# LECCIÓN 1. PRÁCTICA 2

## Configuración de SSH

SSH viene de la contracción **Secure Shell**, aunque en realidad hace referenciia a un protocolo. Gracias a él vamos a tener la posibilidad de:

- Autenticarnos.
- Posibilita seguridad mediante encriptación.
- Un control sobre la integridad de los datos.

Surge ante las debilidades del protocolo telnet.

El supuesto es: vamos a tener siempre un cliente que se conecta a un servidor. Por economia del lenguaje, ssh se refiere tanto al cliente como al servidor, pero en realidad sería: cliente ssh (sshd) y servicio ssh, del lado del servidor.

En la lista de direcorios, vamos a tener un directorio /etc/ssh y dos archivos que modifican el servicio, que no podemos confundir:

- /etc/ssh/sshd_config
- /etc/ssh/ssh_config

## Instalación del servicio ssh

### UBUNTU

1. Creamos una máquina de Ubuntu.
2. Configuramos la red (añadimos un adaptador solo-anfitrión y añadimos una ip asociada a dicha interfaz de red).

   > Para añadir una dirección de red:

   ```
   Nos metemos en el archivo /etc/netplan/00...

   Y lo editamos de esta forma:


       ethernets:
           enp0s3:
           dhcp4: true
           enp0s8:
           addresses:
           - 192.168.56.15/24
       version: 2
   ```

3. Vamos a comunicarnos con los servidores de paquetes con **apt**, para busca **ssh**:

1. Para buscar un paquete concreto:

   ```
   apt search nombre_paquete
   ```

2. Instalamos **openssh-server**

4. ¿El instalador va iniciar el servicio directamente? Para comprobarlo:

   ```
   ps -Af | grep sshd
   ```

5. Podemos hacer una prueba de invocar al cliente y concetarnos con el servidor:

   ```
   ssh localhost
   ```

   Si nos sale un error de que no encuentra el puerto, lo que hemos hecho es activar el sshd con:

   ```
   sudo systemctl start sshd
   ```

6. Una vez arreglado el error, volvemos a hacer ssh localhost y ponemos nuestra contraseña. Ya nos habríamos autenticado; hemos podido acceder a nuestra máquina a través de ssh.

7. Mediante

   ```
   ctrl d
   ```

   hacemos logout y se cierra la conexión al localhost.

8. Comprobamos que se nos ha creado un nuevo directorio en home, **ssh**:

   ```
   ls -la
   ```

   Nos metemos en ese directorio:

   ```
   cd .ssh/
   ```

   Y ahí ya vemos que se ha creado **known_hosts** y que se ha creado el **fingerprint** para localhost(esa cadena de números y letras tan larga que aparece dentro de ese archivo):

   ```
   cat known_hosts
   ```

9. Dentro de las opciones que tenemos que desabilitar, tenemos que ser conscientes de que el usuario que está en todas las máquinas es el usuario **root**. Vamos a editar nuestro archivo de configuración para desabilitar el acceso a root. Cómo podemos acceder remotamente a la administración de nuestro servidor?. Con privilegios de superusuario editaremos el archivo de configuración del servicio:

   ```
   sudo vi /etc/ssh/sshd_config
   ```

Editamos la linea "#PermitRootLogin prohibit-password" y sustituimos esto último por "no"

10. Hacemos:

    ```
    systemctl restart sshd (da igual si ponemos sshd o ssh)(no poner sudo).
    ```

    Nos autenticamos.

11. Nos vamos a acceder de forma remota a nuestra terminal via ssh. Para ello, vamos a abrir una terminal en el host anfitrión y vamos a hacer:

    > ssh -l root 192.168.56.15

Nos autenticamos y ya estariamos dentro. Nosotros hemos tenido problemas con esa dirección: con .1 si funciona. Pero da error: "Permission denied". No tenemos otro mecanismo para acceder.

## ROCKY

1. Instalamos máquina.
2. Rocky instala y activa por defecto el servicio sshd.
3. Si ponemos

   > systemctl status ssh

   Nos va a decir que no encuentra el comando.Mientras que con sshd, sí que lo reconoce. Obsérvese que aquí si que se aprecia la diferencia entre ssh y sshd, a diferencia de en ubuntu.

4. Hacemos la prueba de conexión con localhost:

   > ssh localhost

5. Comprobamos que tenemos el fichero known_hosts (que está en .ssh) con el fingerprint.
6. Cerramos la conexión con

   > ctrl d

7. Ahora intentamos conectarnos via ssh a la máquina virtual (primero con el host: tu_nombre_usuario) desde una terminal del host anfitrión (remotamente):

   > ssh quintinmr@192.168.56.10

   > **ATENCIÓN** porque esa dirección la tenemos nosotros que configurar:

   ```
   1. Tenemos que poner un nuevo adaptador solo anfitrión.

   2. Eliminar la "Wired connection 1":

       nmcli c delete "Wired..."

   3. Añadir la nueva dirección de red a la interfaz enp0s8 (que es la
   nueva, la el solo anfitrión):

       nmcli c add type ethernet con-name enp0s8 ip4 192.168.56.10/24
   ```

8. Accedemos como root de la misma forma. Rocky si permite acceder como root, con password, a diferencia de Ubuntu.

9. Editamos el fichero /etc/ssh/sshd_config y buscamos el PermitRootLogin y lo ponemos a no.

> **Advertencia:** cuando se hace una modificación en la configuración, hay que recargar el servicio: restart.

10. El puerto por defecto de ssh es el 22. Una buena práctica, es cambiar el puerto para que no sea trivial acceder a él. Para modificarlo:

    1. Hacemos uso del comando **sed**:

       ```
       sed s/'Port 22'/'Port 22022'/ -i /etc/ssh/sshd_config
       ```

       Que lo que hace es sustituir una cadena por otra en el archivo que le pasamos por defecto.

11. Probamos otra vez a acceder de forma remota dese la terminal del anfitrión. Observamos que sigue accediendo en el puerto 22. Para especificar un puerto distinto, usamos la opción **-p 22022**. Observamos el error

    ```
    "No route to host"
    ```

12. Para arreglar este problema lo que tenemos que hacer es quitar el "**#**" en "Port 22022".

13. Si hacemos de nuevo **systemctl restart sshd**, nos dará un error.

14. Para saber dónde esta el error, hhacemos uso de la herramienta **journalctl**:

    ```
    journalctl -xe
    ```

15. La solución al problema la tenemos en el mismo archivo de configuración: si nos leemos lo que nos dice la información inicial del archivo, hacemos

    ```
    dnf provides semanage
    ```

16. Hacemos

    ```
    dnf install nombre_archivo_que_nos_sale_en_anterior_llamada_dnf
    ```

17. Hacemos

    ```
    sudo semanage port -l | grep ssh
    ```

18. Hacemos (si olvidamos, mirar /etc/ssh/sshd_config, justo encima del port)

    ```
    sudo semanage port -a -t ssh_port_t -p tcp 22022
    ```

19. Comprobamos que se ha cambiado el puerto con el comando anterior (17); vemos que se ha añadido el puerto como un puerto válido.

20. Si intentamos acceder desde el terminal remoto, con nuestro usuario al puerto 22022, vemos que no es posible todavía.

21. Si nos vamos a la máquina virtual y hacemos en modo normal (no root) ssh localhost -p 22022, vemos que sí nos deja acceder. ¿Qué pieza de software nos permite controlar o cerrar algunos puertos? El **firewall**. Queremos añadir un puerto al firewall:

```
sudo firewall-cmd --add-port 22022/tcp --permanent
```

```
sudo firewall-cmd --add-port 22022/tcp
```

```
sudo firewall-cmd --reload
```

## Arreglamos el problema de Ubuntu

1. Hacemos uso de la herramienta **ufw** para manipular los puertos.
2. Modificamos el valor del puerto. Para ello, nos metemos en el fichero **/etc/ssh/sshd_config** y:
    1. Ponemos el campo Port a 22022
    2. Descomentamos la linea
3. Recargamos el sshd:

    ```
    sudo systemctl restart sshd (sin root)
    ```

4. Comprobamos, desde la terminal remota que tenemos acceso:

    ```
    ssh 192.168.56.15 -l usuario -p 22022
    ```

5. Vemos el estado de **ufw**. Estará inactivo:

    ```
    sudo ufw status
    ```

    Para activarlo:

    ```
    sudo ufw enable
    ```

    Al habilitar el cortafuegos, nos ha cortado la conexión.

6. Tenemos que indicarle al firewall que nos permita el tráfico en el puerto 22022.

    ```
    sudo ufw allow 22022
    ```