- Gestión de interfaces:
  - Administración direcciones IP
  - Configuración del dispositivos físicos y lógicos
  - Administración de ARP
  - Configuración del dispositivo MACsec
- Enrutamiento:
  - Ver rutas del sistema
  - Añadir rutas estáticas
  - Borrar rutas estaticas
  - Hacer persistentes las rutas
- Domain Name System o DNS:
  - smallpdf
- Resumen general
- Conclusion final

Iproute2 es un paquete de utilidades desarrollado por Alexey Kuznetsov. Este paquete es un conjunto de herramientas muy potentes para administrar interfaces de red y conexiones en sistemas Linux.

Este paquete reemplaza completamente las funcionalidades presentes en ifconfig, route, y arp y las extiende llegando a tener características similares a las provistas por dispositivos exclusivamente dedicados al ruteo y control de tráfico.

Este paquete lo podemos encontrar incluido en distribuciones de Debian y RedHat con versiones del núcleo mayores a 2.2.

El comando *ip* se utiliza para asignar una dirección a una interfaz de red y/o configurar parámetros de interfaz de red en sistemas operativos Linux. Este comando reemplaza el comando *ifconfig* antiguo y ahora obsoleto en las distribuciones modernas de Linux. Como curiosidad de este comando es que no es necesario ponerlo completo, si pones abreviaciones de este funciona correctamente siempre que no hay ambigüedad con otros, como con los comandos en Cisco.

# Gestión de interfaces

## Administración direcciones IP

Podemos ver la configuración de las interfaces con el comando *ip address*. También se puede ver de forma abreviada:

```
# Ver interfaces
ip address

# Forma abreviada
ip a
ip addr

# Interfaz enp0s3
ip address show enp0s3
```

También podemos añadir interfaces y borrarlas con el argumento *add* y *del*. En el caso de añadir una IP, si ya existe se añadirá una nueva dirección IP "virtual".

```
# Añadir IP
sudo ip address add 192.168.1.105/24 dev enp0s3

# Borrar IP
sudo ip address del 192.168.1.105/24 dev enp0s3
```

## Configuración del dispositivos físicos y lógicos

Las interfaces se manejan con el comando *ip link*. Con este comando podemos modificar "físicamente" las interfaces, tumbándolas y levantándolas.

```
# Levantar la interfaz
sudo ip link set enp0s3 up

# Tumbar la interfaz
sudo ip link set enp0s3 down
```

También se pueden crear nuevos interfaces, algunos de estos interfaces son: bridge, can, macvlan, vlan, macsec, etc). Por ejemplo podemos crear un interfaz bridge y añadirle interfaces a ese bridge de la siguiente forma:

```
sudo ip link add br0 type bridge
sudo ip link set enp0s3 master br0
sudo ip link set enp0s8 master br0
sudo ip link set enp0s9 master br0
```

Se pueden ver las interfaces asociadas a cada bridge con el siguiente comando:

```
bridge link
```

También es se pueden crear VLANs asociadas a una interfaz con el siguiente comando:

```
sudo ip link add link enp0s3 name enp0s3.50 type vlan id 50
```

Fuentes: 0, 1, 2

## Administración de ARP

Para ver la tabla ARP tenemos el comando *ip neigh*.

```
ip neigh show
```

En la ultima columna nos muestra el estado del vecino, que puede ser uno de estos estados:

- REACHABLE: la entrada ARP es válida y hay conectividad.
- STALE: la entrada ARP es válida pero no hay conectividad.
- FAILED: no hay conectividad y la MAC no ha sido detectada.
- DELAY: a la espera de confirmación tras el envío de un paquete.

También puede ser muy interesante poner fija la tabla ARP para evitar que nos puedan hacer MAC Spofing, esto se realiza con:

```
# Para evitar conflictos hay que bajar la interfaz o quitarle la IP
sudo ip addr flush dev enp0s3

# Asignamos la MAC asociada a la IP
sudo ip neigh add 192.168.1.1 lladdr d0:6e:d5:54:3d:a4 nud permanent dev enp0s3

# Volvemos a solicitar las IPs
sudo netplan apply
```

Para borrar una entrada sería:

```
sudo ip neigh del 192.168.1.1 dev enp0s3
```

Para vaciar la tabla ARP o borrar una única entrada sería:

```
sudo ip -s -s neigh flush all
sudo ip -s -s neigh flush 192.168.1.1
```

Fuentes: 0, 1

## Configuración del dispositivo MACsec

MACsec es un estándar IEEE (IEEE 802.1AE) para seguridad MAC, introducido en 2006. Define una forma de establecer una conexión independiente de protocolo entre dos hosts con confidencialidad, autenticidad y/o integridad de datos, utilizando GCM-AES-128. MACsec opera en la capa Ethernet y, como tal, es un protocolo de capa 2, lo que significa que está diseñado para proteger el tráfico dentro de una red de capa 2, incluidas las solicitudes DHCP o ARP. No compite con otras soluciones de seguridad como IPsec (capa 3) o TLS (capa 4), ya que todas esas soluciones se utilizan para sus propios casos de uso específicos.

IMPORTANTE: NO LO HE PROBADO AÚN

```
sudo ip link add macsec0 link eth1 type macsec
```

Fuentes: 0

# Enrutamiento

Una ruta estática no es más que una forma de especificar el tráfico que no debe pasar por la puerta de enlace predeterminada. Se puede usar el comando *ip* para agregar una ruta estática a una red diferente a la que no se puede acceder a través de su puerta de enlace predeterminada. Por ejemplo, la puerta de enlace VPN.

## Ver rutas del sistema

Podemos ver cuales son las rutas por defecto del sistema con el comando *ip route*.

```
ip route show
```

También se puede hacer un mayor debug solicitando al sistema que te diga cual es la IP e interfaz por la que saldrá un paquete indicando la IP destino. (Ojo, no es capaz de resolver un FQND).

```
ip route get 192.168.1.1

ip route get 8.8.8.8
```

## Añadir rutas estáticas

La sintaxis para añadir rutas estáticas es la siguiente:

*ip route add {NETWORK/MASK} via {GATEWAYIP}*

*ip route add {NETWORK/MASK} dev {INTERFACE}*

*ip route add default {NETWORK/MASK} dev {INTERFACE}*

*ip route add default {NETWORK/MASK} via {GATEWAYIP}*

Por ejemplo a continuación se puede ver como se enruta el trafico de una red por la interfaz vlan.100 y como se envía en trafico para un host por la interfaz enp0s3.

```
# RED
sudo ip route add 192.168.20.0/24 via 192.168.10.10 dev vlan.100

# HOST
sudo ip route add 192.168.20.5 via 192.168.10.10 dev enp0s3
```

## Borrar rutas estaticas

La sintaxis para añadir rutas estáticas es la siguiente:

*sudo ip route add default via {NETWORK/MASK} dev {INTERFACE}*

*sudo ip route del default*

```
sudo ip route del 192.168.20.0/24 via 192.168.10.10 dev vlan.100

sudo ip route del default
```

## Hacer persistentes las rutas

Los comandos mencionados anteriormente son volatiles, y cuando sereinicia el ordenador se borrar. Se podrian hacer las rutas persistentes de la siguiente forma:

### Enrutamiento persistentes en (RHEL, Fedora, CentOS)

En el directorio /etc/sysconfig/network-scripts/ hay que crear un fichero por cada interfaz de host, donde el nombre del fichero sera el nombre de la interfaz. En nuestro caso, esto será route-enp0s3.

En nuestro caso hay que añadir al fichero */etc/sysconfig/network-scripts/route-enps03* las siguientes lineas:

```
10.0.2.0/32 via 192.168.43.1
10.0.2.15  via 192.168.43.1
```

Guardamos el fichero y reiniciamos el servicio NetworkManager:

```
sudo systemctl restart NetworkManager
```

Fuentes: 0, 1

**Enrutamiento persistentes en (Debian, Ubuntu)**

PENDIENTE DE ESCRIBIR CON NETPLAN

## Domain Name System o DNS

cat /etc/resolv.conf

sudo ss -lpunta

systemd-resolve –status

dig www.mytcip.com para ver el server 127.0.0.53

systemd-resolve –statistics sudo systemd-resolve –flush-caches systemd-resolve –statistics

POSIBLEMENTE LA PARTE DE CONFIGURACION DE DNS VAYA EN NETPLAN

https://wiki.archlinux.org/index.php/Systemd-resolved_(Espa%C3%B1ol)

sudo apt install resolvconf https://www.systeminside.net/como-soluciono-resolucion-dns-ubuntu/

https://moss.sh/es/configuracion-problematica-systemd-resolved/

ip COMMAND Cheat Sheet for Red Hat Enterprise Linux