# Comandos útiles

## Vagrant

Vagrant nos permite gestionar máquinas virtuales de un modo muy cómodo. A continuación algunos de los comandos más usados:

```
vagrant up      # levanta la vm
vagrant reload  # recargar la vm
vagrant halt    # apaga la vm
vagrant ssh     # sesión ssh a la vm
vagrant destroy # elimina la vm
```

Para que inicie la máquina virtual en modo gráfico se debe especificar en el Vagrantfile la opción `vb.gui = true`.

### Ejercicio

Pruebe los comandos anteriores parado en el directorio donde se encuentra el archivo Vagrantfile provisto.

## Redes

El amado `ifconfig` ha sido deprecado, ahora se utiliza el comando `ip`.

El querido `netstat` también se esfumó y ahora usamos `ss` (socket statistics).

Los nombres `ethX` también desaparecieron por los fácilmente memorizables `esp02n0x`.

### Nombres de las interfaces

`ls /sys/class/net`

- en01, en02, … andem1, em2,

Firmware-numbered interfaces embedded on the motherboard.

- ens1, ens2, …

Firmware-numbered PCIe hotplug interfaces.

- enp2s0

At PCI bus address 02:00.0.

- p7p1

A card plugged into PCIe slot #7.

### Ejercicio

Examine con el comando ls todas las interfaces de red disponibles.

### Servicio de red

Si bien sigue funcionando `/etc/init.d/network restart` es conveniente usar `systemctl` para consultar su estado, iniciar, detener o reiniciar un servicio del siguiente modo:

```
systemctl [status|stop|start|restart] network.service
```

## Ejercicio

Pruebe reiniciar la red, y luego verifique el estado de la misma utilizando systemctl. Que información del estado se muestra?.

## Direcciones estáticas

Para una interfaz de red que se denomine `enp0s3` la configuración se encuentra en `# vim /etc/sysconfig/network-scripts/ifcfg-enp0s3`. Por ejemplo:

```
DEVICE="enp0s3"
BOOTPROTO=static
ONBOOT=yes
TYPE="Ethernet"
IPADDR=192.168.20.60
NAME="System enp0s3"
HWADDR=00:0C:29:28:FD:4C
GATEWAY=192.168.20.1
```

## Ejercicio

Examine el directorio /etc/sysconfig/network-scripts y comente que se encuentra dentro del mismo.

## ip

Se puede encontrar una comparación entre los comando útiles de ifconfig y ip en https://p5r.uk/blog/2010/ifconfig-ip-comparison.html.

Veamos los más usuales:

```
ip addr add 192.168.50.5 dev eth1       # agregar ip
ip addr show                            # mostrar ip
ip addr del 192.168.50.5/24 dev eth1    # borrar ip
ip link set eth1 up                     # habilitar iface
ip link set eth1 down                   # deshabilitar iface
```

Algunos para rutas:

```
ip route show # muestra ruta
ip route add 10.10.20.0/24 via 192.168.50.100 dev eth0  # agrega ruta
ip route del 10.10.20.0/24                              # borra ruta
ip route add default via 192.168.50.100                 # default gateway
```

Para agregar una ruta estática en forma permanente se debe modificar `/etc/sysconfig/network-scripts/route-eth0` agregándola del siguiente modo:

```
10.10.20.0/24 via 192.168.50.100 dev eth0
```

## Ejercicio

Pruebe agregar una ruta estática de manera temporal, utilizando el comando ´ip´, verifique su creación y luego elimínela. Ahora prueba agregar la misma ruta pero de forma permanente. Reinicie la máquina virtual y verifique que la misma se encuentre seteada.

## Interfaces virtuales

Con el ya casi obsoleto comando `ifconfig` se creaba una interfaz virtual asociada a una real (física) haciendo `ifconfig eth0:0 192.168.1.2 netmask 255.255.255.0 up`. El número luego de los dos puntos la identificaba, y el nombre previo era la placa física a la que estaba asociada.

**Con el comando ip**, se asocia una nueva dirección ip a un dispositivo de red haciendo ``ip address add [ip]/[mask] dev [nic] label [nic]:[name] ``, donde *[ip]/[mask]* hace referencia a la dirección ip y a la máscara de red, *nic* al nombre del dispositivo físico y *[name]* al nombre de interfaz virtual, que en general suele ser un número aunque no está limitado a ello. En el siguiente ejemplo creamos una interfaz virtual asociada a la placa inalámbrica wlp2s:

```
ip address add 10.10.10.47/24 dev wlp2s0 label wlp2s0:1
```

Con lo precedente se agregan ips en forma temporal, si es necesario hacer el cambio permanente se deben crear tantos archivos como interfaces virtuales se requieran en `/etc/sysconfig/network-scripts`, usando la nomeclatura `ifcfg-[nic]:[name]`, donde *[nic]* es el nombre de la interfaz física y *[name]* el número de la interfaz alias.

Por ejemplo, el archivo `/etc/sysconfig/network-scripts/ifcfg-enp0s3:1` tendría algo similar a lo siguiente:

```
DEVICE="enp0s3:1"
BOOTPROTO=static
ONBOOT=yes
TYPE="Ethernet"
IPADDR=10.10.10.66
NETMASK=255.255.255.0
HWADDR=00:0C:29:28:FD:CC
GATEWAY=10.10.10.1
```

Finalmente reiniciar el servicio de red: `systemctl restart network`

## ss

```
ss              # lista todas las conexiones establecidas (tcp/udp/unix)
ss -[tux]       # t tcp, u udp, x unix establecidos
ss -ta          # sockets tcp establecidos y escuchando
ss -tan         # muestra números de puertos e ips en vez de nombres
ss -ltn         # solo escuchando, tcp, números
ss -ltnp        # procesos que abrieron los sockets (sudo)
```

Es interesante utilizar el filtrado basado en direcciones y puertos. Por ejemplo para mostrar todas las conexiones cuyo puerto de origen o destino sean ssh:

```
ss -at '( dport = :ssh or sport = :ssh )'
```

U otro ejemplo donde el número de puerto destino es el 80 o 443:

```
ss -nt '( dst :443 or dst :80 )'
```

Combinado con el comando watch es posible ver en tiempo real las conexiones que se establecen bajo el filtrado previo. Por ejemplo, para ver en tiempo real con un intervalo de 1 segundo el filtrado previo:

```
watch -n1 "ss -nt '( dst :443 or dst :80 )'"
```

## Ejercicio

Deje corriendo el comando anterior en una consola, y pídale a su compañero que desde su equipo se conecte vía ssh al suyo para verificar como se muestran las conexiones (estado, origen, etc).

## dhclient

La bandera `-r` explicitamente libera la asociación actual, por ejemplo:

```
$ sudo dhclient -r
```

Ahora obtenemos una nueva IP:

```
$ sudo dhclient
```

¿Cómo lo renovamos para una interfaz específica, digamos, eth0?

```
$ sudo dhclient -r eth0
$ sudo dhclient eth0
```

# Firewalld

Firewalld es un frontend para iptables que viene por defecto a partir de CentOS 7.

```
systemctl [disable|stop|start|status] firewalld
firewall-cmd --state                                # ver estado
```

# Administrar servicios

*Systemd* es un administrador de sistema y servicios para los sistemas operativos Linux. Está diseñado para mantener compatibilidad con los scripts init de SysV.

*Systemd* introduce el concepto de *unidades* que son representadas por archivos de configuración almacenados en

- `/usr/lib/systemd/system/` creados con la instalación de paquetes RPM
- `/run/systemd/system/` creados en tiempo de ejecución
- `/etc/systemd/system/` creados por `systemctl enable`

que encapsulan información sobre los servicios del sistema, sockets, etc. Para una lista completa sobre los tipos de unidades de systemd vea la Tabla 9.1 "Available systemd Unit Types" (p.99) del *Red Hat Enterprise Linux 7 System Administrator's Guide*.

En versiones previas se utilizaban los scripts *init* que se almacenaban en `/etc/rc.d/init.d` y generalmente eran escritos en Bash y permitian al administrador controlar el estado de los servicios y demonios en el sistema. Bien, ahora estos script han sido reemplazados con los *service units*.

Estos *service units* finalizan con la extensión **.service**. A continuación un resumen de su uso mas frecuente:

```
systemctl [start|stop|restart|status] name.service
systemctl reload name.service
systemctl [enable|disable|is-enabled] name.service

# Displays the status of all services.
systemctl list-units --type service --all
```

```
# Lists all services and checks if they are enabled
systemctl list-unit-files --type service
```

Para más detalles se recomienda la lectura de *CHAPTER 9. MANAGING SERVICES WITH SYSTEMD* de *Red Hat Enterprise Linux 7 System Administrator's Guide*.

# Modos de inicio

Al instalar GNOME o KDE el nivel de ejecución por defecto sigue siendo el modo consola, para cambiar este comportamiento y que automáticamente ingrese al entorno gráfico es necesario hacer:

```
systemctl set-default graphical.target
```

Antes de systemd se modificaba en `/etc/inittab` el nivel de ejecución, ahora se denominan `targets` y se utiliza el comando previo con dos opciones:

- multi-user.target
- graphical.target

Para saber el target en el que se encuentra basta con ejecutar `systemctl get-default`

Al setear un target por defecto lo que se hace es crear un enlace simbólico en `/etc/systemd/system/default.target` apuntando a `graphical.target` o `multi-user.target` en /usr/lib/systemd/system/

# Proxy

Con el fin de economizar tráfico frecuentemente se accede a una red a través de un servidor proxy. Muchos de los comandos que realizamos en el sistema pueden ser redireccionados a través de un proxy con solo configurar la variable de entorno `http_proxy` y `https_proxy`.

Esto se realiza haciendo uso del comando `export` del siguiente modo:

`$export http_proxy="http://PROXY:PUERTO"` y
`$export https_proxy="http://PROXY:PUERTO"`

Se debe tener en cuenta que esto se mantiene siempre y cuando persista el usuario en la misma terminal. Si se desea realizar una acción con permisos de `sudo` entonces primero debe pasarse a administrador con `sudo -s` y finalmente realizar el `export`.

Si se desea habilitar el proxy para YUM en forma permanente entonces se debe modificar el archivo `/etc/yum.conf` agregando una linea que contenga `proxy=http://PROXY:PUERTO`. Para conocer todas las opciones de configuración ver `man yum.conf`.

En los labs de la UTN FRSF el proxy es `frsf.utn.edu.ar` y el puerto `8080`.

# Referencias


- *Red Hat Enterprise Linux 7 System Administrator's Guide*, 2014. D. Brien.