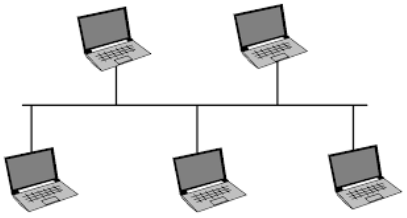
5. RED



5.1.- Introducción

Una de la cosas más importantes a configurar en un sistema es la red, sin red o con ella mal configurada, un sistema estará inaccesible desde fuera de su propio terminal con lo que es como si no existiese para el resto del mundo. Hay que conocer unos conceptos previos que pese a ser muy básicos, en ocasiones se olvidan o incluso ni se conocen.

Para configurar la red en un sistema, necesitamos interfaces de red. Una interfaz de red es una forma abstracta de acceder al hardware, las tarjetas de red, que lleva el sistema. Tradicionalmente, los interfaces de red en Linux han sido nombrados como *eth0*, *eth1*, … En CentOS7, la asignación de nombres a los interfaces ha cambiado; se asignan nombres fijos basados en el firmware, la topología y tipo de dispositivo. El nombre se forma con:

- primeros caracteres: indican el tipo, *en* para interfaces ethernet, *wl* para interfaces WLAN, *ww* para interfaces WWAN.
- siguientes caracteres indican el tipo de adaptador: *o* para on-board, *s* para slot hotplug, *p* para PCI.
- por último, un número que indica un índice, ID o puerto.

Si con estos parámetros no se puede determinar el nombre, se siguen usando los nombres tradicionales.

Por ejemplo, la primera tarjeta de red pinchada en el equipo tendrá el nombre de *eno1* y la tarjeta PCI *enp2s0*.

Existe una interfaz que siempre está configurada en el sistema, la interfaz de *loopback*, con el nombre *lo*, con IPv4 *127.0.0.1* e IPv6 *127.0.0.1::1*. Esta interfaz es virtual, no existe físicamente y representa al propio sistema. Se utiliza cuando en una transmisión de datos, el destino es el propio sistema.

5.2.- Conceptos

5.2.1.- TCP/IP (**T**ransmission **C**ontrol **P**rotocol/**I**nternet **P**rotocol)

Conjunto de protocolos que son la base de Internet permitiendo que máquinas con sistemas operativos distintos puedan comunicarse entre sí. Desarrollado por el departamento de defensa de EEUU en los 70 como un estándar de modelo de red de 4 capas, las cuales son:

- *Aplicación*: define cómo clientes y servidores deben comunicarse.

  - Protocolos: SSH, HTTP, HTTPS, NFS, CIFS, SMTP, FTP, DNS, ….

- *Transporte*: define cómo debe realizarse el enrutamiento junto con mecanismos de control de transmisión, se encarga de la fiabilidad. Los protocolos de aplicación usan puertos TCP y/o UDP (se pueden consultar en */etc/services*).

  - Protocolos: TCP (confiable, orientado a conexión, envía paquetes) y UDP (no confiable, envía datagramas).

  - Maneja: *sockets* (dirección IP + número puerto).

- *Internet o capa de red*: transmite datos de un sistema origen a un sistema destino, capturando los paquetes que correspondan de la red. Cada *host* tiene una dirección IP y *prefix* o máscara.

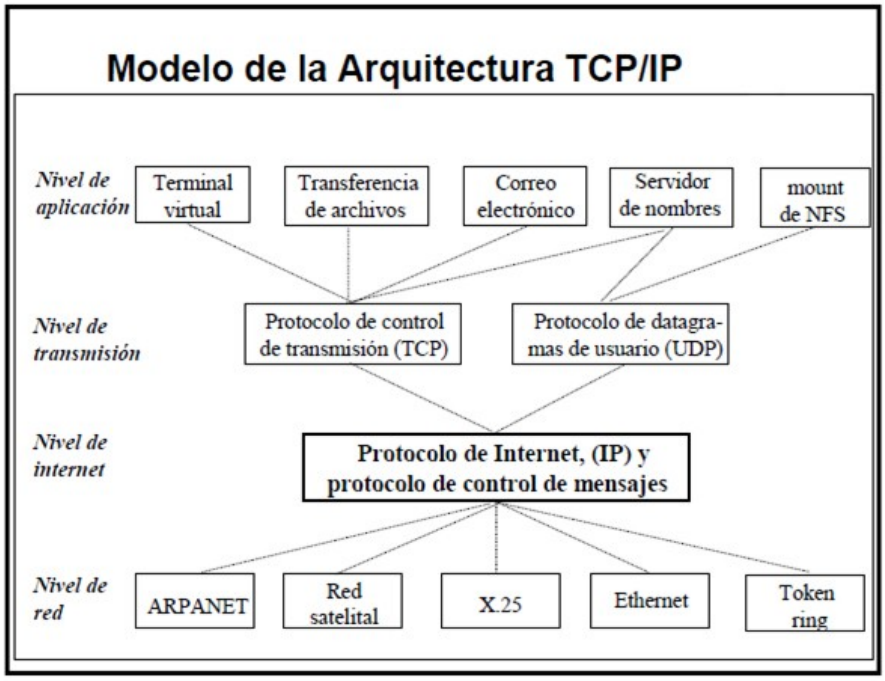
  - Protocolos: IP (IPv4 y IPv6), ICMP, ARP, RARP.

  - Maneja: direcciones IP.

- *Enlace o acceso a la red*: proporciona la conexión al medio físico, cómo deben enviarse los datos independientemente del tipo de red físico utilizado.

  - Protocolos: Ethernet, Wireless, ATM, Framerelay, PPP, …

  - Maneja: direcciones MAC.



Capas de TCP/IP

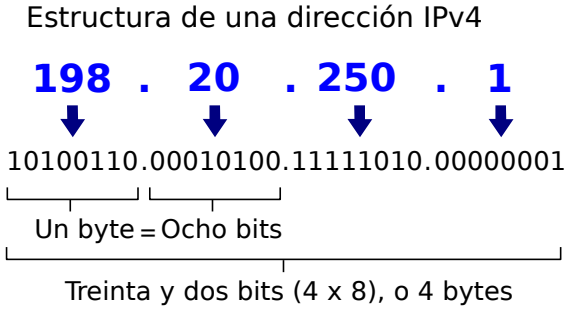
5.2.2.- IPv4

Versión 4 del protocolo IP de Internet. Su objetivo es proporcionar una dirección única a cada sistema dentro de una red para que sea identificado de forma unívoca.

Existen unos bloques de direcciones reservados, p.e. *10.0.0.0/8, 172.16.0.0/16* y *192.168.0.0/16* para redes privadas, *127.0.0.0/8* para la subred de *localhost*, ...

Actualmente el ICANN (**I**nternet **C**orporation for **A**ssigned **N**ames and **N**umbers) se encarga de la asignación de direcciones de Internet alrededor del mundo.

→ *Formato de dirección IPv4*: 32 bits (4 bytes) representados como 4 octetos separados por el carácter punto. La forma más común es la representación decimal de 4 números donde cada uno de ellos va de 0 a 255.



Formato de direcciones IPv4

Las direcciones IPv4 tienen dos partes: parte de red y parte de *host*. Todos los *host* de una misma subred tienen la misma parte de red y difieren en la parte de *host*. Para saber qué parte es de red y qué parte de *host*, se necesita la máscara asignada a la subred.

La dirección más baja en una subred, donde la parte de *host* son todo ceros, se conoce como la dirección de red de esa subred y la dirección más alta, todo a unos en la parte de *host*, es la dirección de *broadcast*. Ninguna de estas dos direcciones son utilizables por un sistema.

La máscara se puede expresar de dos formas: la tradicional, usando 4 octetos separados por puntos donde la parte de red va toda a unos y la parte de *host* a cero o con la notación CIDR, más moderna, que especifica un prefijo de red o *prefix*, un número que indica cuántos unos hay en la máscara, (empezando por la izquierda).

En IPv4 existe el concepto de dirección de *broadcast*, es una dirección en una subred donde los paquetes que se envíen son recibidos por todos los sistemas conectados a esa subred.

Existe el comando *ipcalc* que dada una dirección IP y una máscara o *prefix*, con la opción *-n* obtenemos la dirección de red del *host* y con la opción *-b*, la dirección de *broadcast*.

5.2.3.- IPv6

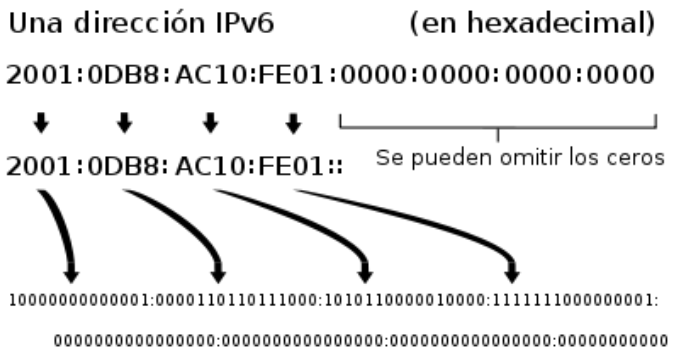
Es la siguiente versión del protocolo IPv4. Debido al agotamiento de las direcciones IPv4, ya en los 90 se comienza a desarrollar una ampliación, llegando a la solución de IPv6 lanzado finalmente en 2012.

Existe un problema con la implantación de IPv6, la dificultad de comunicar un equipo con una dirección IPv4 con otro que tenga dirección IPv6. La solución es optar por una pila de protocolos doble, donde los sistemas tengan dirección IPv4 e IPv6 hasta conseguir que todo funcione con IPv6.

→ *Formato de dirección IPv6*: Las direcciones IPv6 están formadas por 128 bits formando 8 grupos de 4 dígitos hexadecimales separados por el carácter dos puntos donde cada grupo representa 16 bits.

Hay unas normas de escritura para acortar la representación:

- los ceros consecutivos no se escriben, se pone sólo ::
- los ceros a la izquierda en un grupo se omiten
- si hay varios bloques "::", se usa "::" para el bloque más largo y el resto se escriben como ":0:"
- a igual longitud de ceros en varios grupos, se deja "::" para el que está más a la izquierda y el resto con ":0:"
- se usan letras minúsculas siempre
- para indicar un número de puerto con ":puerto", la IP se pone entre corchetes



Formato de direcciones IPv6

En IPv6 se suele usar una máscara estándar /64, con lo que divide la dirección en dos partes lógicas: prefijo de subred de 64 bits y los 64 restantes para identificar la interfaz.

Existe una dirección de *link-local* que es una dirección no enrutable que se usa para hablar con los *host* de una determinada subred. Cada interfaz en la subred configura de forma automática esta IP en la subred *fe80::*. De forma automática, el sistema construye la dirección de *link-local* a partir de la MAC de la interfaz, obteniéndose una IP única que se puede usar como una dirección normal pero añadiendo detrás el carácter % seguido de la interfaz que se quiere usar para el envío.
En IPv6 no existe el concepto de *broadcast*, en su lugar tenemos *multicast*, enviando paquetes a la dirección de *multicast ff02::1* (con %<interfaz> al final) todos los sistemas en el *link-local* los recibirán.

5.2.4.- Tabla de enrutamiento

Cada *host* tiene una tabla de enrutamiento que le indica cómo debe enrutar el tráfico para unas redes en particular. Está compuesta por entradas que tienen una red de destino, la interfaz que debe usar para enviar el tráfico a esa red y (si se necesita) el router al que enviárselo. Además hay una entrada, la *default route*, indicada en IPv4 como *0.0.0.0/0*, para el caso que no sepa enrutar algo, lo envía al router ahí configurado. Para ver la tabla de enrutamiento del sistema utilizaremos el comando *ip route* y para modificarla con *ip route add|del|append|change* (ver más abajo).

5.2.5.- Conceptos varios

- *DHCP*: Asignación dinámica de dirección IP al arrancar un sistema, se conecta a un servidor DHCP que en base a su MAC le proporciona una IP, *gateway*, servidor DNS, ….
- *Gateway*: sistema que está conectado a varias redes y que es capaz de enrutar paquetes de una red a otra. El *default gateway* que se configura en cada sistema, debe estar en la misma subred que este.
- *DNS*: protocolo que permite asociar direcciones IP con nombres de dominio. Los sistemas utilizan direcciones IP para comunicarse unos con otros pero son difíciles de recordar para las personas y se utilizan nombres de *host* dentro dominios. El servidor DNS es el sistema al que se le harán las consultas DNS; no es necesario que en la misma red que un sistema, basta con que sea accesible.

5.3.- Archivos de red relevantes

- ***/etc/hostname***: hostname del sistema establecido de forma estática.

  Aparece el nombre del sistema, normalmente FQDN (**F**ully **Q**ualified **D**omain **N**ame). Se modifica con el comando *hostnamectl*. En versiones anteriores de CentOS, el hostname del sistema se almacenaba en un parámetro del archivo */etc/sysconfig/network*.

- ***/etc/nsswitch.conf***: orden de búsqueda en las bases de datos de red. En la primera columna aparece la categoría y en la segunda dónde hacer la búsqueda para esa categoría. Las líneas más relevantes con los valores típicos son:
  - *passwd files sss*: dónde busca los usuarios del sistema; primero en el archivo local */etc/passwd* y luego en los servicios de directorio configurados.
  - *shadow files sss*: dónde busca las contraseñas de los usuarios; primero en */etc/shadow* y luego en los servicios de directorio.
  - *group files sss*: donde busca los grupos del sistema; primero en */etc/group* y luego en los servicios de directorio.
  - *hosts files dns*: dónde busca la resolución de nombres a IPs; primero en */etc/hosts* y luego en el DNS (configurado en */etc/resolv.conf* o en la propia interfaz).
- ***/etc/resolv.conf***: resolución de nombres. Este archivo es modificado por Network Manager al hacer cambios en las conexiones o en los archivos *ifcfg*, luego aunque lo modifiquemos a mano, es posible que Network Manager lo sobrescriba. Líneas principales:
  - *nameserver* <*servidor_dns*>: pueden aparecer hasta tres líneas *nameserver*, indican que servidor de DNS consultar y se consultan en el orden en el que aparecen dichas líneas.
  - *search* <*domain*>: si en una búsqueda DNS se proporciona un nombre corto, no FQDN, este es el dominio que le añadirá antes de realizar la consulta.
  - *domain* <*dominio_sistema*>: indica cuál es el dominio del sistema y donde hará la búsqueda en el caso de proporcionar un nombre corto.

  Las líneas *domain* y *search* son incompatibles entre ellas, en el caso de aparecer las dos, la última en ser leída es la que se tiene en cuenta.
- ***/etc/hosts***: mecanismo simple de resolución de nombres local. Asocia direcciones IP con nombres de host y sus alias. Tiene una línea por dirección IP de la forma:

  <dir_IP> <nombre_sistema_fqdn> <alias1> <alias2> …

  La IP y la lista de nombres del sistema, se pueden separar por espacios o tabuladores.

  Siempre apareceren las dos líneas (una de IPv4 y otra de IPv6) de la interfaz de loopback:

  ```
  127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
  ::1       localhost localhost.localdomain localhost6 localhost6.localdomain
  ```
- ***/etc/networks***: parecido a */etc/hosts*, asocia nombres a redes. El formato de cada línea es <*nombre_red*> <*direccion_red*> separados por espacios o tabuladores.

- ***/etc/services***: relaciona las aplicaciones con sus puertos y protocolos correspondientes. Es un archivo estándar en Linux/Unix. El formato de cada línea es: *<servicio> <puerto>/<protocolo> <alias_servicio>*. El alias del servicio es optativo y puede aparecer más de uno y el protocolo es *tcp* o *udp*. Si un mismo servicio tiene varios puertos y/o protocolos, se repite la línea para ese servicio cambiando los valores de puerto y/o protocolo.

5.4.- Comandos de red importantes

- ***hostname***: ver y modificar en *runtime* el nombre de *host* del sistema.

  Durante el proceso de arranque del sistema operativo, se establece el nombre del sistema con la ejecución del comando *hostname* sin parámetros. Si existe el archivo */etc/hostname*, se lee de aquí, y esto significa que se configuró de forma estática. Si no existe el archivo, se consulta el */etc/hosts* y si ahí no lo encuentra, se hará una consulta al DNS por el hostname del sistema dando la IP (resolución inversa).

  Si al comando *hostname* le pasamos un nombre, se modifica el hostname del sistema pero no es persistente, en el siguiente arranque de la máquina se ha perdido. Para hacerlo persistente, podemos modificar a mano el archivo */etc/hostname* o usar el comando *hostanamectl*.
- ***hostnamectl***: ver y establecer de forma persistente el nombre de *host* y otros datos del sistema. Subcomandos relevantes:
  - *status*: *hostname* del sistema y su información relevante.
  - *set-hostname <nombre>*: cambiamos el *hostname* del sistema y se escribe en */etc/hostname*. Si le pasamos “” en el nombre, reseteamos el *hostname*.
  - *set-location <texto>*: añadimos un texto con indicaciones de donde está ubicado el sistema que será luego visible con *hostnamectl status*.
- ***ip***: ver y modificar rutas, dispositivos, etc. del sistema. Sus subcomandos más relevantes:
  - *addr list*, *addr show*, *a*: muestra todos los interfaces de red del sistema y su configuración. Para ver sólo la información de uno en concreto, le pasaremos el nombre de la interfaz.
  - *addr add <dir_ip> dev <interfaz>*, *addr del <dir_ip> dev <interfaz>*: añadir o eliminar una IP de una interfaz de forma no persistente (para que lo sea, usar *nmcli*).
  - *link set <interfaz> down*, *link set <interfaz> up*: desactiva o activa una interfaz dado.

Mejor hacerlo con *nmcli*.

- *route show*: ver la tabla de enrutamiento del sistema.
- *route add <ip_destino>,route del <ip_destino>, append <ip_destino>,change <ip_destino>*: pone, borra, añade y modifica una ruta estática a la tabla de enrutamiento del sistema. La ip de destino puede ser una dirección de red, donde si no se indica prefix, se presupone *24*. Se puede añadir *dev <interfaz>* para que sólo añada la ruta a esa interfaz.

- ***ping***: envía paquetes ICMP a otros hosts dando su dirección IP o su hostname. Por defecto, sigue enviando de forma indefinida hasta que lo detengamos con CTL+C. Que un host no conteste a un comando *ping*, no significa que esté inaccesible, se puede configurar un sistema de forma que no conteste a los paquetes ICMP. Opciones relevantes:
  - *-c <numero_paquetes>*: envía el número de paquetes dado y se detiene.
  - *-4*, *-6*: hago un ping con IPv4 o IPv6. Por defecto si no se indica esta opción, es IPv4.
  - *-i <segundos>*: segundos que deben transcurrir entre paquetes. Por defecto si no se indica nada es 0.2 segundos.
  - *-I <interfaz>*: usa la interfaz dada para enviar los paquetes.
- ***traceroute*** o ***tracepath***: muestran la ruta que siguen los paquetes desde el origen hasta el destino. Cada línea en la salida representa un salto entre subredes. Su sintaxis es *traceroute <opciones> <host_destino>*. Opciones importantes:
  - *-i <interfaz>*: interfaz por donde se enviarán los paquetes.
  - *-m <numero>*: número máximo de saltos que se darán para intentar llegar al destino.
  - *-n*: mostrar las direcciones IP en lugar de los hostnames.
  - *-T, -I*: envía paquetes TCP o ICMP. Por defecto si no se indica ninguna de estas dos opciones, los paquetes son UDP.
- ***ss***: ver estadísticas de red. Reemplaza al comando *netstat*. Sintaxis: *ss <opciones>*.

  Opciones más relevantes:
  - *-n*: muestra números en lugar de nombres para los interfaces y puertos.
  - *-t*, *-u*: muestra *sockets* TCP y UDP respectivamente.
  - *-l*: sólo muestra *sockets* a la escucha (*listening*).
  - *-a*: muestra todos los *sockets*, los que están a la escucha y los establecidos.
  - *-p*: muestra el proceso que está usando el *socket*.

NOTA: Reglas mnemotécnicas para *ss*, usar con las opciones “del tulipan”: *ss -tulpn* o “del atún”: *ss -atun*.

- ***host:*** pasando como argumento un hostname o ip, hace una consulta al servidor DNS del sistema.

5.5.- Network Manager

Conjunto de herramientas de gestión de red que hacen que no sea necesario editar manualmente los archivos de configuración de red. Proporciona interfaces de gestión GUI, CLI y TUI. Es el mecanismo por defecto en CentOS7.

Es muy importante cuando se administra un sistema, saber manejar el CLI de Network manager, *nmcli*, con soltura además de conocer cuál es el formato de los archivos de configuración de red y su equivalencia con los parámetros de *nmcli*.

En Network Manager, un dispositivo es una interfaz de red y una conexión es un conjunto de configuraciones para un dispositivo. Puede haber varias conexiones creadas para el mismo dispositivo pero sólo una activa al tiempo.

5.5.1.- Ver información con nmcli

Para ver las conexiones existentes en el sistema y a que dispositivo están ligadas: *nmcli con show*. Añadiendo la opción *--active*, sólo mostrará las que están activas. Si queremos ver todos los parámetros de alguna en particular, añadimos el nombre de la conexión, entrecomillándolo si lleva espacios en blanco (comillas dobles o simples).

Los parámetros que salen en minúscula son los configurados en la conexión mientras que los que aparecen en mayúsculas son los que están en uso, en *runtime*.

Para ver los dispositivos del sistema y su estado: *mncli dev status* y para ver la información alguno en concreto: *nmcli dev show <nombre_dispositivo>*.

5.5.2.- Crear conexiones con nmcli

El orden en el que aparecen los argumentos es importante, lo mejor es usar el tabulador para se que vayan proporcionando los parámetros a configurar, siendo el último, la dirección IP y el *gateway*.

Sintaxis:

nmcli con add con-name <nombre_conexión> type <tipo> ifname <nombre_dispositivo> <param1> <valor1> <param2> <valor2> … ip4 <dir_ip/prefix> gw <gateway>

5.5.3.- Modificar conexiones con nmcli

Con *nmcli con mod <nombre_conexión> <param> <valor>* modificaremos el parámetro dado. En algunos parámetros se pueden añadir valores a los ya existentes, para esto basta poner delante del parámetro el carácter + y lo mismo para eliminar algún valor, con el carácter -.

Este comando guarda los cambios en los archivos de configuración pero para activar los cambios en la conexión, ésta necesita ser activada o reactivada.

5.5.4.- Activar/Desactivar una conexión o dispositivo con nmcli

Para activar una conexión, basta hacer: *nmcli con up <nombre_conexión>*. Se desactivará cualquier otra conexión activa que esté ligada al mismo dispositivo.

Para desactivar una conexión,se hará: *nmcli con down <nombre_conexión>*.

Esta activaciones/desactivaciones no son persistentes, el que una conexión esté activa en el arranque del sistema, lo marca el parámetro *connection.autoconnect* a *yes*. Ojo si varias conexiones tienen el *autoconnect* a *yes* y están ligadas al mismo dispositivo, cuál quedará activa será aleatorio.

En el caso de los dispositivos, al hacer un *nmcli dev dis <dispositivo>*, desconectamos el dispositivo y por tanto, desactivamos cualquier conexión ligada a él.

5.5.5.- Archivos de configuración de las interfaces/conexiones

Los archivos de configuración de los interfaces están en */etc/sysconfig/network-scripts* con el nombre *ifcfg-<interfaz>* o *ifcfg-<conexión>*. Los parámetros en estos archivos están configurados uno por línea con el formato de línea de *<parámetro>=<valor>* y da igual su orden. Existen parámetros diferentes dependiendo de si la configuración es estática o dinámica (establecida por DHCP):

Parámetros comunes a ambas configuraciones:

- *DEVICE=<interfaz>*: interfaz de red que usará la interfaz/conexión.
- *NAME=<nombre>*: nombre de la interfaz/conexión.
- *ONBOOT=<yes|no>*: indica si la interfaz/conexión se activará o no en el arranque del sistema.
- *UUID=<uuid>*: identificador asociado a la interfaz/conexión.
- *USERCTL=<yes|no>*: si cualquier usuario puede gestionar la interfaz/conexión o sólo root.
- *HWADDR=<mac>*: dirección MAC del dispositivo asociado al interfaz/conexión.

Parámetros específicos de una conexión estática:

- *BOOTPROTO=none*: indicamos que la interfaz/conexión tiene una configuración estática IPv4.
- *IPV6_AUTOCONF=no*: indicamos que la interfaz/conexión tiene una configuración estática IPv6.
- *IPADDR0=<IP_sistema>*: dirección IPv4 que tiene el sistema. Se puede poner más de una usando los parámetros IPADDR1, IPADDR2, ...
- *PREFIX0=<prefijo>*: prefijo de la subred IPv4 donde está el sistema. Si tiene más de una IP hay que dar el resto de prefijos con PREFIX1, PREFIX2, … Sustituye al parámetro NETMASK.
- *IPV6ADDR=<dir_ipv6>/<prefix>*: direccion IPv6 con su prefijo.
- *DEFROUTE=<yes|no>:* establece esta interfaz/conexión como ruta por defecto o no.
- *GATEWAY0=<gateway>*: dirección del *gateway* IPv4.
- *DNS1=<IP_dns>*: dirección del servidor DNS IPv4 o IPv6.
- *PEERDNS=yes|no*: modifica el */etc/resolv.conf* en el caso de definir DNS1 o no.

Parámetros específicos de una conexión dinámica:

- *BOOTPROTO=dhcp*: indicamos que la interfaz/conexión tome los parámetros IPv4 del servidor DHCP.
- *IPV6_AUTOCONF=no* y *DHCPV6C=yes*: los parámetros IPv6 toman valor por DHCP.

Si se realizan cambios en los archivos de configuración editándolos directamente, luego hay que ejecutar *nmcli con reload* para hacer que Network Manager lea estos cambios. También hay que reiniciar los interfaces para que los cambios entren en funcionamiento, para esto, primero se desactiva y después se activa la conexión asociada a los interfaces con *nmcli con down <conexion>* y *nmcli con up <conexion>*. Si estamos conectados por SSH al sistema, hay que poner ambos comandos en la misma linea separados por **&&** o **;** para no perder la conexión.

NOTA: En el archivo */usr/share/doc/initscripts-*/sysconfig.txt*, en la parte de *network-scripts*, hay información detallada del formato de archivo *ifcfg*.

5.5.6.- Parámetros nmcli vs parámetros ifcfg

nmcli	ifcfg	Descripción
BOOTPROTO=none|dhcp	ipv4.method=manual|auto	Configuración IPv4 estática o por DHCP
IPADDR0=10.11.1.102	ipv4.addresses="10.11.1.102/24 10.11.1.103/24"	Direcciones IP del sistema con sus prefijos
PREFIX0=24		
IPADDR1=10.11.1.103		
PREFIX1=24		
GATEWAY0=10.11.1.254	ipv4.gateway=10.11.1.254	*Gateway* por defecto del sistema
DNS1=10.11.1.254	ipv4.dns=10.11.1.254	Servidor DNS para la interfaz/conexión
ONBOOT=yes|no	connection.autoconnect=yes|no	Si debe iniciar la tarjeta en el arranque del sistema
DEVICE=eth0	connection.interface-name=eth0	Dispositivo de red asociado a la interfaz/conexión
PEERDNS=yes|no	ipv4.ignore-auto-dns=yes|no	Modifica el */etc/resolv.conf* o no
NAME=eth0	connection.id=eth0	Nombre de la conexión
HWADDR=...	802-3-ethernet.mac-address=...	Dirección MAC asociada
IPV6_AUTOCONF=no	ipv6.method=manual	Configuración IPv6 estática
IPV6_AUTOCONF=no DHCPV6C=yes	ipv6.method=dhcp	Configuración de IPv6 por DHCP
IPV6ADDR=<ipv6>/<prefix>	ipv6.addresses="<ipv6>/<prefix>"	Direcciones IPv6
DNS0=...	ipv6.dns=...	DNS IPv6

5.6.- Pasar una configuración dinámica a estática.

Una vez que el sistema ha obtenido los datos que proporciona el servidor DHCP, lo normal es dejar la conexión configurada de forma estática. Para esto seguiremos los pasos:

1.- Obtener los datos relevantes de la conexión dinámica, como son interfaz, IPs, *gateway*, servidores DNS, usando *nmcli con show <nombre_conexión_actual>*. Si el nombre de la conexión lleva espacios, debe ir entre comillas simples o dobles.

2.- Crear una nueva conexión y darle esos parámetros, bien en el comando de creación de la conexión *nmcli con add <nueva_conexión> <param> <valor1> <param2> <valor2>* ...o después de su creación usando *nmcli con mod <nueva_conexion> <parametro> <valor>*.

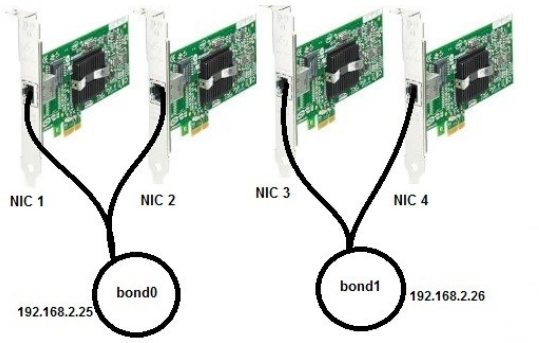
3.- El parámetro *autoconnect* de la conexión actual debe estar a *no* para que en el siguiente arranque del sistema se utilice la nueva conexión con *nmcli con <conexion_actual> connection.autoconnect no*.

4.- Levantar la nueva conexión con *nmcli con up <nueva_conexión>*. Esto deja a *down* la conexión actual. Es mejor no eliminar la conexión actual hasta que estemos seguros de que la nueva conexión está funcionando sin problemas, de esta forma, siempre podremos volver a ella y seguir teniendo red.

5.7.- Teaming

En servidores es habitual de disponer de varias tarjetas de red, con ellas se forma un enlace lógico para proporcionar redundancia, incrementar el rendimiento con un mayor ancho de banda y balanceo de carga. Este concepto se implementó en el kernel y se llamó "*bonding*". En CentOS7, existe una nueva implementación de este concepto y se ha pasado a llamar "*teaming*".

El *driver* de *bonding* existente no se ha visto afectado, el *teaming* no es un reemplazo del *bonding*, simplemente es una alternativa para realizar lo mismo de una forma más eficiente y más extensible gracias al diseño modular del *teaming*.



Teaming o Bonding

En CentOS 7 el *teaming* se implementa proporcionando un pequeño *driver* en el kernel para un manejo más rápido del flujo de paquetes y un demonio en el espacio de usuario, *teamd* que gestiona la lógica y procesamiento de los paquetes.

Este demonio implementa el balanceo de carga y la lógica de active-backup como p.e. *round-robin* usando software llamados *runners*.

Runners disponibles:

- *broadcast*: *runner* simple que transmite cada paquete por todos los puertos.
- *roundrobin*: *runner* simple que transmite los paquetes por turnos por los puertos.
- *activebackup*: *runner* de *failover* que controla si el enlace cambia y selecciona un puerto activo por donde transmitir los paquetes. Sólo se transmite por uno, el resto están de backup.
- *loadbalance*: *runner* que monitoriza el tráfico y usa una función hash para conseguir un balanceo perfecto cuando selecciona un puerto para transmitir los paquetes.

- *lacp*: *runner* que implementa ACP 802.3ad (Link Aggregation Control Protocol), muy parecido a *loadbalance*.

Todas las interacciones con la red se hacen sobre la interfaz de *team*, que está compuesto de varios interfaces que se les llama “de puerto de red”.

5.7.1.- Funcionamiento de teamimg

El comportamiento por defecto de la interacción de la interfaz de *team*, llamado *master* y de los interfaces de puerto, llamados *slaves* es el siguiente:

- Cuando se inicia la interfaz de *team*, no se inician automáticamente los interfaces de puerto pero si iniciamos uno de los interfaces de puerto, sí se iniciará de forma automática la interfaz de *team*.
- Si paramos la interfaz de *team*, se paran todos sus interfaces de puerto.
- Un *master* sin puertos puede iniciar conexiones con IP estáticas o esperar por los puertos que están arrancando conexiones DHCP.

NOTA: El demonio de *teaming*, *teamd* puede no estar instalado por defecto, en tal caso deberemos instalar el paquete *teamd*.

5.7.2.- Configurar teaming

Tenemos varios métodos de configurar el *teaming*:

- Usando la herramienta de interfaz de usuario en modo texto del Network Manager *nmtui.*
- Usando la herramienta en modo comando *nmcli*.
- Utilizando el demonio de *team teamd*: para esto se usan archivos de configuración que se pueden ver en */usr/share/doc/teamd-*/example-configs/* y se activan con *teamd -g -f <fich>.conf -d.*
- Usando archivos de configuración en */etc/sysconfig/network-scripts*
- Con la interfaz gráfica de Network Manager.

En nuestro caso vamos a usar el comando *nmcli* para realizar la configuración del *team*; es el modo más utilizado ya que no siempre disponemos de entorno gráfico en los sistemas o queremos hacer a mano los archivos de configuración.

Los pasos a seguir son:

1. Crear la interfaz de *team*

Añadiremos una conexión con el comando *nmcli* utilizando el tipo *team* con la siguiente sintaxis:

nmcli con add type team con-name <nombre_conexion_team> ifname <nombre_interfaz_team> [conf <conf_json>]

donde <*nombre_conexión*> es el nombre de la conexión que queramos darle a la nueva conexión de *teaming*, <*nombre_interfaz*> es el nombre de la interfaz virtual que se creará y <*conf_json*> es donde se especifica el *runner* a utilizar con la sintaxis:

'{"runner": {"name": "<nombre_runner>"}}' donde <*nombre_runner*> es: *broadcast, roundrobin, activebackup, loadbalance* o *lacp*.

NOTA: Un error muy común es poner mal el nombre del *runner*, p.e., ponerlo con guiones. No aparece ningún error pero no funcionará el *teaming*.

2. Determinar los atributos IPv4 y/o IPv6 de la interfaz de *team*: con el comando *nmcli con mod* <*nombre_conexion*> <*parametro*> <*valor*> se pondrán los distintos atributos de la conexión.

3. Asignar los interfaces de puerto: utilizando *nmcli* crearemos cada interfaz de puerto indicando en tipo *team-slave* y se indica el nombre de la conexión del *master* con la siguiente sintaxis:

nmcli con add type team-slave con-name <nombre_conexion_puerto> ifname <interfaz_puerto> master <nombre_conexion_team>

4. Levantar los interfaces de puerto y de *team*: se levanta la conexión del *team*: *nmcli con up* <*nombre_conexion_team*> y para cada uno de los esclavos: *nmcli con up* <*nombre_conexion_puerto*>.

Cuando la interfaz de *team* está levantado, con el comando *teamdctl* puedo ver el estado del *team* y de los puertos: *teamdctl* <*nombre_conexion_team*> *state*, también con *ip link* podemos ver la activacion del *team* y de los puertos.

Es importante saber cómo quedan los archivos de configuración del *team* y de los puertos en */etc/sysconfig/network-scripts*:

ifcfg-<*nombre_conexión_team*>

```
DEVICE=<nombre_interfaz_team>
DEVICETYPE=Team
TEAM_CONFIG="{\"runner\": {\"<nombre_runner>\"}}"
BOOTPROTO=none
IPADDR0=x.x.x.x
PREFIX0=x
NAME=<nombre_conexión_team>
ONBOOT=yes|no
```

y ifcfg-<nombre-conexión-puerto>:

```
DEVICE=<nombre_interfaz_puerto>
DEVICETYPE=TeamPort
TEAM_MASTER=<nombre_conexión_team>
NAME=<nombre_conexion_puerto>
ONBOOT=yes|no
```

En el caso de problemas, tengo los comandos *teamnl* y *teamdctl* para ver que es lo que está ocurriendo en las conexiones de *team* y de puerto. Incluso se puede sacar una copia de la configuración del *team* con *teamdctl <nombre_conexion_team> config dump* y así ver la información detallada en el terminal o si la vuelco a un archivo, luego usarla en otro sistema usando *nmcli con mod <nombre_conexion_team> team.config <archivo>*.

5.8.- Caso práctico

En **central** y **server1**, se configurará el *hostname* de forma estática.

En **server1**, se va a pasar su configuración de red dinámica a estática creando una conexión llamada *estatica* que tome los parámetros de dirección IPv4, servidor DNS y *gateway* de la configuración dinámica existente en el sistema ahora.

Se le dará a **server1** la dirección IPv6 de *fddb:fe2a:ab1e::c0a8:1/64*.

Desde la máquina física se configuran en **server1** dos tarjetas de red extras en las que se va a configurar *teaming* con ellas: *ens9* y *ens10*. La conexión de *team* se llamará *team1* y las conexiones de los puertos se llamarán *team-slave1* y *team-slave2* y se utilizará *round-robin* para balancear la carga entre ellas. A la interfaz de *team* se le pondrá la IP de *10.11.1.200/24*.

RESOLUCIÓN

- Configuramos en **central** el nombre de *host* de forma estática:

```
[root@central ~]# hostname
central.miempresa.com
[root@central ~]# cat /etc/hostname
localhost.localdomain
[root@central ~]# hostnamectl set-hostname central.miempresa.com
[root@central ~]# cat /etc/hostname
central.miempresa.com
```

- Configuramos en **server1** el *hostname* de forma estática:

```
[root@server1 ~]# hostname
server1.miempresa.com
[root@server1 ~]# cat /etc/hostname
```

localhost.localdomain

[root@server1 ~]# hostnamectl set-hostname server1.miempresa.com

[root@server1 ~]# cat /etc/hostname

server1.miempresa.com

- Configuramos en **server1** la red de forma estática:

[root@server1 ~]# nmcli con show eth0

Vemos que está por DHCP (*ipv4.method* es *auto*) , la dirección IPv4 es *10.11.1.101/24* (*IP4.DNS[1]*), el *gateway* es *10.11.1.254* (*IP4.GATEWAY)*, el servidor DNS es *10.11.1.254* (*IP4.DNS[1]*) y el nombre de dominio es *miempresa.com* (*IP4.DOMAIN[1]*). Con estos datos, creamos la configuración estática en una conexión llamada *estatica*.

[root@server1 ~]# nmcli con add con-name estatica ifname eth0 type ethernet ipv4.addresses 10.11.1.101/24 ipv4.gateway 10.11.1.254 ipv4.dns 10.11.1.254 ipv4.method manual

y evitamos que la conexión actual *eth0* se levante en el arranque:

[root@server1 ~]# nmcli con mod eth0 connection.autoconnect no

- Damos a **server1** la dirección IPv6 de *ddb:fe2a:ab1e::c0a8:1/64* y activo la conexión:

[root@server1 ~]# nmcli con mod estatica ipv6.addresses fddb:fe2a:ab1e::c0a8:1/64

[root@server1 ~]# nmcli con mod estatica ipv6.method manual

[root@server1 ~]# nmcli con up estatica

Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/3)

- Verificamos la configuración:

[root@server1 ~]# ip addr show dev eth0

2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP qlen 1000

link/ether 52:54:00:7f:d6:fc brd ff:ff:ff:ff:ff:ff

inet 10.11.1.101/24 brd 10.11.1.255 scope global eth0

valid_lft forever preferred_lft forever

inet6 fddb:fe2a:ab1e::c0a8:1/64 scope global

valid_lft forever preferred_lft forever

inet6 fe80::e257:feb4:3f6d:49f0/64 scope link

valid_lft forever preferred_lft forever

- Comprobamos que podemos hacer ping:

```
[root@server1 ~]# ping 10.11.1.101
PING 10.11.1.101 (10.11.1.101) 56(84) bytes of data.
64 bytes from 10.11.1.101: icmp_seq=1 ttl=64 time=0.055 ms
64 bytes from 10.11.1.101: icmp_seq=2 ttl=64 time=0.077 ms
^C
--- 10.11.1.101 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.055/0.066/0.077/0.011 ms
[root@server1 ~]# ping -6 fddb:fe2a:ab1e::c0a8:1
PING fddb:fe2a:ab1e::c0a8:1(fddb:fe2a:ab1e::c0a8:1) 56 data bytes
64 bytes from fddb:fe2a:ab1e::c0a8:1: icmp_seq=1 ttl=64 time=0.084 ms
64 bytes from fddb:fe2a:ab1e::c0a8:1: icmp_seq=2 ttl=64 time=0.092 ms
^C
--- fddb:fe2a:ab1e::c0a8:1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 999ms
rtt min/avg/max/mdev = 0.084/0.088/0.092/0.004 ms
```

- Creamos la interfaz de *team* y le damos la IP:

```
[root@server1 ~]# nmcli con add type team con-name team1 ifname team1
                                                    config '{"runner": {"name": "roundrobin"}}'
Connection 'team1' (184c2c55-6152-40e2-97b8-212a7e9df53f) successfully added.
[root@server1 ~]# nmcli con mod team1 ipv4.addresses 10.11.1.200/24
[root@server1 ~]# nmcli con mod team1 ipv4.method manual
```

- Creamos los interfaces de puerto:

```
[root@server1 ~]# nmcli con add type team-slave con-name team-slave1 ifname ens9
                                                                    master team1
Connection 'team-slave1' (ba63e960-0dbb-4b3a-bade-3786e002a64e) successfully added.
[root@server1 ~]# nmcli con add type team-slave con-name team-slave2 ifname ens10
                                                                    master team1
Connection 'team-slave2' (411dab7f-d6a9-40b9-b203-16f1495c7186) successfully added.
```

- Comprobamos cómo está la interfaz de *team*, de momento sin interfaces de puerto:

```
[root@server1 ~]# teamdctl team1 state
setup:
  runner: roundrobin
```

- Levantamos los interfaces de puerto y de *team*:

```
[root@server1 ~]# nmcli con up team-slave1
Connection successfully activated (D-Bus active path:
/org/fredesktop/NetworkManager/ActiveConnection/9)
[root@server1 ~]# nmcli con up team-slave2
Connection successfully activated (D-Bus active path:
/org/fredesktop/NetworkManager/ActiveConnection/10)
```

- Comprobamos ahora la interfaz de *team*:

```
[root@server1 ~]# teamdctl team1 state
setup:
  runner: roundrobin
ports:
  ens9
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
  ens10
    link watches:
      link summary: up
      instance[link_watch_0]:
        name: ethtool
        link: up
        down count: 0
[root@server1 ~]# teamnl team1 ports
 4: ens10: up 100Mbit FD
 3: ens9: up 100Mbit FD
```

- Eliminamos las conexiones que se crearon para los interfaces *ens9* y *ens10* por defecto, llamadas "Wired connection 1" y "Wired connection 2":

```
root@server1 ~]# nmcli con del  Wired\ connection\ 1
Connection 'Wired connection 1' (776c5fc2-ae92-39c3-9f3c-48387a205143)
                                                    successfully deleted.
[root@server1 ~]# nmcli con del Wired\ connection\ 2
Connection 'Wired connection 2' (5a21c537-d51d-3cca-8feb-39d0f132f443)
                                                    successfully deleted.
```

Quedando sólo estas:

```
[root@server1 ~]# nmcli con show
NAME                UUID                                  TYPE            DEVICE
estatica            b31c57c1-45d5-490c-b8bc-119adf120cca  802-3-ethernet  eth0
```

```
team-slave1  ba63e960-0dbb-4b3a-bade-3786e002a64e  802-3-ethernet  ens9
team-slave2  411dab7f-d6a9-40b9-b203-16f1495c7186  802-3-ethernet  ens10
team1        184c2c55-6152-40e2-97b8-212a7e9df53f  team            team1
eth0         ffbec545-27b6-4c8a-87e5-3ad59fd4b301  802-3-ethernet  --
```

- Comprobamos que funciona la interfaz de *team*, desde **central** hago ping a la dirección *10.11.1.200*:

```
[root@central ~]# ping 10.11.1.200
PING 10.11.1.200 (10.11.1.200) 56(84) bytes of data.
64 bytes from 10.11.1.200: icmp_seq=1 ttl=64 time=0.319 ms
64 bytes from 10.11.1.200: icmp_seq=2 ttl=64 time=0.411 ms
```

….