Tarea 5.1: SSH y SCP

Descripción de la tarea

Vamos a llevar a cabo el inicio de un servicio en SSH para establecer conexiones entre máquinas y SCP para la compartición de archivos.

Pasos de la tarea

1. Dentro de la misma red y conectado por SSH a los puertos máquina A (2222) y máquina B (2223).

Primero, instalaremos dos maquinas en nuestro software de virtualización.. Vamos a llamarlas **máquina A** y **máquina B**. Ambas maquinas tendrán dos tarjetas de red:

- Una tarjeta estará en modo **NAT**, a través de ella nos conectamos a las maquinas desde la anfitriona y nos descargaremos el software necesario. En esta tarjeta tendremos que configurar el reenvío de puertos en cada una de las maquinas para poder conectarnos desde el anfitrión.
    - Maquina A: puerto anfitrión 2222 y puerto invitado 22.
    - Maquina B: puerto anfitrión 2223 y puerto invitado 22.
- La otra tarjeta de red en **red interna**. Con la que nos comunicaremos entre las dos maquinas virtuales.

```
PS C:\Users\Carballeira\Desktop\curso-linux> ssh -p 2222 usuario@localhost
```

```
PS C:\Users\Carballeira\Desktop\curso-linux> ssh -p 2223 usuario@localhost
```

Nota: En caso de no estar presente el servicio SSH tendrá que ser instalado.

```
usuario@usuario:~$ sudo apt update
usuario@usuario:~$ sudo apt install openssh-server

usuario@usuario:~$ sudo systemctl enable ssh
usuario@usuario:~$ sudo systemctl start ssh
usuario@usuario:~$ systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset:
enabled)
     Active: active (running) since Wed 2025-04-30 08:52:00 CEST; 6min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
   Main PID: 720 (sshd)
      Tasks: 1 (limit: 9438)
     Memory: 6.3M
        CPU: 1.009s
     CGroup: /system.slice/ssh.service
             └─720 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"
```

```
abr 30 08:51:59 usuario systemd[1]: Starting OpenBSD Secure Shell server...
abr 30 08:52:00 usuario sshd[720]: Server listening on 0.0.0.0 port 22.
abr 30 08:52:00 usuario sshd[720]: Server listening on :: port 22.
abr 30 08:52:00 usuario systemd[1]: Started OpenBSD Secure Shell server.
abr 30 08:55:46 usuario sshd[2626]: Accepted password for usuario from 10.0.2.2
port 65043 ssh2
abr 30 08:55:46 usuario sshd[2626]: pam_unix(sshd:session): session opened for
user usuario(uid=1000) by (uid=0)
```

```
sudo dnf install openssh-server
sudo systemctl enable sshd
sudo systemctl start sshd
sudo systemctl status sshd
```

2. En la máquina A vamos a crear el usuario Alex y en B el usuario Brais.

Nota: Previamente puede ser necesario instalar el paquete whois para darle en el mismo comando al usuario una contraseña por defecto.

```
usuario@usuario:~$ sudo apt update
usuario@usuario:~$ sudo apt install whois -y
```

En la máquina A, creamos el usuario *Alex*:

```
usuario@usuario:~$ sudo useradd -m -d /home/alex -s /bin/bash -G sudo -p
'$(mkpasswd "abc123.")' alex
usuario@usuario:~$ tail -n 1 /etc/passwd
alex:x:1001:1001::/home/alex:/bin/bash
```

En la máquina B, creamos el usuario *Brais*:

```
usuario@usuario:~$ sudo useradd -m -d /home/brais -s /bin/bash -G sudo -p
'$(mkpasswd "abc123.")' brais
usuario@usuario:~$ tail -n 1 /etc/passwd
brais:x:1001:1001::/home/brais:/bin/bash
```

A partir de este punto nos convertimos en los usuarios *Alex* y *Brais* para seguir con las pruebas.

```
usuario@usuario:~$ su - alex
alex@usuario:~$ whoami
alex
```

```
usuario@usuario:~$ su - brais
brais@usuario:~$ whoami
brais
```

3. Nuestra máquina A va a actuar como cliente y la máquina B como servidor.

Para ello nuestras máquinas tendrán que estar en un mismo rango ip. Tenemos pues que realizar la configuración pertinente y comprobar si existe ping entre las máquinas. Para ello máquina A será la 192.168.100.2/24 y máquina B será 192.168.100.3/24.

Configuración de máquina A.

```
alex@usuario:~$ sudo ip a a 192.168.100.2/24 dev enp0s8
[sudo] contraseña para alex:
alex@usuario:~$ ip -c 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 08:00:27:83:2d:76 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
       valid_lft 84466sec preferred_lft 84466sec
    inet6 fd00::3209:7c0d:c4a7:e59f/64 scope global temporary dynamic
       valid_lft 86385sec preferred_lft 14385sec
    inet6 fd00::30fb:c695:118e:bbc1/64 scope global dynamic mngtmpaddr
noprefixroute
       valid_lft 86385sec preferred_lft 14385sec
    inet6 fe80::ccd6:e18f:7946:551f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 08:00:27:93:15:96 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.2/24 scope global enp0s8
       valid_lft forever preferred_lft forever
```

Configuración de máquina B.

```
brais@usuario:~$ sudo ip a a 192.168.100.3/24 dev enp0s8
[sudo] contraseña para brais:
brais@usuario:~$ ip -c 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 08:00:27:83:2d:76 brd ff:ff:ff:ff:ff:ff
    inet 10.0.2.15/24 brd 10.0.2.255 scope global dynamic noprefixroute enp0s3
       valid_lft 84788sec preferred_lft 84788sec
    inet6 fd00::7cee:1930:f76:b249/64 scope global temporary dynamic
       valid_lft 86306sec preferred_lft 14306sec
    inet6 fd00::30fb:c695:118e:bbc1/64 scope global dynamic mngtmpaddr
noprefixroute
       valid_lft 86306sec preferred_lft 14306sec
    inet6 fe80::ccd6:e18f:7946:551f/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP
group default qlen 1000
    link/ether 08:00:27:df:fc:d2 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.3/24 scope global enp0s8
       valid_lft forever preferred_lft forever
    inet6 fe80::912f:306a:42cc:3626/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Ping entre máquinas.

```
alex@usuario:~$ ping -c 3 192.168.100.3
PING 192.168.100.3 (192.168.100.3) 56(84) bytes of data.
64 bytes from 192.168.100.3: icmp_seq=1 ttl=64 time=10.5 ms
64 bytes from 192.168.100.3: icmp_seq=2 ttl=64 time=3.35 ms
64 bytes from 192.168.100.3: icmp_seq=3 ttl=64 time=3.24 ms

--- 192.168.100.3 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2005ms
rtt min/avg/max/mdev = 3.244/5.696/10.496/3.394 ms
```

```
brais@usuario:~$ ping -c 3 192.168.100.2
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=64 time=7.87 ms
64 bytes from 192.168.100.2: icmp_seq=2 ttl=64 time=2.29 ms
64 bytes from 192.168.100.2: icmp_seq=3 ttl=64 time=3.80 ms

--- 192.168.100.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 2.287/4.654/7.874/2.359 ms
```

4. Realiza el proceso de conexión de una máquina A hacia B explicando el comando. ¿Qué se crea cuando nos conectamos al servidor desde el cliente?

```
alex@usuario:~$ ssh brais@192.168.100.3
The authenticity of host '192.168.100.3 (192.168.100.3)' can't be established.
ED25519 key fingerprint is SHA256:Ei+jyMSuP40ZKrFLQGIugsjpiOeCu7MvxpJjSl4caEc.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.100.3' (ED25519) to the list of known hosts.
brais@192.168.100.3's password:

brais@usuario:~$
```

Con el comando anterior se realiza la conexión de forma segura desde la máquina A (cliente/Alex) a la máquina B (servidor/Brais) usando el usuario brais. El comando inicia una sesión cifrada para ejecutar comandos remotamente o administrar el servidor. En la primera conexión, el cliente SSH no conoce la identidad del servidor, por eso aparece el mensaje de confirmación de la conexión. Esto significa que el cliente nunca antes se ha conectado a ese servidor y necesita verificar su clave pública para evitar ataques de intermediarios. Si aceptas (escribes "yes"), el cliente almacena la clave pública del servidor en el archivo `~/.ssh/known_hosts` del usuario local. Por otra parte, `~/.ssh/known_hosts.old` es una copia de seguridad automática que se crea cuando modificas el archivo known_hosts, por ejemplo, al eliminar o actualizar una entrada.

5. Crea el directorio "prueba" con el archivo "prueba.txt" en la ruta temporal de A y envía la información al servidor.

```
alex@usuario:~$ mkdir /tmp/prueba && echo "prueba.txt" > /tmp/prueba/prueba.txt &&
cat /tmp/prueba/prueba.txt
prueba.txt

alex@usuario:~$ scp -r /tmp/prueba brais@192.168.100.3:~
brais@192.168.100.3's password:
prueba.txt
100%   11     1.3KB/s   00:00

alex@usuario:~$ ssh brais@192.168.100.3 ls -l
brais@192.168.100.3's password:
total 8
drwxrwxr-x 2 brais brais 4096 abr 30 09:39 prueba
drwx------ 3 brais brais 4096 abr 30 09:27 snap
```

Como podemos ver, creamos el directorio y el archivo en `/tmp/prueba` y lo compartimos con el comando `scp -r /tmp/prueba brais@192.168.100.3:~` que indica que se envía de forma cifrada y recursiva todo el contenido de `/tmp/prueba` al destino `brais@192.168.100.3:~`, es decir, el directorio de trabajo de *brais*.

6. Crea el directorio "prueba2" con el archivo "prueba2.txt" en la ruta temporal del servidor y envíala al cliente.

```
alex@usuario:~$ ssh brais@192.168.100.3
brais@192.168.100.3's password:

brais@usuario:~$
```

```
brais@usuario:~$ mkdir /tmp/prueba2 && echo "prueba2.txt" >
/tmp/prueba2/prueba2.txt && cat /tmp/prueba2/prueba2.txt
prueba2.txt
```

Nota: Aunque estamos dentro del servidor, para no realizar el mismo paso que en el punto anterior, vamos a desconectarnos y realizar el envío de información desde máquina A. Como podemos ver, el origen (máquina B) y destino (máquina A) de información se invierten en este caso.

```
alex@usuario:~$ scp -r brais@192.168.100.3:/tmp/prueba2 /tmp && ls -l /tmp/prueba2
brais@192.168.100.3's password:
prueba2.txt
100%   12     0.6KB/s   00:00
total 4
-rw-rw-r-- 1 alex alex 12 abr 30 09:45 prueba2.txt
```

7. Transmite los directorios "prueba" y "prueba2" a tu ordenador anfitrión al escritorio.

```
PS C:\Users\Carballeira\Desktop> mkdir ejercicioSSH


    Directorio: C:\Users\Carballeira\Desktop


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        30/04/2025      9:46                ejercicioSSH
```

En este caso, mi ordenador anfitrión tiene como Sistema Operativo Windows y para enviar los archivos creo una carpeta llamada `C:\Users\Carballeira\Desktop\ejercicioSSH`. A mayores, es necesario hacer un port forwarding a los puertos correspondientes.

```
PS C:\> scp -P 2222 -r alex@localhost:/tmp/prueba
"C:\Users\Carballeira\Desktop\ejercicioSSH"
alex@localhost's password:
prueba.txt
100%   11     1.3KB/s   00:00
```

```
PS C:\> scp -P 2223 -r brais@localhost:/tmp/prueba2
"C:\Users\Carballeira\Desktop\ejercicioSSH"
brais@localhost's password:
prueba2.txt
100%   12     0.8KB/s   00:00
```

```
PS C:\> dir C:\Users\Carballeira\Desktop\ejercicioSSH


    Directorio: C:\Users\Carballeira\Desktop\ejercicioSSH


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----        30/04/2025      9:49                prueba
d-----        30/04/2025      9:50                prueba2
```

8. Crea el directorio "prueba3" con 200 archivos.txt en el servidor y transmítelo al escritorio del ordenador.

```
PS C:\> ssh -p 2223 brais@localhost 'mkdir /tmp/prueba3 && for i in $(seq 1 200);
do echo "Soy el archivo$i" > /tmp/prueba3/archivo$i.txt; done'
brais@localhost's password:

PS C:\> scp -r -P 2223 brais@localhost:/tmp/prueba3
'C:\Users\Carballeira\Desktop\ejercicioSSH'
brais@localhost's password:
archivo149.txt
100%   18     2.9KB/s   00:00
archivo180.txt
100%   18     2.5KB/s   00:00
...
...
...

PS C:\Users\Carballeira\Desktop\ejercicioSSH\prueba3> dir


    Directorio: C:\Users\Carballeira\Desktop\ejercicioSSH\prueba3


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----        30/04/2025      9:56             16 archivo1.txt
-a----        30/04/2025      9:56             17 archivo10.txt
-a----        30/04/2025      9:56             18 archivo100.txt
-a----        30/04/2025      9:56             18 archivo101.txt
...
```

```
...
...
```

9. Genera un par de claves en el cliente y haz la conexión con el servidor. Crea el passphrase = servidorssh.

Tenemos que generar el par de claves pública-privada en máquina A y compartir la cláve pública a máquina B que actua de servidor. Para generar el par de claves, se utiliza el comando `ssh-keygen` con las siguientes opciones principales:

- `-t` : Especifica el tipo de clave (RSA, DSA, ECDSA, etc.).
- `-b` : Define el tamaño de la clave (ejemplo: 4096 bits).
  - *Nota*: Estas claves solo se usan para autenticación, por lo que aumentar su tamaño no afectará al rendimiento de la CPU durante transferencias SSH.
- `-C ""`: Añade un comentario para identificar la clave (ejemplo: "clave_servidor").
- `-f` : Especifica la ruta y nombre de los archivos de claves (ejemplo: `~/.ssh/clave_personal`).

```
alex@usuario:~$ ssh-keygen -t rsa -b 4096 -C "clave_para_servidor" -f
~/.ssh/clave_servidor
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/alex/.ssh/clave_servidor
Your public key has been saved in /home/alex/.ssh/clave_servidor.pub
...
```

```
alex@usuario:~$ ls -l .ssh/
total 16
-rw------- 1 alex alex 3434 abr 30 10:00 clave_servidor
-rw-r--r-- 1 alex alex  745 abr 30 10:00 clave_servidor.pub
-rw------- 1 alex alex  978 abr 30 09:27 known_hosts
-rw-r--r-- 1 alex alex  142 abr 30 09:27 known_hosts.old
```

Compartimos la clave pública a nuestro servidor con el comando `ssh-copy-id`.

```
alex@usuario:~$ ssh-copy-id -i .ssh/clave_servidor.pub brais@192.168.100.3
```

Realizamos la conexión al servidor, ahora no nos pide la contraseña de *brais* sino el passphrase introducido al generar el par de claves.

```
alex@usuario:~$ ssh brais@192.168.100.3
...
brais@usuario:~$
```