADCAST,MULTICAST> mtu 1500

ether 52:54:00:d3:6c:52 txqueuelen 1000 (Ethernet)

RX packets 4408 bytes 3667508 (3.4 MiB)

RX errors 0 dropped 5 overruns 0 frame 0

TX packets 680 bytes 46899 (45.7 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Con estos comandos hemos determinado que las interfaces de red que posee el sistema son “lo” y “ens3”, con dirección física **52:54:00:d3:6c:52**. “lo” siempre es el “loopback”, o sea, un dispositivo de red especial que apunta a nuestro mismo equipo. Nunca conecta a otra red, y siempre tiene una IP en el rango 127.0.0.1 - 127.255.255.254.

Entonces, hemos determinado que nuestro sistema tiene un dispositivo de red cuyo nombre es “ens3”.

#### Configurar IP en dispositivo

Ahora, teniendo la información sobre nuestro dispositivo, y contando con la información sobre nuestra red, podemos configurar la dirección IP de nuestro sistema. En este caso particular, elegimos la IP 10.100.90.70 con máscara de subred 255.255.255.0 (también escribible como /24).

##### Ip addr

root@debian-prueba:~# **ip addr add 10.100.90.70/24 dev ens3**

root@debian-prueba:~# ip addr show

[...]

**2: ens3**: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo\_fast state **DOWN** group default qlen 1000

link/ether 52:54:00:d3:6c:52 brd ff:ff:ff:ff:ff:ff

**inet 10.100.90.70/24** scope global ens3

valid\_lft forever preferred\_lft forever

Se ha asignado la IP 10.100.90.70/24 a la interfaz ens, pero el resultado de “ip addr show” nos muestra que el estado del enlace es “DOWN”, esto significa que aunque esté configurado, no está enviando ni recibiendo datos

root@debian-prueba:~# **ip link show ens3**

2: ens3: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo\_fast **state DOWN** mode DEFAULT group default qlen 1000

link/ether 52:54:00:d3:6c:52 brd ff:ff:ff:ff:ff:ff

Entonces, lo que hacemos es establecer el enlace como “UP”

root@debian-prueba:~# **ip link set up ens3**

root@debian-prueba:~# ip link show ens3

2: ens3: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast **state UP** mode DEFAULT group default qlen 1000

link/ether 52:54:00:d3:6c:52 brd ff:ff:ff:ff:ff:ff

Ahora podemos verificar la conexión. Si tenemos otro host en la red, podemos hacer “ping”, un comando muy usado para verificar la conexión por IP.

root@debian-prueba:~# ping 10.100.90.1

PING 10.100.90.1 (10.100.90.1) 56(84) bytes of data.

64 bytes from 10.100.90.1: icmp\_seq=1 ttl=255 time=1.55 ms

64 bytes from 10.100.90.1: icmp\_seq=2 ttl=255 time=1.72 ms

[...]

##### ifconfig

root@debian-prueba:~# **ifconfig ens3 add inet 10.100.90.70 netmask 255.255.255.0**

root@debian-prueba:~# ifconfig -a

ens3: flags=4163<**UP**,BROADCAST,RUNNING,MULTICAST> mtu 1500

**inet 10.100.90.70 netmask 255.255.255.0** broadcast 10.100.90.255

inet6 fe80::5054:ff:fed3:6c52 prefixlen 64 scopeid 0x20<link>

ether 52:54:00:d3:6c:52 txqueuelen 1000 (Ethernet)

RX packets 6043 bytes 3969712 (3.7 MiB)

RX errors 0 dropped 5 overruns 0 frame 0

TX packets 716 bytes 49763 (48.5 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

[...]

En el caso de ifconfig, al configurar una IP en la interfaz, lo deja en UP automáticamente. Esto podría depender de la distribución. Si no ocurre, también puedes usar “ifconfig ens3 up”

### Rutas, routers y gateways

Las rutas IP es la información que debe tener el sistema operativo para saber cómo llegar a otras direcciones IP. Cuando la interfaz no tiene una IP asignada, entonces no tiene rutas IP. El comando para verificar las rutas es “ip route”, y antiguamente era “route” o “netstat -r”.

root@debian-prueba:~# **ip route**

10.100.90.0/24 dev ens3 proto kernel scope link src 10.100.90.70

root@debian-prueba:~# **route**

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

10.100.90.0 0.0.0.0 255.255.255.0 U 0 0 0 ens3

Las salidas de ambos comandos es distinta, pero nos quiere decir la misma información:

Para llegar a las IPs 10.100.90.0/24 (o 255.255.255.0), hacerlo a través del enlace en la interfaz “ens3”

Esto quiere decir, que ahora puedo llegar a la red 10.100.90.0 directamente. Esta ruta especial (porque es de la red física local) es agregada automáticamente por el Kernel cuando se establece un enlace de red.

Hay que recordar que las IPs que empiezan con “10.” Son privadas y no están en la internet. ¿Qué pasa si quiero llegar a una IP que sí debe estar en la internet? Un ejemplo famoso de IP pública es “8.8.8.8”

root@debian-prueba:~# ping 8.8.8.8

connect: La red es inaccesible

Aquí intentamos llegar a 8.8.8.8, pero “la red es inaccesible”, porque el sistema operativo no sabe cómo llegar a ella. No tiene una ruta para esa red, y debemos agregarla. Pero nada es gratis en este mundo: **Tenemos que decirle al sistema operativo que hay un computador en la red que tiene configurada rutas hacia otras IPs que no pertenecen a la red.** Este computador especial en la red se llama “Gateway”. Un Gateway suele ser un dispositivo llamado “Router”, que es un computador especial dedicado exclusivamente a administrar y descubrir rutas hacia otras redes.

En esta red especial, el administrador de redes nos dijo que el Gateway tiene IP 10.100.90.1 (está dentro de la red). Entonces, literalmente, tenemos que decirle al sistema operativo “cuando te pase una IP a la que no tengas conexión directa, utiliza a este ‘gateway’ para llegar a ella”. A este tipo de ruta se le llama “default”. El Gateway actúa como un intermediario, y para nuestro sistema operativo es transparente, es como si la conexión fuera a través del Gateway, pero a su vez, el Gateway podría estar comunicándose con otros routers, y éstos con otros, así sucesivamente hasta llegar al destino. Para las aplicaciones, esto es transparente.

##### Agregando una ruta “default”

IMPORTANTE: cuando se agregue cualquier ruta, debe existir previamente la ruta hacia el Gateway a usar. Esto implica que, inevitablemente, usaremos el Gateway de nuestra red.

root@debian-prueba:~# ip route add default via 10.100.90.1 dev ens3

root@debian-prueba:~# ip route

**default via 10.100.90.1 dev ens3**

10.100.90.0/24 dev ens3 proto kernel scope link src 10.100.90.70

root@debian-prueba:~# route add default gw 10.100.90.1 ens3

root@debian-prueba:~# route

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

**default 10.100.90.1 0.0.0.0 UG 0 0 0 ens3**

10.100.90.0 0.0.0.0 255.255.255.0 U 0 0 0 ens3

Ahora cualquier IP desconocida para nuestro sistema será alcanzada a través del “default Gateway”. Nótese como en el comando “ip” se usa la palabra “via”, es como decir “otras IPs via 10.100.90.1”. En el comando Route, en la columna Flags, se agrega la letra “G”, que indica que esa ruta es a través de un Gateway (la U quiere decir UP).

root@debian-prueba:~# ping 8.8.8.8

PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.

64 bytes from 8.8.8.8: icmp\_seq=1 ttl=54 time=4.08 ms

64 bytes from 8.8.8.8: icmp\_seq=2 ttl=54 time=3.78 ms

64 bytes from 8.8.8.8: icmp\_seq=3 ttl=54 time=3.79 ms

^C

--- 8.8.8.8 ping statistics ---

3 packets transmitted, 3 received, 0% packet loss, time 5ms

rtt min/avg/max/mdev = 3.775/3.880/4.079/0.157 ms

Revisemos otro comando: “traceroute”. Este comando intenta mostrarnos todos los routers necesarios para llegar a un destino. Los routers pueden rechazar la solicitud.

root@debian-prueba:~# traceroute 8.8.8.8

traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets

1 10.100.90.1 (10.100.90.1) 0.892 ms 1.052 ms 1.273 ms

2 10.60.90.254 (10.60.90.254) 1.365 ms 1.483 ms 1.570 ms

3 \* \* \*

4 \* \* \*

5 \* \* \*

...

En este caso, los routers que venían después de 10.60.90.254 rechazaron la solicitud de traceroute. Pero ya sabemos que 10.100.90.1 (nuestro Gateway) no conoce el camino a 8.8.8.8, entonces usa otro router, el cual es 10.60.90.254. Y así sucesivamente hasta llegar al router que está dentro de la misma red física que 8.8.8.8 .

##### Agregando otras rutas

Podría ocurrir que se quiere agregar rutas a IP usando otros gateways, o incluso otras interfaces. Por ejemplo, cuando tenemos un computador con un dispositivo Wifi y otro por cableado, el sistema operativo automáticamente prefiere el cableado para conectar a internet, o sea, el Gateway default será a través del dispositivo cableado (en el ejemplo es ens3). ¿Y si queremos , por diversos motivos, que las conexiones a alguna red particular sea a través de el dispositivo wifi (wlan0)?

root@debian-prueba:~# ip route

**default via 10.100.90.1 dev ens3**

10.100.90.0/24 dev ens3 proto kernel scope link src 10.100.90.70

192.168.0.0/24 dev wlan0 proto kernel scope link src 192.168.0.100

Agreguemos la ruta a 8.8.8.8 por wlan0:

root@debian-prueba:~# ip route add host 8.8.8.8 via 192.168.0.1 dev wlan0

root@debian-prueba:~# ip route

default via 10.100.90.1 dev ens3

10.100.90.0/24 dev ens3 proto kernel scope link src 10.100.90.70

192.168.0.0/24 dev wlan0 proto kernel scope link src 192.168.0.100

8.8.8.8 via 192.168.0.1 dev wlan0

Y probando con traceroute:

root@debian-prueba:~# traceroute google.cl

traceroute to google.cl (64.233.186.94), 30 hops max, 60 byte packets

1 10.100.90.1 (10.100.90.1) 0.650 ms 1.161 ms 1.319 ms

2 10.60.90.254 (10.60.90.254) 1.295 ms 1.449 ms 1.602 ms

3 \* \* \*

4 \* \* \*

root@debian-prueba:~# traceroute 8.8.8.8

traceroute to 8.8.8.8 (8.8.8.8), 30 hops max, 60 byte packets

1 192.168.0.1 (192.168.0.1) 0.313 ms 0.274 ms 0.258 ms

2 10.0.20.1 (10.0.20.1) 0.810 ms 1.416 ms 1.871 ms

3 host193.200.73.0.static.ifxnw.cl (200.73.0.193) 4.080 ms 4.061 ms 4.024 ms

4 216.241.0.1 (216.241.0.1) 4.438 ms 4.541 ms 4.472 ms

5 static.ifxnw.cl (216.241.0.2) 4.256 ms 4.749 ms 4.768 ms

6 192.168.173.26 (192.168.173.26) 4.678 ms 7.037 ms 3.755 ms

7 208.50.13.49 (208.50.13.49) 5.654 ms 5.662 ms 5.624 ms

8 gw-losparques.nacional.red.movistar.cl (186.148.61.33) 5.597 ms 5.558 ms 5.526 ms

9 186-148-61-98.unknown.mundo.movistar.cl (186.148.61.98) 5.468 ms 5.444 ms 5.419 ms

Para agregar rutas, no podemos usar nombres (por ejemplo, ruta a google.cl), tenemos que traducir el nombre a IP. Para ello podemos usar “ping”.

[root@lisa edgar]# ping google.cl

PING google.cl (**172.217.192.94**) 56(84) bytes of data.

64 bytes from 172.217.192.94 (172.217.192.94): icmp\_seq=1 ttl=49 time=4.60 ms

64 bytes from 172.217.192.94 (172.217.192.94): icmp\_seq=2 ttl=49 time=5.23 ms

64 bytes from 172.217.192.94 (172.217.192.94): icmp\_seq=3 ttl=49 time=4.72 ms

##### Borrar rutas

root@debian-prueba:~# ip route del default dev ens3

root@debian-prueba:~# ip route del 8.8.8.8 dev wlan0

root@debian-prueba:~# ip route del 10.100.91.0/24 dev ens3

### Quitar IP del enlace, apagar el enlace

##### Con iproute2

root@debian-prueba:~# **ip addr del 10.100.90.70/24 dev ens3**

root@debian-prueba:~# ip addr show

1: lo: <LOOPBACK,UP,LOWER\_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000

link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00

inet 127.0.0.1/8 scope host lo

valid\_lft forever preferred\_lft forever

inet6 ::1/128 scope host

valid\_lft forever preferred\_lft forever

**2: ens3: <BROADCAST,MULTICAST,UP,LOWER\_UP> mtu 1500 qdisc pfifo\_fast state UP group default qlen 1000**

**link/ether 52:54:00:d3:6c:52 brd ff:ff:ff:ff:ff:ff**

**inet6 fe80::5054:ff:fed3:6c52/64 scope link**

**valid\_lft forever preferred\_lft forever**

root@debian-prueba:~# **ip link set down ens3**

##### Con ifconfig

root@debian-prueba:~# ifconfig ens3 del 10.100.90.70 netmask 255.255.255.0

root@debian-prueba:~# ifconfig

ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

inet 10.100.90.70 netmask 255.255.255.0 broadcast 10.100.90.255

inet6 fe80::5054:ff:fed3:6c52 prefixlen 64 scopeid 0x20<link>

ether 52:54:00:d3:6c:52 txqueuelen 1000 (Ethernet)

RX packets 6181 bytes 3992460 (3.8 MiB)

RX errors 0 dropped 5 overruns 0 frame 0

TX packets 726 bytes 50559 (49.3 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536

inet 127.0.0.1 netmask 255.0.0.0

inet6 ::1 prefixlen 128 scopeid 0x10<host>

loop txqueuelen 1000 (Local Loopback)

RX packets 0 bytes 0 (0.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 0 bytes 0 (0.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Con ifconfig, aunque se haya solicitado borrar la IP, el estado de la interfaz no cambiará hasta que se solicite agregar otra IP. Para este, se utiliza agregar la IP 0.0.0.0

root@debian-prueba:~# ifconfig ens3 0.0.0.0

root@debian-prueba:~# ifconfig

ens3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

inet6 fe80::5054:ff:fed3:6c52 prefixlen 64 scopeid 0x20<link>

ether 52:54:00:d3:6c:52 txqueuelen 1000 (Ethernet)

RX packets 6253 bytes 4009172 (3.8 MiB)

RX errors 0 dropped 5 overruns 0 frame 0

TX packets 732 bytes 51059 (49.8 KiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536

inet 127.0.0.1 netmask 255.0.0.0

inet6 ::1 prefixlen 128 scopeid 0x10<host>

loop txqueuelen 1000 (Local Loopback)

RX packets 0 bytes 0 (0.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 0 bytes 0 (0.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Y luego, para dar de baja la interfaz:

root@debian-prueba:~# ifconfig ens3 down

## Resolución de nombres

Ya vimos como asignar una dirección IP a una interfaz en el sistema, y entendemos qué son las rutas y como manejarlas. Pero ahora necesitamos saber un poco más sobre cómo nos conectamos a otros sistemas. ¿Qué pasa cuando en el navegador escribimos [www.google.cl](http://www.google.cl)? ¿Cómo sabe el sistema que existe aquel nombre? ¿Y cómo sabe lo que significa (en este caso, una IP)? Hay que tomar en cuenta que los nombres de dominio (como Google.cl) son elementos externos al sistema operativo, y, por muy famosos que sean algunos, no tiene porqué saberlos. Todos los días se crean miles de nombres de dominio nuevos, y tiene que existir un mecanismo para que el sistema sepa qué significan. Hay que recordar que nada es gratis.

### Hostname y archivo hosts

Hostname es el nombre que se le da al computador y está pensado para identificarlo en una red. Puede ser una palabra, en general minúsculas, y tiene un formato específico:

* Debe empezar con letra
* Puede contener números
* Puede tener guiones
* Los puntos separa dominios\*\*\*

Un ejemplo de hostname de un equipo es “edgar-computer”, o “www”, o “mail”. En general, el nombre puede servir para identificar al equipo, o para indicar alguna función específica. En una red local podemos tener equipos con nombres de los usuarios, y otros equipos con nombres de la función que cumplen en la red (www es servidor web, mail es servidor de email, ftp es servidor de archivos, etc).

Si tenemos una red local, normalmente a los nombres de host se les pondrá al final la palabra “.local”. Entonces, el nombre “www.local” podrá indicarnos el servidor web en nuestra red local. Pero si tuviéramos muchas redes, entonces podríamos usar nombres que nos identifique dentro de nuestra red. Por ejemplo, [www.ifa](http://www.ifa) podría ser el servidor web del ifa. [www.ifa.uv](http://www.ifa.uv) es el servidor web de algo llamado “ifa” de la universidad de Valparaíso. [www.ifa.uv.cl](http://www.ifa.uv.cl) es el servidor web del ifa, de la Universidad de Valparaíso, en Chile.

Todos aquellos nombres que son separados por el punto se les llama “dominio”, y el nombre más a la izquierda será el “nombre del host”, o “hostname”. Entonces, [www.ifa.uv.cl](http://www.ifa.uv.cl) se lee como:

* El host www
* En el subdominio ifa
* En el dominio uv
* En el domino de nivel superior cl
* En general: hostname.subdomain.subdomain.subdomain.domain.topleveldomain

El archivo hosts (/etc/hosts) es un archivo muy simple el cual asocia una dirección IP con uno (o muchos) nombres de hosts

Por ejemplo

127.0.0.1 localhost

127.0.1.1 pluton.ifa.uv.cl pluton

# The following lines are desirable for IPv6 capable hosts

::1 ip6-localhost ip6-loopback

fe00::0 ip6-localnet

ff00::0 ip6-mcastprefix

ff02::1 ip6-allnodes

ff02::2 ip6-allrouters

# servers in IFA

10.100.90.209 galaxy.ifa.uv.cl galaxy

10.100.90.213 mad.ifa.uv.cl mad

10.100.91.243 colossus.ifa.uv.cl colossus.local colossus

10.100.91.191 reksio.ifa.uv.cl reksio

10.100.90.178 snoopy.ifa.uv.cl snoopy

10.100.90.221 debris.ifa.uv.cl debris

10.100.90.222 binary.ifa.uv.cl binary

10.100.91.37 hungry.ifa.uv.cl hungry

10.100.90.191 greedy.ifa.uv.cl greedy

10.100.92.30 ruska.ifa.uv.cl ruska

10.100.92.31 pataky.ifa.uv.cl pataky

Este archivo es sólo local, significa que no se propaga a través de la red, y cada computador maneja el suyo propio, entonces el inconveniente es que si queremos usar nombres para acceder a servicios en nuestra red, debemos agregar los nombres manualmente a todos los archivos hosts en todos los computadores.

### DNS y resolv.conf

El DNS (Domain Name System o Servers) es el sistema automatizado y público que asigna nombres de dominio a IPs públicas y privadas.

El sistema DNS es más complejo que sólo devolver IPs. El sistema de nombres está administrado por la ICANN ([www.icann.org](http://www.icann.org)) el cual administra los Dominios de nivel superior (ejemplos, .com, .net, .org, .cl, .edu, etc). Se puede hacer una analogía a una jerarquía de directorios en un sistema de archivos. En la raíz encontraríamos com, net, org, edu, cl, y otros. Si entráramos a cl, encontraríamos uv, entel, vtr, google, y muchos más. Si entramos a uv, entontraríamos a ifa, ciencias, www… En este caso, “www” ya no sería una carpeta; sería un archivo, y en la perspectiva del DNS, ya no sería un dominio, sino un host.

Si algún día compras un dominio en “.cl” (administrado por nic.cl), supongamos, “dominio.cl”, para poder tener control de ese dominio, deberás agregarlo a un servidor de nombre público (normalmente viene con los hosting). En dicho servidor DNS podrás controlar los subdominios de “dominio.cl”. Algunas entradas:

* A, nombre de host asociado a IP
* AAAA, nombre de host asociado a IPv6
* CNAME, palabra que apunta a otro host, por ejemplo, tu servidor web se llama “nubes.dominio.cl”, pero con CNAME lo puedes llamar también [www.dominio.cl](http://www.dominio.cl)
* MX, nombre asociado a servidor WEB, para ser usado con el protocolo SMTP.

### Resolv.conf

Cuando una aplicación inicia una conexión a un servidor usando el nombre, el sistema operativo debe convertir ese nombre en una dirección IP.

Primero intentará con /etc/hosts. Si falla, intentará con un servidor DNS, configurado en /etc/resolv.conf. El servidor DNS devolverá la IP solicitada. A este acto se le llama “resolución de nombres”.

Ejemplo de resolv.conf

nameserver 10.50.200.14

nameserver 10.50.1.16

search ifa.uv.cl uv.cl

En este caso, tenemos dos servidores DNS en nuestra lista (nameserver), si el primero no está disponible, entonces intentará con el segundo. La línea que dice “search” indica que, si el sistema ha solicitado resolver una palabra, y no se ha podido, entonces debe probar también con “palabra.ifa.uv.cl” y “palabra.uv.cl”, en ese orden.

### Systemd y resolvconf

En un sistema Unix tradicional, el archivo resolv.conf es administrado por el usuario administrador del sistema. En los sistemas modernos Linux, resolv.conf puede ser administrado por el demonio “resolvconf”, o “systemd” (sí, el mismo que init), o ifupdown.

## Ifupdown

El sistema ifupdown administra las interfaces de redes de forma automática a partir de los scripts que residen en /etc/network. El principal de ellos es /etc/network/interfaces. En un sistema de escritorio Linux normal, las redes serán administradas por el demonio NetworkManager, pero este demonio verificará que las interfaces a administrar no se encuentren en el archivo /etc/network/interfaces. Si los encuentra, dejará que ifupdown los administre.

Cuando un dispositivo está configurado con ifupdown, éste se administra con dos comandos:

* ifup dispositivo: configura el dispositivo y lo levanta (UP)
* ifdown dispositivo: desconfigura al dispositivo, y lo baja (down)

### Configurar /etc/network/interfaces

Cuando tenemos un equipo debian nuevo, recién instalado, de cero, y sin nada, nuestro archivo /etc/network/interfaces luce así:

# This file describes the network interfaces available on your system

# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/\*

# The loopback network interface

auto lo

iface lo inet loopback

La línea que dice “source” nos informa que podemos poner archivos de configuración en el directorio /etc/network/interfaces.d y será como haberlos escrito directamente en este archivo. Así es más fácil administrar algunas configuraciones.

La línea “auto lo” nos dice que el dispositivo loopback se activará automáticamente al iniciar el computador.

La línea “iface” configura la interfaz “lo”, como loopback. Las líneas que empiezan con “iface” son las que indican a otros sistemas que la interfaz está administrada por ifupdown.

##### Agregando un dispositivo

Suponiendo que vamos a configurar el dispositivo “ens3”, podemos configurarlo para una red IP de las siguientes formas:

* iface ens3 inet dhcp
* iface ens3 inet static

La primera opción configura al dispositivo para que, al activarse, solicite una IP automática al servidor DHCPD de la red (es lo que normalmente se encuentra en las redes domésticas). La segunda opción nos permitirá configurarlo con parámetros escogidos por nosotros. Nos enfocaremos en eso ahora:

iface ens3 inet static

address 10.100.90.70 #la dirección IP de nuestro computador

netmask 255.255.255.0 #la máscara de subred

gateway 10.100.90.1 #El default gateway

# hasta este punto, es como escribir

# ip addr add 10.100.90.70/24 dev ens3

# ip link set up ens3

# ip route add default via 10.100.90.1

Ahora podemos guardar el archivo y escribir

# ifup ens3

Después de esto, la interfaz debería estar configurada. Probar con “ping 8.8.8.8”.

* OJO: Si sale error de que no puede configurar el dispositivo, verificar que el archivo esté bien escrito, y probar también con estos comandos antes:
  + ifdown ens3
  + ip addr flush dev ens3
  + ip link set down ens3
  + ifup ens3

Sólo queda indicar el servidor DNS, y tenemos dos opciones; administrarlo en /etc/resolv.conf, o instalar el paquete “resolvconf”

Si elegimos resolvconf (si no está instalado, **instalar**), entonces tenemos que agregar las líneas:

dns-nameservers ip1 ip2 ip3 #ips separadas por espacio

por ejemplo:

dns-nameservers 8.8.8.8 8.8.4.4 #8.8.8.8 y 8.8.4.4 son servidores DNS famosos

Finalmente, el dispositivo está configurado, pero para activar estas configuraciones el usuario debe escribir “ifup ens3”. Para que sea automático, al iniciar el sistema, agregar la línea (antes de iface):

auto ens3

El archivo final:

# This file describes the network interfaces available on your system

# and how to activate them. For more information, see interfaces(5).

source /etc/network/interfaces.d/\*

# The loopback network interface

auto lo

iface lo inet loopback

auto ens3

iface ens3 inet static

address 10.100.90.70 #la dirección IP de nuestro computador

netmask 255.255.255.0 #la máscara de subred

gateway 10.100.90.1 #El default gateway

dns-nameservers 8.8.8.8 8.8.4.4

## Ver el estado de las conexiones

Para ver las conexiones establecida a otros servidores tenemos los comandos “ss” y el antiguo “netstat”

ss -4 #nos mostrará conexiones del protocolo IPv4

netstat -4 #nos mostrará conexiones del protocolo IPv4

root@debian-prueba:/home/alumno# ss -4

Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port

tcp ESTAB 0 0 10.100.90.70:ssh 10.100.90.149:58608

~ $ ss -4

Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port

udp ESTAB 0 0 127.0.0.1:59028 127.0.0.1:59028

tcp ESTAB 0 0 10.100.90.149:48312 172.217.192.19:https

tcp ESTAB 0 0 127.0.0.1:postgresql 127.0.0.1:53538

tcp ESTAB 0 0 127.0.0.1:53572 127.0.0.1:postgresql

tcp ESTAB 0 0 127.0.0.1:53554 127.0.0.1:postgresql

tcp ESTAB 0 0 10.100.90.149:40490 185.206.24.10:https

tcp ESTAB 0 0 10.100.90.149:54854 198.252.206.25:https

tcp CLOSE-WAIT 1 0 10.100.90.149:54686 35.186.224.53:https

tcp ESTAB 0 0 10.100.90.149:ssh 10.100.90.189:60502

Esto nos muestra las conexiones a servidores externos desde nuestro computador. Pero si queremos ver qué servidores tenemos, entonces

ss -4l #puertos TCP o UDP abiertos y escuchando

netstat -4l #puertos TCP o UDP abiertos y escuchando

# también podemos agregar la opción -n para convertir todo en número

# y la opción -t para mostrar conexiones TCP

$ ss -4l

Netid State Recv-Q Send-Q Local Address:Port Peer Address:Port

??? UNCONN 0 0 0.0.0.0:icmp 0.0.0.0:\*

udp UNCONN 0 0 0.0.0.0:44776 0.0.0.0:\*

udp UNCONN 0 0 127.0.0.1:63457 0.0.0.0:\*

udp UNCONN 0 0 192.168.123.1:domain 0.0.0.0:\*

udp UNCONN 0 0 127.0.0.53%lo:domain 0.0.0.0:\*

tcp LISTEN 0 128 127.0.0.1:5900 0.0.0.0:\*

tcp LISTEN 0 128 0.0.0.0:111 0.0.0.0:\*

tcp LISTEN 0 128 0.0.0.0:80 0.0.0.0:\*

tcp LISTEN 0 5 0.0.0.0:41809 0.0.0.0:\*

tcp LISTEN 0 10 0.0.0.0:57621 0.0.0.0:\*

tcp LISTEN 0 32 192.168.123.1:53 0.0.0.0:\*

tcp LISTEN 0 128 127.0.0.53%lo:53 0.0.0.0:\*

tcp LISTEN 0 128 0.0.0.0:5910 0.0.0.0:\*

tcp LISTEN 0 128 0.0.0.0:22 0.0.0.0:\*

tcp LISTEN 0 5 127.0.0.1:631 0.0.0.0:\*