

Unidad 2



Administración de la red en Linux

Administración de sistemas operativos





Índice

2.1. Configuración de la red en Ubuntu. Netplan

- 2.1.1. Ver IP actual en Ubuntu
- 2.1.2. Configurar una IP estática en Ubuntu
- 2.1.3. Configurar una IP dinámica en Ubuntu

2.2. Acceso remoto a Linux

- 2.2.1. Secure Shell. Servidor Secure Shell
- 2.2.2. Servidor VNC

2.3. Seguridad en la red. Iptables

- 2.3.1. Fundamentos de Iptables
- 2.3.2. Instalación de Iptables
- 2.3.3. Definición de reglas
- 2.3.4. Filtrado de paquetes basados en la fuente
- 2.3.5. Eliminación de reglas
- 2.3.6. Cambios persistentes



Introducción

Aunque durante la vida de la informática normalmente los departamentos informáticos de las grandes organizaciones han estado separados en distintos ámbitos, hoy en día lo más recomendable es ser versátil. Comparándolo con esta unidad, debemos de saber que todo administrador de sistemas debe de saber cómo funciona de manera más o menos clara la administración de la red.

Hay ocasiones en la que tendremos los distintos servicios de la red alojados en distintos nodos que se encuentran separados, por lo que debemos de conocer herramientas y utilidades que nos permitan gestionar de manera remota dichos sistemas.

A lo largo de esta unidad hablaremos sobre los tipos de conexión remota que tenemos disponibles además de unas nociones básicas sobre la seguridad en las redes.

Al finalizar esta unidad

- + Sabremos como configurar el servidor DHCP tanto para delimitar y asignar rangos de IP como para asignar IP fijas a determinados equipos.
- + Podremos usar mecanismos de encriptación de la información transferida.
- + Conoceremos como emplear pasarela o túnel SSH para conexiones seguras.
- + Seremos capaces de controlar y filtrar el tráfico de la red con Iptables, describiendo las reglas y aplicando filtros basados en el origen y el destino de los paquetes.
- + Conoceremos como configurar DNS en el servidor
- + Seremos capaces de describir métodos de acceso y administración remota de sistemas, como el acceso remoto por la línea de comandos y el acceso remoto mediante interfaz gráfica.



2.1.

Configuración de la red en Ubuntu. Netplan

Llamamos **Netplan** a la nueva herramienta que desde Ubuntu 17 se usa para la configuración de red en dichos sistemas. Esta herramienta funciona por línea de comandos y se basa en la configuración de las interfaces de red usando ficheros con extensión *.yaml*. Para comunicarse con el *kernel*, *Netplan* está en funcionamiento junto a los demonios *NetworkManager* & *systemd-networkd* de la red.

Para cambiar la configuración de la red, *Netplan* lee la información que se guarda en los ficheros *.yaml* de la carpeta */etc/netplan/* y la lleva a cabo. Las configuraciones de toda la red pueden alojarse en estos archivos, sin tener que realizarla de manera manual por comandos.

```
profesor@universae:~$ ls /etc/netplan/
00-installer-config.yaml
profesor@universae:~$
```

Imagen 1. Directorio */etc/netplan*

2.1.1. Ver IP actual en Ubuntu

Si queremos saber cuáles es la dirección IP actual de nuestro servidor, podemos lanzar el comando **ip a** o **ip addr**.

```
profesor@universae:~$ ip a
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: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:38:46:2f brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 metric 100 brd 10.0.2.255 scope global dynamic enp0s3
        valid_lft 85864sec preferred_lft 85864sec
    inet6 fe80::a00:27ff:fe38:462f/64 scope link
        valid_lft forever preferred_lft forever
3: enp0s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:2e:25:32 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.24/24 brd 10.0.0.255 scope global enp0s0
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:27ff:fe2e:2532/64 scope link
        valid_lft forever preferred_lft forever
profesor@universae:~$ _
```

Imagen 2. Comando **ip a**



2.1.2. Configurar una IP estática en Ubuntu

Como hemos dicho en el apartado anterior, para configurar la red con *netplan*, hay que indicar la configuración de la red en los ficheros alojados en */etc/netplan* y que tengan la extensión *.yaml*.

Para realizar la configuración de una IP estática con *Netplan* hacemos lo siguiente:

1. Lo primero es crear un archivo de configuración con el siguiente comando:

```
sudo nano /etc/netplan/fichero.yaml
```

Es importante que tengamos en cuenta si queremos o no, deshabilitar el que viene por defecto con las opciones de red correspondientes. Si queremos deshabilitarlo, no es recomendable borrarlo, sino simplemente renombrarlo con otra extensión.

2. Una vez en el fichero habrá que editarlo del siguiente modo:

```
network:
  versión: 2
  ethernet:
    Interfaz:
      dhcp4: no
      addresses:
        - Dirección_IP/Máscara
      routes:
        - to: 0.0.0.0
          via: Gateway
      nameservers:
        addresses: [Dirección_IP_Servidor_DNS]
```

```
GNU nano 6.2
network:
  version: 2
  ethernet:
    enp0s9:
      dhcp4: no
      addresses:
        - 192.168.1.10/24
      routes:
        - to: 0.0.0.0
          via: 192.168.1.1
      nameservers:
        addresses: [192.168.1.5]
```

Imagen 3. Fichero de configuración de red



3. Cuando lo hayamos configurado debemos de seguir los siguientes pasos para comprobar su sintaxis y aplicar la configuración:

- a. Lanzamos el comando de comprobación de la sintaxis:

```
sudo netplan try
```

- b. Si todo ha ido bien, lanzamos el comando de aplicación de los cambios:

```
sudo netplan apply
```

- c. Encendemos y habilitamos la resolución de nombres con los siguientes comandos:

```
sudo systemctl start systemd-resolved
```

```
sudo systemctl enable systemd-resolved
```

```
profesor@universae:~$ sudo netplan try
sudo: unable to resolve host universae.lan: Temporary failure in name resolution
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 109 seconds
Configuration accepted.
profesor@universae:~$ sudo netplan apply
profesor@universae:~$ systemctl start systemd-resolved
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-units ====
Authentication is required to start 'systemd-resolved.service'.
Authenticating as: Profesor (profesor)
Password:
==== AUTHENTICATION COMPLETE ====
profesor@universae:~$ systemctl enable systemd-resolved
==== AUTHENTICATING FOR org.freedesktop.systemd1.manage-unit-files ====
Authentication is required to manage system service or unit files.
Authenticating as: Profesor (profesor)
Password:
==== AUTHENTICATION COMPLETE ====
profesor@universae:~$
```

Imagen 4. Comprobación de sintaxis y activación de los servicios

- d. Comprobamos que la IP se ha cambiado correctamente como hemos visto en el apartado anterior

```
4: enp0s9: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:00:27:01:00:60 brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.10/24 brd 192.168.1.255 scope global enp0s9
        valid_lft forever preferred_lft forever
    inet6 fe80::a00:2:ff:fe81:68/64 scope link
        valid_lft forever preferred_lft forever
```

Imagen 5. ip a

MUY IMPORTANTE

Netplan no acepta tabulaciones en sus ficheros de configuración de red



2.1.3. Configurar una IP dinámica en Ubuntu

Para configurar en Ubuntu una IP dinámica, debemos de seguir los siguientes pasos:

1. Lo primero es crear de nuevo otro fichero *.yaml*.
2. Una vez creado, la estructura ahora es la siguiente:

```
network:
  versión: 2
  ethernet:
    Interfaz:
      dhcp4: yes
```

```
network:
  version: 2
  ethernet:
    enp0s9:
      dhcp4: yes
```

Imagen 6. Configuración de IP dinámica con *Netplan*

3. Probamos la sintaxis y aplicamos los cambios.

```
profesor@universae:~$ sudo netplan try
Do you want to keep these settings?

Press ENTER before the timeout to accept the new configuration

Changes will revert in 72 seconds
Configuration accepted.
profesor@universae:~$ sudo netplan apply
profesor@universae:~$ _
```

Imagen 7. Comprobación de la sintaxis y aplicación de cambios

4. La configuración ya se debe de aplicar de manera correcta.





2.2.

Acceso remoto a Linux

Una de las principales tareas de un Administrador de sistemas es la de poder controlar diversos equipos de manera centralizada, es por eso que el *Acceso Remoto* es algo de lo más implementado en servidores. En los sistemas Linux, tenemos varias utilidades que nos permiten trabajar con acceso remoto.

2.2.1. Secure Shell. Servidor Secure Shell

Secure Shell o como se conoce mundialmente, *SSH* es el servicio más conocido para conexiones remotas. Este funciona de manera similar a *telnet*, pero es más seguro debido a que la conexión se encuentra cifrada. Su cifrado impone autenticidad e integridad sobre la información que transmitimos mediante la red. Para autenticarse en sistemas remotos usa el sistema de clave pública. SSH es la más ligera de las herramientas de acceso remoto, y aunque es mediante línea de comandos, es muy sencilla de usar.

Para poder conectarse remotamente a una máquina, en dicha máquina debe de estar instalado el paquete *OpenSSH-Server* que, por lo general, Ubuntu nos indica si queremos instalarlo cuando instalamos el sistema.

La configuración de los parámetros de conexión de SSH se puede realizar en el fichero `/etc/ssh/sshd_config`, donde podemos destacar algunas opciones de configuración:

- > **PermitRootLogin.** Indica si se puede conectar a SSH con el usuario *root* del sistema.
- > **Port.** Nos indica por qué puerto se va a realizar la conexión SSH, por defecto es el 22.
- > **LogincGraceTime.** Nos establece el tiempo límite que tenemos para introducir la contraseña de usuario durante la conexión.
- > **MaxAuthTries.** Establece cuantas veces podemos poner la contraseña errónea antes de proceder a la desconexión.
- > **MaxStartups.** Indica el número máximo de conexiones desde una dirección IP de manera simultánea.
- > **AllowUsers.** Indica que usuarios pueden iniciar sesión por SSH.
- > **DenyUsers.** Indica que usuarios no pueden iniciar sesión por SSH.



```

GNU nano 6.2 /etc/ssh/sshd_config
# $OpenBSD: sshd_config,v 1.103 2018/04/09 20:41:22 fjf Exp $

# This is the sshd server system-wide configuration file. See
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/bin:/bin:/usr/sbin:/sbin

# The strategy used for options in the default sshd_config shipped with
# OpenSSH is to specify options with their default value where
# possible, but leave them commented. Uncommented options override the
# default value.

Include /etc/ssh/sshd_config.d/*.conf

#Port 22
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m

```

Imagen 8. Fichero `/etc/ssh/sshd_config`

Si realizamos cualquiera cambio en la configuración, debemos de lanzar el siguiente comando para reiniciar el servicio:

```
systemctl restart sshd
```

Para poder conectarnos con el cliente a la máquina de manera remota, el paquete que debemos de instalar es *OpenSSH-Client*.

Una vez que lo tenemos todo preparado, podemos conectarnos al servidor de tres maneras diferentes:

- Indicando usuario e IP de conexión de manera separadas, el comando queda:

```
ssh -v -l usuario IP
```

```

profesor@servidor2:~$ ssh -v -l profesor 10.0.0.24
OpenSSH_8.4p1 Ubuntu-6ubuntu2.1, OpenSSL 1.1.1f 24 Aug 2021
debug1: Reading configuration data /etc/ssh/ssh_config
debug1: /etc/ssh/ssh_config line 19: include /etc/ssh/ssh_config.d/*.conf matched no files
debug1: /etc/ssh/ssh_config line 21: Applying options for *
debug1: Connecting to 10.0.0.24 [10.0.0.24] port 22.
debug1: Connection established.
debug1: identity file /home/profesor/.ssh/id_rsa type -1
debug1: identity file /home/profesor/.ssh/id_rsa-cert type -1
debug1: identity file /home/profesor/.ssh/id_dsa type -1
debug1: identity file /home/profesor/.ssh/id_dsa-cert type -1
debug1: identity file /home/profesor/.ssh/id_ecdsa type -1
debug1: identity file /home/profesor/.ssh/id_ecdsa-cert type -1
debug1: identity file /home/profesor/.ssh/id_ecdsa_sk type -1
debug1: identity file /home/profesor/.ssh/id_ecdsa_sk-cert type -1
debug1: identity file /home/profesor/.ssh/id_ed25519 type -1
debug1: identity file /home/profesor/.ssh/id_ed25519-cert type -1
debug1: identity file /home/profesor/.ssh/id_ed25519_sk type -1
debug1: identity file /home/profesor/.ssh/id_ed25519_sk-cert type -1
debug1: identity file /home/profesor/.ssh/id_xmss type -1
debug1: identity file /home/profesor/.ssh/id_xmss-cert type -1
debug1: Local version string SSH-2.0-OpenSSH_8.4p1 Ubuntu-6ubuntu2.1
debug1: Remote protocol version 2.0, remote software version OpenSSH_8.9p1 Ubuntu-3
debug1: match: OpenSSH_8.9p1 Ubuntu-3 pat OpenSSH* compat 0x04000000
debug1: Authenticating to 10.0.0.24:22 as 'profesor'
debug1: SSH2_MSG_KEXINIT sent
debug1: SSH2_MSG_KEXINIT received
debug1: Kex: algorithm: curve25519-sha256
debug1: Kex: host key algorithm: ecdsa-sha2-nistp256
debug1: Kex: server->client cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: Kex: client->server cipher: chacha20-poly1305@openssh.com MAC: <implicit> compression: none
debug1: expecting SSH2_MSG_KEX_ECDH_REPLY
debug1: Server host key: ecdsa-sha2-nistp256 SHA256:0MUVxGmQ233gAXsInV41MG20GUG1naU7MPJH1cJTigY
The authenticity of host '10.0.0.24 (10.0.0.24)' can't be established.
ECDSA key fingerprint is SHA256:0MUVxGmQ233gAXsInV41MG20GUG1naU7MPJH1cJTigY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? _

```

Imagen 9. Conexión por SSH 1



- > Indicando conjuntamente usuario y dirección:

`ssh usuario@IP`

```
profesor@servidor2:~$ ssh profesor@10.0.0.24
The authenticity of host '10.0.0.24 (10.0.0.24)' can't be established.
ECDSA key fingerprint is SHA256:0MUvXGmQ230gAXshNv41HC20GUGinaU7MPJ1icJTigY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.0.24' (ECDSA) to the list of known hosts.
profesor@10.0.0.24's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-41-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of mié 03 ago 2022 08:29:26 CEST

System load:  0.0               Processes:    120
Usage of /:   66.9% of 9.75GB   Users logged in: 1
Memory usage: 17%              IPv4 address for enp0s3: 10.0.2.15
Swap usage:   0%               IPv4 address for enp0s8: 10.0.0.24

 * Super-optimized for small spaces - read how we shrank the memory
   footprint of MicroK8s to make it the smallest full K8s around.

   https://ubuntu.com/blog/microk8s-memory-optimisation

1 update can be applied immediately.
To see these additional updates run: apt list --upgradable

32 updates could not be installed automatically. For more details,
see /var/log/unattended-upgrades/unattended-upgrades.log

*** System restart required ***
You have mail.
Last login: Wed Aug  3 08:17:41 2022
profesor@universae:~$
```

Imagen 10. Conexión por SSH 2

- > Indicando simplemente la dirección, cogerá el usuario por defecto con el que hemos *logueado* anteriormente o nos preguntará por un nuevo usuario, dependiendo de la versión y del sistema:

`ssh IP`

```
profesor@servidor2:~$ ssh 10.0.0.24
profesor@10.0.0.24's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-41-generic x86_64)
```

Imagen 11. Conexión por SSH 3

Si queremos finalizar la conexión SSH, lanzamos el comando `exit`.

```
profesor@universae:~$ exit
logout
Connection to 10.0.0.24 closed.
profesor@servidor2:~$
```

Imagen 12. Fin de la conexión

IMPORTANTE

Cuando realizamos la configuración inicial de SSH en la máquina remota, es importante que se cambien los máximos parámetros posibles que vienen por defecto, para así dotar de mayor seguridad a nuestro sistema.

Si queremos conectarnos de manera gráfica, es decir, para poder ejecutar aplicaciones gráficas de la máquina remota, debemos de añadir la opción `-X`.



Acceso mediante SSH desde Windows

Vamos a ver otro caso que se nos puede dar,

Imaginemos que tenemos un servidor con SSH instalado Linux, pero a nuestro alcance no tenemos ningún cliente Linux para poder efectuar la conexión en este caso lo que tendremos que hacer es conectarnos por un cliente Windows, ya que esto sí que es posible.

Para conectar a servidores Linux usando Windows, tenemos una herramienta llamada *PuTTY*. Dicha herramienta es un descargable que habrá que instalar y mediante el que, si introducimos la dirección IP del servidor, podemos conectarnos.

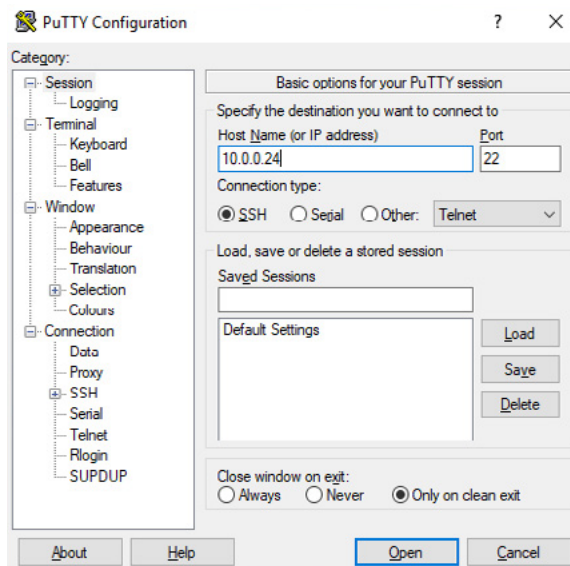


Imagen 13. Conexión por *PuTTY*

Cuando efectuemos la conexión, nos saldrá un mensaje de aviso similar al de los clientes Linux.

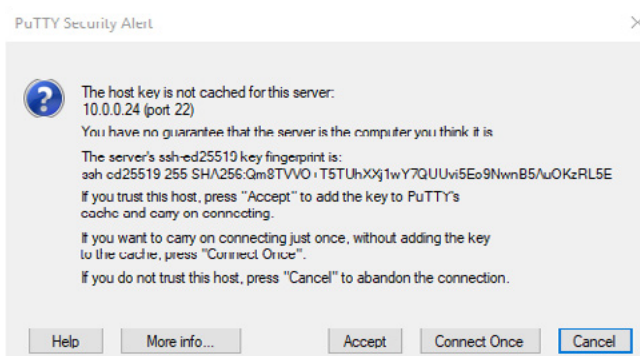


Imagen 14. Mensaje de aviso

Una vez aceptada la conexión, podemos comprobar que todo funciona correctamente y que tenemos una sesión de terminal Linux abierta de manera correcta.

```
login as: profesor
profesor@10.0.0.24's password:
Welcome to Ubuntu 22.04.1 LTS (GNU/Linux 5.15.0-41-generic x86_64)
```

Imagen 15. Terminal en funcionamiento



2.2.2. Servidor VNC

Otra opción de administración remota que se nos presenta es la de VNC. Esta herramienta es un *software* libre que se basa en la arquitectura cliente servidor. Como la función principal de VNC es la de compartir una pantalla con cualquier sistema que admita VNC, es una herramienta multiplataforma.

Instalación de servidor VNC

El paquete que nosotros vamos a instalar para la configuración de nuestro servidor de VNC es *TightVNC* que funciona sobre *Ubuntu Server*. Para ello, instalamos el paquete *tightvncserver*.

Una vez instalado, debemos de crear un usuario específico para la conexión ya añadirlo al grupo de *sudo*, para que pueda ejecutar comandos con dicha orden.

```
profesor@universae:~$ sudo adduser conexionvnc
Adding user `conexionvnc' ...
Adding new group `conexionvnc' (1004) ...
Adding new user `conexionvnc' (1004) with group `conexionvnc' ...
Creating home directory `/home/conexionvnc' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for conexionvnc
Enter the new value, or press ENTER for the default
  Full Name []:
  Room Number []:
  Work Phone []:
  Home Phone []:
  Other []:
Is the information correct? [Y/n]
profesor@universae:~$ sudo gpasswd -a conexionvnc sudo
Adding user conexionvnc to group sudo
profesor@universae:~$ su conexionvnc
Password:
In run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
conexionvnc@universae:/home/profesor$ _
```

Imagen 16. Añadiendo el usuario de conexión

Cuando ya tenemos todo esto hecho, vamos a terminar de configurar el servidor con el comando **vncserver**. La salida de este comando nos indicará que asignemos una contraseña de conexión y, además, nos creará los primeros archivos de configuración.

```
conexionvnc@universae:~$ sudo vncserver

You will require a password to access your desktops.

Password:
Warning: password truncated to the length of 8.
Verify:
Would you like to enter a view-only password (y/n)? n
xauth:  file /root/.Xauthority does not exist

New 'X' desktop is universae.1an:1

Creating default startup script /root/.vnc/xstartup
Starting applications specified in /root/.vnc/xstartup
Log file is /root/.vnc/universae.1an:1.log

conexionvnc@universae:~$
```

Imagen 17. Comando vncserver

Podemos ver que se nos ha solicitado también una contraseña de manera opcional para el acceso de solo vista, a lo que hemos indicado que no, ya que en este momento no es necesario.



Conexión segura VNC

Cuando iniciamos la conexión VNC, no se usa ningún protocolo seguro, por lo que habrá que usar una pasarela SSH para que se establezca primeramente una conexión con el servidor e indicarle al cliente que usemos de conexión que se use dicha pasarela en vez de conectar de manera directa.

Para crear la pasarela segura, debemos de usar el siguiente comando:

```
ssh -L 5901:127.0.0.1:5901 -C -n -l usuario_vnc  
IP_servidor
```

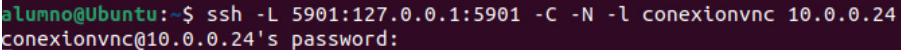


Imagen 18. Pasarela segura mediante SSH

Las distintas opciones que hemos especificado indican lo siguiente:

- > **-L**: indica las vinculaciones del puerto. Aquí lo que hacemos es vincular el puerto 5901 del servidor remoto con el puerto 5901 local, ya que este es el puerto por defecto de VNC.
- > **-C**. Permite la compresión para poder mejorar nuestra conexión remota.
- > **-N**. Indica que no se ejecute el comando remoto a la orden SSH.
- > **-l**. Indica que vamos a especificar el nombre de usuario para la conexión-

Mientras que esta pasarela se encuentra abierta, no conectaremos por terminal, si no que en este momento nos tocará ejecutar el cliente VNC e indicarle que queremos conectar por la dirección *localhost:5901*.



2.3.

Seguridad en la red. Iptables

En una red, como se ha dicho en varias ocasiones es muy importante la seguridad, casi podríamos decir que es el ámbito más importante. Aunque sabemos que los sistemas basados en Unix son los más seguros, hay que tener algún *firewall* en nuestra red que funcione de manera que proteja la red al máximo posible.

En Linux tenemos implementado el *firewall* *Iptables*. Este *software* que ya viene implementado se usa para controlar el tráfico de la red entrante y saliente de algún servidor. Además, se pueden crear reglas para dejar claro que accesos se encuentran autorizados y cuáles no.

2.3.1. Fundamentos de Iptables

En una red, la información siempre se envía a través de paquetes. *Iptables* lo que hace es crear tablas de filtros de estos paquetes para regular el tráfico entrante y saliente.

En cada tabla podemos almacenar distintas cadenas. Nos referimos a cada una como el conjunto de reglas que indican que debe de hacer el sistema con los paquetes que coincidan con las características que se definen en dicha regla.

En nuestro caso concreto, trabajaremos con la tabla filtro (*filter*), que viene definida por defecto, pero se puede modificar. Esta tabla tiene tres cadenas definidas.

- > **INPUT:** es la cadena que controla los paquetes entrantes.
- > **FORWARD:** es la cadena que filtra los paquetes entrantes que se tienen que reenviar.
- > **OUTPUT:** es la cadena que controla los paquetes que salen del servidor.

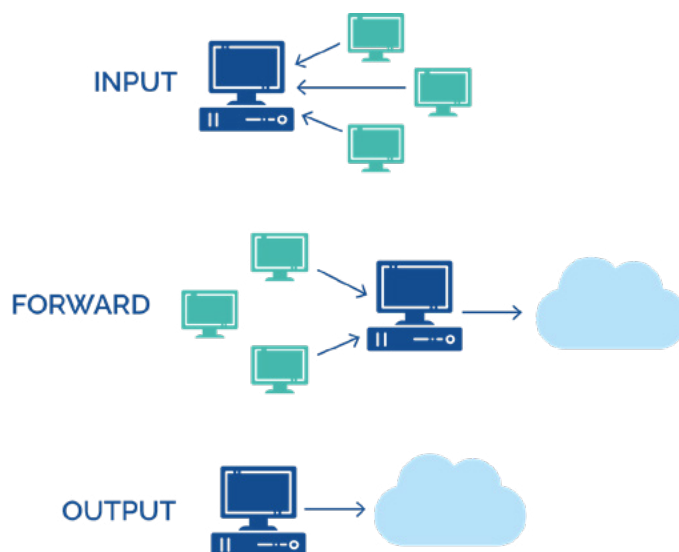


Imagen 19. Tabla de filtros en *Iptables*



2.3.2. Instalación de Iptables

En *Ubuntu Server*, *Iptables* viene por defecto instalado, pero si esto no fuera así, el paquete que debemos instalar se llama igual.

Para ver la configuración actual de *Iptables* podemos lanzar el siguiente comando:

```
iptables -L -v
```

```
profesor@universae:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
profesor@universae:~$ _
```

Imagen 20. Lista de reglas

Las dos opciones que se han especificado se usan para dos propósitos: la opción **-L** nos lista todas las reglas mientras que **-v** hace que la lista que se muestre sea algo más tediosa. Hay que tener muy claro que las opciones de este comando distinguen entre mayúsculas y minúsculas.

Como no hemos configurado aún nada, no hay nada cargado de momento en esta lista.

2.3.3. Definición de reglas

Cuando hablamos de definir una regla nos referimos a añadirla a la cadena. El comando de creación de reglas es del siguiente modo:

```
iptables -A cadena [opciones] -j acción
```

La opción **-A** indica que añadimos una regla a una cadena en específico, mientras que la opción **-j** indica que hay que hacer con todo lo que se filtre por esta regla.

En el siguiente ejemplo permitimos todo el tráfico que provenga de nuestra dirección de *localhost*:

```
profesor@universae:~$ sudo iptables -A INPUT -s 127.0.0.1 -j ACCEPT
profesor@universae:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
    2   140 ACCEPT     all  --  any    any    localhost                anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source                   destination
profesor@universae:~$ _
```

Imagen 21. Regla de entrada en iptables

Podemos ver que se ha indicado la opción **-s**, ya que esta indica cual es el origen, esto lo detallamos más a fondo en el siguiente apartado.



2.3.4. Filtrado de paquetes basados en la fuente

Como hemos adelantado antes, la opción **-s** nos permite indicar cual es el origen de los paquetes, aunque también se puede indicar la interfaz con la opción **-i**.

Vamos a ver esto con ejemplos.

En la siguiente imagen podemos comprobar como aceptamos el tráfico que viene de la dirección **10.0.0.25** y como denegamos el que viene de la dirección **10.0.0.30**. Paso seguido comprobamos que dichas reglas se han añadido.

```
profesor@universae:~$ sudo iptables -A INPUT -s 10.0.0.25 -j ACCEPT
profesor@universae:~$ sudo iptables -A INPUT -s 10.0.0.30 -j DROP
profesor@universae:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
    8  560 ACCEPT     all  --  any    any     localhost      anywhere
    2  145 ACCEPT     all  --  any    any     10.0.0.25      anywhere
    0    0 DROP       all  --  any    any     10.0.0.30      anywhere
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
profesor@universae:~$ _
```

Imagen 21. Reglas de entrada de paquetes de direcciones concretas

Ahora vamos a implementar un rango de direcciones IP, que se puede hacer con la opción **-m** e indicando el rango con la opción **--src-range**.

```
profesor@universae:~$ sudo iptables -A INPUT -m iprange --src-range 10.0.0.30-10.0.0.50 -j ACCEPT
profesor@universae:~$ sudo iptables -L -v
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
   14  998 ACCEPT     all  --  any    any     localhost      anywhere
    2  145 ACCEPT     all  --  any    any     10.0.0.25      anywhere
    0    0 DROP       all  --  any    any     10.0.0.30      anywhere
    0    0 ACCEPT     all  --  any    any     anywhere       anywhere
range 10.0.0.30-10.0.0.50
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source         destination
profesor@universae:~$
```

Imagen 22. Regla de entrada de paquetes de un rango de direcciones

A la hora de la verdad, lo correcto es implementar reglas de aceptación y denegar todo el demás tráfico con el comando que detallamos a continuación:

```
iptables -A INPUT -j DROP
```



2.3.5. Eliminación de reglas

Para eliminar reglas en *iptables*, primero tenemos que visualizar la tablas con los números de cada regla. Esta acción se realiza con el comando:

`iptables -L --line-numbers`

```
profesor@universae:~$ sudo iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target    prot opt source                destination
1    ACCEPT    all  --  localhost             anywhere
2    ACCEPT    all  --  10.0.0.25             anywhere
3    DROP      all  --  10.0.0.30             anywhere
4    ACCEPT    all  --  anywhere              anywhere
50
Chain FORWARD (policy ACCEPT)
num  target    prot opt source                destination
Chain OUTPUT (policy ACCEPT)
num  target    prot opt source                destination
profesor@universae:~$
```

Imagen 23. Lista de reglas con números

Una vez que visualizamos el número de las reglas, podemos eliminarlas ejecutando el siguiente comando:

`iptables -D INPUT nº_regla`

```
profesor@universae:~$ sudo iptables -D INPUT 4
profesor@universae:~$ sudo iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target    prot opt source                destination
1    ACCEPT    all  --  localhost             anywhere
2    ACCEPT    all  --  10.0.0.25             anywhere
3    DROP      all  --  10.0.0.30             anywhere
Chain FORWARD (policy ACCEPT)
num  target    prot opt source                destination
Chain OUTPUT (policy ACCEPT)
num  target    prot opt source                destination
profesor@universae:~$
```

Imagen 24. Eliminar reglas concretas

Para eliminar todas las reglas y comenzar con una configuración inicial desde cero se debe de usar el comando `iptables` con solamente la opción `-F`.

```
profesor@universae:~$ sudo iptables -F
profesor@universae:~$ sudo iptables -L --line-numbers
Chain INPUT (policy ACCEPT)
num  target    prot opt source                destination
Chain FORWARD (policy ACCEPT)
num  target    prot opt source                destination
Chain OUTPUT (policy ACCEPT)
num  target    prot opt source                destination
profesor@universae:~$
```

Imagen 25. Eliminar todas las reglas



2.3.6. Cambios persistentes

Las reglas que se han definido hasta el momento se guardan en la memoria volátil del equipo, por lo que una vez que reiniciemos los cambios no surtirán efecto. Si queremos que los cambios se queden almacenados, el comando que hay que usar es el siguiente:

`sudo /sbin/iptables-save`

```
profesor@universae:~$ sudo /sbin/iptables-save
# Generated by iptables-save v1.8.7 on Wed Aug  3 12:21:08 2022
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -m iprange --src-range 10.0.0.30-10.0.0.50 -j ACCEPT
COMMIT
# Completed on Wed Aug  3 12:21:08 2022
profesor@universae:~$
```

Imagen 26. Guardar las reglas de manera persistente

Dicho comando se encarga de guardar las actuales reglas definidas en el sistema para que una vez que reiniciemos, coja dicha configuración.

Si, por otro lado, queremos que se almacene la configuración sin reglas, deberemos de eliminar todas las reglas y luego guardar la configuración.

```
profesor@universae:~$ sudo iptables -F
profesor@universae:~$ sudo /sbin/iptables-save
# Generated by iptables-save v1.8.7 on Wed Aug  3 12:21:29 2022
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
COMMIT
# Completed on Wed Aug  3 12:21:29 2022
profesor@universae:~$
```

Imagen 27. Eliminar la configuración persistente



 www.universae.com

