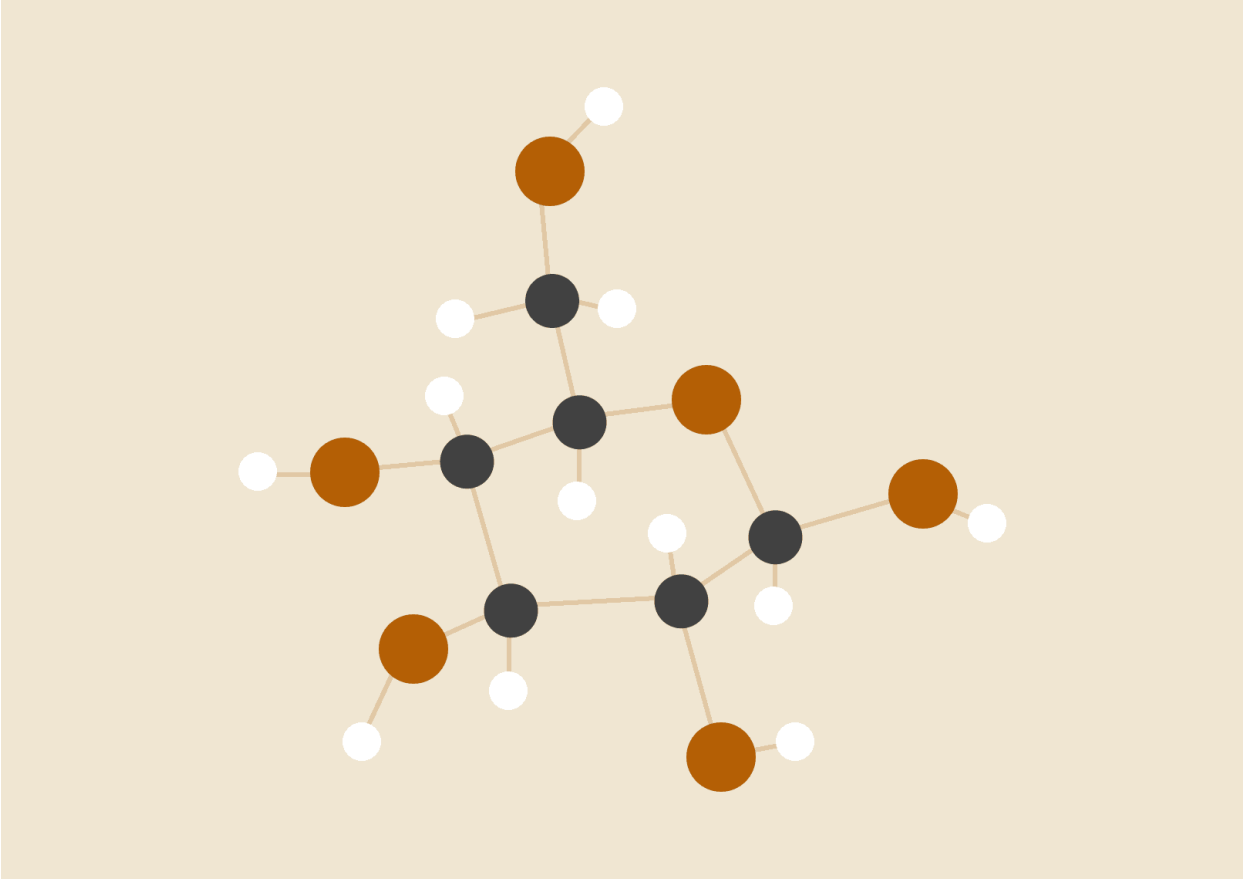
PRÁCTICA DOS

Instalación y configuración del servicio SSH



Marcos García Rodríguez

2.º DAW

Acceso a las instancias

Entramos en AWS y creamos las dos instancias con los conocimientos previos

Nos conectamos a las dos instancias, cada una desde un terminal, con el comando:

- **Servidor**:
  ssh -i "WebDocker2025.pem" ubuntu@ec2-34-230-20-144.compute-1.amazonaws.com

- **Cliente**:
  ssh -i "daweb2025.pem" ubuntu@ec2-18-215-152-152.compute-1.amazonaws.com

E indicamos al servidor que dejamos que instale el fingerprint.

Actualizamos los repositorios:

- **apt update**
- **apt upgrade**

Instalamos el paquete Docker sobre el que trabajaremos en los contenedores.

Creación de los contenedores docker

El servidor con el puerto de escucha 1022

- **Servidor**

```
ubuntu@ip-172-31-23-94:~$ sudo docker run -it --name sshServer -p 1022:22 ubuntu:24.04 /bin/bash
Unable to find image 'ubuntu:24.04' locally
24.04: Pulling from library/ubuntu
```

- **Cliente**

```
ubuntu@ip-172-31-22-120:~$ sudo docker run -it --name sshClient ubuntu:24.04 /bin/bash
Unable to find image 'ubuntu:24.04' locally
24.04: Pulling from library/ubuntu
```

Volvemos a actualizar los repositorios dentro de ambos contenedores.

- **apt install openssh-server**

```
root@4b7cc4b93a20:/# apt install openssh-server
Reading package lists... Done
Building dependency tree... Done
```

Por el contrario, en el cliente es:

- **apt install openssh-client**

```
root@45f1c190ed3d:/# apt install openssh-client
Reading package lists... Done
Building dependency tree... Done
```

Vamos a utilizar los usuarios ubuntu en ambos contenedores, si creamos un usuario nuevo no será posible conectarnos.

Creación de las rutas y claves

Cliente (Donde se crearán las claves)

```
root@45f1c190ed3d:/# su ubuntu
ubuntu@45f1c190ed3d:/$ cd home/ubuntu
ubuntu@45f1c190ed3d:~$ mkdir .ssh
ubuntu@45f1c190ed3d:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .profile  .ssh
ubuntu@45f1c190ed3d:~$
```

En el cliente, procedemos a generar las claves

```
ubuntu@45f1c190ed3d:~/.ssh$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_rsa): certificadoSSH
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in certificadoSSH
Your public key has been saved in certificadoSSH.pub
The key fingerprint is:
SHA256:YQA8f+QJlOxJLHheT4RIFifdqR+ncXDbbamOejwkKfE ubuntu@45f1c190ed3d
The key's randomart image is:
+---[RSA 2048]----+
|   +*O+=..       |
|  ..*oX.* .      |
|   o B Oo+ o . . |
|    . *.B.+ . +  |
|       =SB   o   |
|      . E . .    |
|       . + o     |
|          = .    |
|         .o .    |
+----[SHA256]-----+
```

En el servidor, comprobamos que el servicio está activo. Desde el usuario ROOT

```
root@4b7cc4b93a20:/# service ssh status
 * sshd is not running
root@4b7cc4b93a20:/# service ssh start
 * Starting OpenBSD Secure Shell server sshd                              [ OK ]
root@4b7cc4b93a20:/# service ssh status
 * sshd is running
```

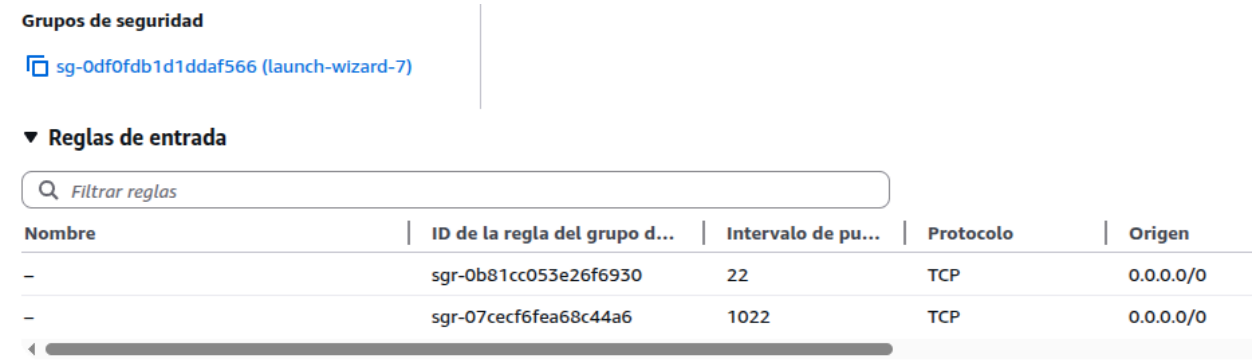
Creamos la carpeta donde almacenaremos la clave en el servidor

```
ubuntu@4b7cc4b93a20:/$ cd home/ubuntu
ubuntu@4b7cc4b93a20:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  .profile
ubuntu@4b7cc4b93a20:~$ mkdir .ssh
ubuntu@4b7cc4b93a20:~$ cd .ssh
ubuntu@4b7cc4b93a20:~/.ssh$ ls -a
.  ..
```

Cambiamos la clave del usuario ubuntu en el servidor por una que conozcamos.

```
root@4b7cc4b93a20:/# passwd ubuntu
New password:
Retype new password:
passwd: password updated successfully
```

En caso de haber utilizado un puerto distinto al 22 al crear el contenedor docker, tendremos que agregar la regla de entrada en AWS para permitir su escucha.



Conexión cliente-servidor

Nos conectamos desde el cliente con el siguiente comando, mandando al servidor la clave pública del cliente.

scp -P 1022 certificadoSSH.pub ubuntu@98.94.6.84:~/.ssh/authorized_keys

```
ubuntu@45f1c190ed3d:~/.ssh$ scp -P 1022 certificadoSSH.pub ubuntu@98.94.6.84:~/.ssh/authorized_keys
The authenticity of host '[98.94.6.84]:1022 ([98.94.6.84]:1022)' can't be established.
ED25519 key fingerprint is SHA256:z1ii3aE1mZ3OxQMLW1gaxC9aAWMQJaEBq8Y8AySLMYA.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '[98.94.6.84]:1022' (ED25519) to the list of known hosts.
```

E introducimos la clave del usuario que acabamos de cambiar.

```
ubuntu@98.94.6.84's password:
certificadoSSH.pub                                    100%  401   640.1KB/s   00:00
```

Como vemos, la clave se transfiere correctamente.

Antes/Después de la carpeta .ssh en el servidor

Antes

```
ubuntu@4b7cc4b93a20:~/.ssh$ ls -a
.  ..
```

Después

```
ubuntu@4b7cc4b93a20:/$ cd /home/ubuntu/.ssh/
ubuntu@4b7cc4b93a20:~/.ssh$ ls -a
.  ..  authorized_keys
```

Nos conectamos desde el cliente indicando el puerto, esta vez en minúsculas.

```
ubuntu@45f1c190ed3d:~/.ssh$ ssh -p 1022 ubuntu@98.94.6.84
ubuntu@98.94.6.84's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
```

Instalación del servicio syslog

Como vemos, estamos con el usuario del servidor, en el servidor, desde la instancia Cliente. Instalamos el servicio Syslog en el cliente desde ROOT

```
root@45f1c190ed3d:/
root@45f1c190ed3d:/# apt install rsyslog -y
Reading package lists... Done
```

Inicializamos el servicio.

```
root@45f1c190ed3d:/# ls /var/log/syslog
ls: cannot access '/var/log/syslog': No such file or directory
root@45f1c190ed3d:/# rsyslogd
rsyslogd: imklog: cannot open kernel log (/proc/kmsg): Operation not permitted.
rsyslogd: activation of module imklog failed [v8.2312.0 try https://www.rsyslog.com/e/2145 ]
root@45f1c190ed3d:/# rsyslogd -n
rsyslogd: pidfile '/run/rsyslogd.pid' and pid 3876 already exist.
If you want to run multiple instances of rsyslog, you need to specify
different pid files for them (-i option).
rsyslogd: run failed with error -3000 (see rsyslog.h or try https://www.rsyslog.com/e/3000 to learn wha
t that number means)
root@45f1c190ed3d:/# ls /var/log/syslog
/var/log/syslog
```

Transferimos el directorio syslog indicando la ruta completa, y el destino. En este caso lo enviaremos directamente a la raíz del usuario ubuntu en el servidor.

```
ubuntu@45f1c190ed3d:/$ scp -P 1022 /var/log/syslog ubuntu@98.94.6.84:~/
ubuntu@98.94.6.84's password:
syslog                                                      100%  602   458.1KB/s   00:00
```

Comprobación final

Y si entramos en el servidor (Desde la propia instancia, o en el servidor), vemos que se ha transferido correctamente.

```
ubuntu@45f1c190ed3d:/$ ssh -p 1022 ubuntu@98.94.6.84
ubuntu@98.94.6.84's password:
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1015-aws x86_64)

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

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.
Last login: Fri Oct 31 14:20:39 2025 from 54.82.124.33
ubuntu@4b7cc4b93a20:~$ ls
syslog
```