

Bastionado de redes y sistemas

Tarea 9: Configuración de sistemas informáticos

Índice

Introducción y objetivos.....	2
Apartado 1.....	3
Bloquear el acceso remoto con el usuario root.....	3
Apartado 2.....	5
Bloquear la sesión si permanece sin actividad durante 5 minutos.....	5
Apartado 3.....	7
Forzar el acceso de los usuarios al servicio con claves públicas/privadas.....	7
Apartado 4.....	10
Limitar el acceso al servicio únicamente para un usuario.....	10
Apartado 5.....	12
Limitar el acceso al servicio para determinadas IPs.....	12
Apartado 6.....	13
Implementar la versión segura del protocolo.....	13
Apartado 7.....	14
Bloquear “Port-forwarding” y ”X11 Forwarding” a través del servicio.....	14
Apartado 8.....	15
Mostrar un mensaje a los usuarios en el inicio de sesión en el que se indique el tipo de sistema al que están accediendo y la obligación de cumplir con la normativa interna de la empresa.....	15
Apartado 9.....	17
Establecer el protocolo de cifrado SHA2-512 o SHA2-256.....	17
Apartado 10.....	19
Activar los logs del servicio para poder enviárselos al SOC.....	19

Introducción y objetivos

El responsable de seguridad de una empresa ha decidido que un servicio crítico y muy utilizado por los administradores debe estar protegido, con el fin de aumentar el nivel de seguridad de las tareas asociadas a la utilización del SSH. Por ello, ha decidido implementar las medidas de seguridad expuestas en cada apartado.

Los objetivos de esta tarea son los siguientes:

- Se han enumerado y eliminado los programas, servicios y protocolos innecesarios que hayan sido instalados por defecto en el sistema.
- Se han configurado las características propias del sistema informático para imposibilitar el acceso ilegítimo mediante técnicas de explotación de procesos.
- Se ha incrementado la seguridad del sistema de administración remoto SSH y otros.
- Se ha instalado y configurado un Sistema de detección de intrusos en un Host (HIDS) en el sistema informático.
- Se han instalado y configurado sistemas de copias de seguridad.

Propuesta de herramientas para su realización:

- **Máquina con sistema operativo Linux:** Centos, Debian, Ubuntu,...
- **Servicio de SSH corriendo en la máquina Linux:** OpenSSH
- **Máquina que permita la conexión al servicio para realizar las pruebas de seguridad.** Se podrá utilizar una máquina asociada al pentesting (ej: Kali Linux) o una herramienta gráfica de un entorno Windows (ej: Putty, MobaXTerm,...)
- **Opcional.** Herramientas de auditoría de seguridad del servicio: nmap, metasploit, hydra, scripts de GitHub...

Se evaluará con 1 punto cada tarea relativa al bastionado. **Deberá aportarse una prueba de que la medida de protección funciona, con una antes y otra después de aplicarla.** Se contará cada apartado como válido si se demuestra la protección a través de los parámetros de configuración del servicio.

Apartado 1

Bloquear el acceso remoto con el usuario root.

Primero tenemos que habilitar el inicio de sesión de `root` con contraseña para poder conectarnos a este usuario mediante SSH. Para ello, ejecutamos los siguientes comandos:

```
# Verifica que el usuario root está en el sistema  
getent passwd root  
  
# Verifica si tiene contraseña establecida  
# Si la salida es root L, login deshabilitado  
# Si la salida es root P, login habilitado  
sudo passwd -S root  
  
# Establece o cambia la contraseña de root  
sudo passwd root
```



López Henestrosa, José Carlos

[Editar perfil](#)

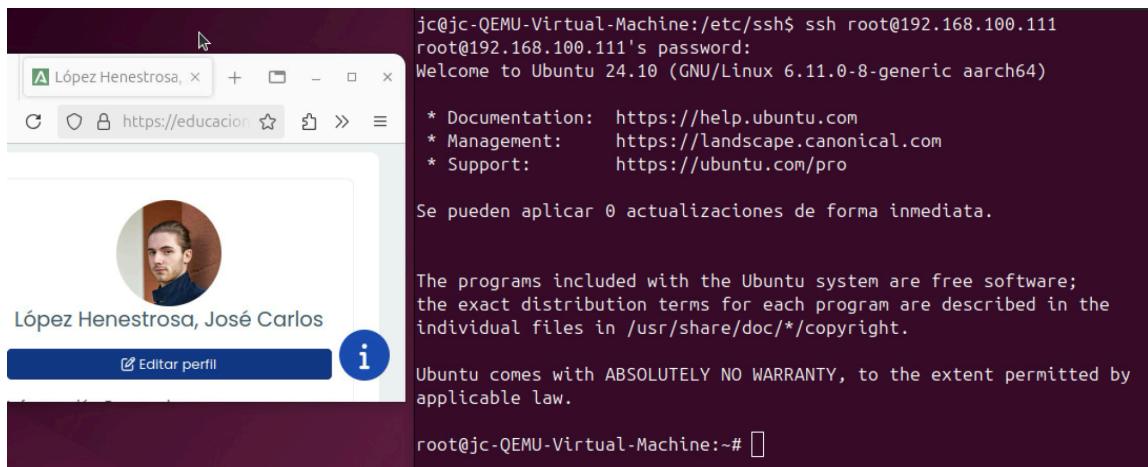


```
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ getent passwd root  
root:x:0:0:root:/root:/bin/bash  
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ sudo passwd -S root  
root L 2024-10-09 0 99999 7 -1  
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ sudo passwd root  
Nueva contraseña:  
Vuelva a escribir la nueva contraseña:  
passwd: contraseña actualizada correctamente
```

Activar login de `root` por contraseña

Ahora probamos a conectarnos mediante SSH al usuario `root`:

```
ssh root@IP_PRIVADA
```

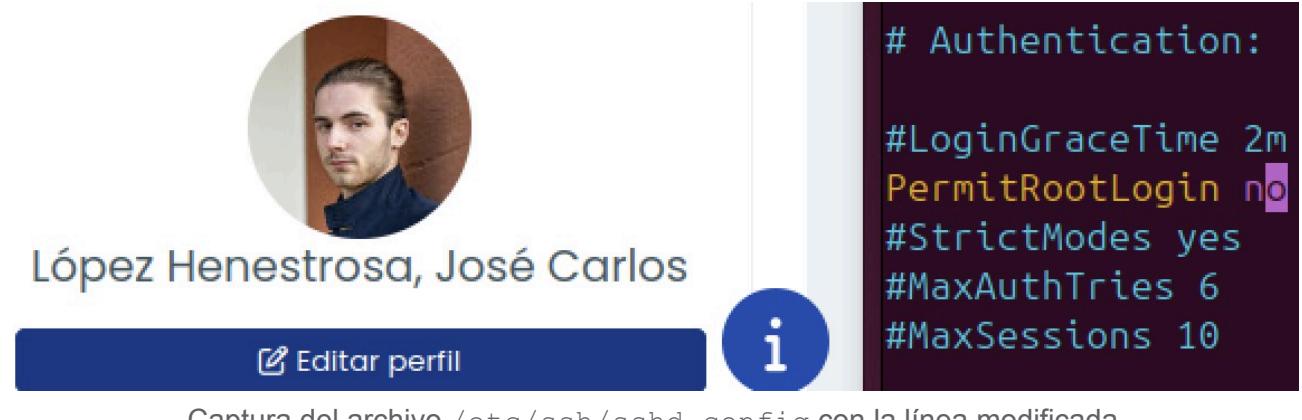


Conexión por SSH correcta

Para bloquear el acceso remoto con el usuario `root`, tenemos que modificar el fichero `/etc/ssh/sshd_config` con privilegios de superusuario, el cual contiene la configuración del servidor SSH. Para ello, utilizamos el siguiente comando:

```
sudo vim /etc/ssh/sshd_config
```

Una vez dentro, tenemos que buscar la línea `PermitRootLogin` y descomentárla. A su vez, tendremos que cambiar el valor por defecto `prohibit-password` por `no` para bloquear el acceso remoto con el usuario `root`.



The screenshot shows a terminal window with the following content:

```
# Authentication:  
#LoginGraceTime 2m  
PermitRootLogin no  
#StrictModes yes  
#MaxAuthTries 6  
#MaxSessions 10
```

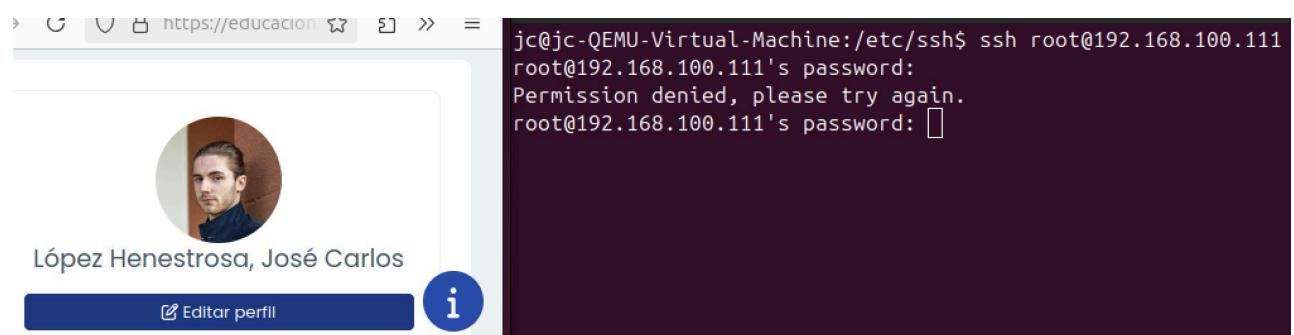
On the left, there is a user profile picture and the name "López Henestrosa, José Carlos". Below the name is a blue button labeled "Editar perfil". On the right, there is a blue circular icon with a white letter "i".

Captura del archivo `/etc/ssh/sshd_config` con la línea modificada

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Si intentamos conectarnos de nuevo al usuario `root` por SSH, veremos que sale el aviso “Permission denied”, lo cual bloquea la conexión con el usuario `root`, por lo que la configuración se ha aplicado correctamente.



The screenshot shows a terminal window with the following content:

```
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ ssh root@192.168.100.111  
root@192.168.100.111's password:  
Permission denied, please try again.  
root@192.168.100.111's password: [REDACTED]
```

On the left, there is a user profile picture and the name "López Henestrosa, José Carlos". Below the name is a blue button labeled "Editar perfil". On the right, there is a blue circular icon with a white letter "i".

Captura del proceso

Apartado 2

Bloquear la sesión si permanece sin actividad durante 5 minutos.

Para conseguir esto, tenemos que tener en cuenta que el proceso varía en función de la versión de ssh que tengamos instalada, ya que a partir de la 9.2 los métodos divergen. En este caso, tengo la versión 9.7 instalada, por lo que tenemos que añadir las directivas `ChannelTimeout` para cerrar automáticamente los canales que no han tenido tráfico en un intervalo de tiempo determinado y `UnusedConnectionTimeout` para terminar las conexiones de cliente que no tienen canales abiertos durante un período de tiempo determinado:



López Henestrosa, José Carlos

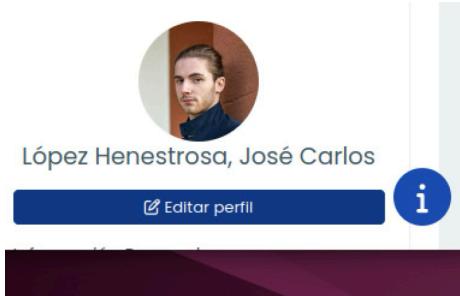
```
#ChrootDirectory none  
#VersionAddendum none  
  
ChannelTimeout *=5m  
UnusedConnectionTimeout 1s
```

Captura del archivo `/etc/ssh/sshd_config` con las directivas añadidas

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Para comprobar que la sesión se cierra correctamente tras el periodo de tiempo detallado, esperamos los 5 minutos, tras los cuales aparece un mensaje que informa de que se ha cerrado la conexión.



```
jc@jc-QEMU-Virtual-Machine:~$ ssh jc@192.168.100.111  
*****  
* Bienvenido al Servidor Ubuntu Corporativo *  
*  
* Este sistema es propiedad de la empresa XYZ. *  
* El acceso está restringido y regulado por la *  
* normativa interna. Cualquier uso indebido será *  
* sancionado.  
*****  
Last login: Sun May 25 20:02:52 2025 from 192.168.100.111  
jc@jc-QEMU-Virtual-Machine:~$ Connection to 192.168.100.111 closed.
```

A pesar de que no se pueda demostrar el proceso fielmente con capturas de pantalla, podemos ver que, tras el periodo de inactividad, la conexión SSH se cierra correctamente.

Fuente:

<https://serverfault.com/questions/1162826/how-to-ensure-that-ssh-drops-the-connection-after-8-hours-of-no-typing>

Apartado 3

Forzar el acceso de los usuarios al servicio con claves públicas/privadas.

Las claves privadas de los usuarios deberán tener configuradas los permisos correctamente para que un usuario no pueda manipular claves privadas de otros usuarios.

Primero, tendremos que desactivar la autenticación mediante contraseña en SSH. Para ello, modificamos el archivo `/etc/ssh/sshd_config` con estas líneas:

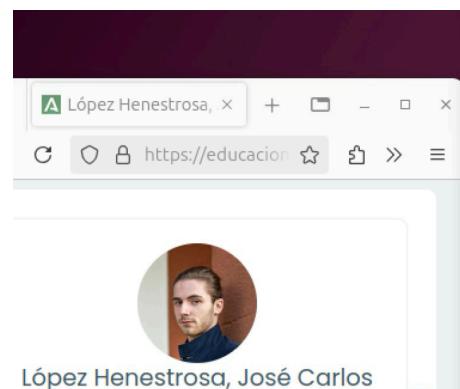
```
PasswordAuthentication no          # valor previo: línea comentada
ChallengeResponseAuthentication no # valor previo: línea comentada
UsePAM no                         # valor previo: yes
PubkeyAuthentication yes            # valor previo: línea comentada
```

Al poner `UsePAM no`, le indicamos al servidor SSH que no utilice Pluggable Authentication Modules y que solo siga los mecanismos básicos de autenticación de OpenSSH, como las claves públicas.

López Henestrosa, José Carlos

```
HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts .
#IgnoreRhosts yes

# To disable tunneled clear text p
PasswordAuthentication no
ChallengeResponseAuthentication no
```



A screenshot of a web browser window. The address bar shows the URL `https://educacion`. Below the address bar, there is a user profile picture of a man with dark hair and a blue background, and the name "López Henestrosa, José Carlos". The main content area of the browser displays the configuration file for `/etc/ssh/sshd_config` with the following content:

```
# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the KbdInteractiveAuthentication and
# PasswordAuthentication. Depending on your PAM configuration,
# PAM authentication via KbdInteractiveAuthentication may bypass
# the setting of "PermitRootLogin prohibit-password".
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and KbdInteractiveAuthentication to 'no'.
UsePAM yes

#AllowAgentForwarding yes
#AllowTcpForwarding yes
#GatewayPorts no
```



López Henestrosa, José Carlos

```
#MaxAuthTries 6  
#MaxSessions 10  
  
PubkeyAuthentication yes  
  
# Expect .ssh/authorized_keys file  
#AuthorizedKeysFile
```

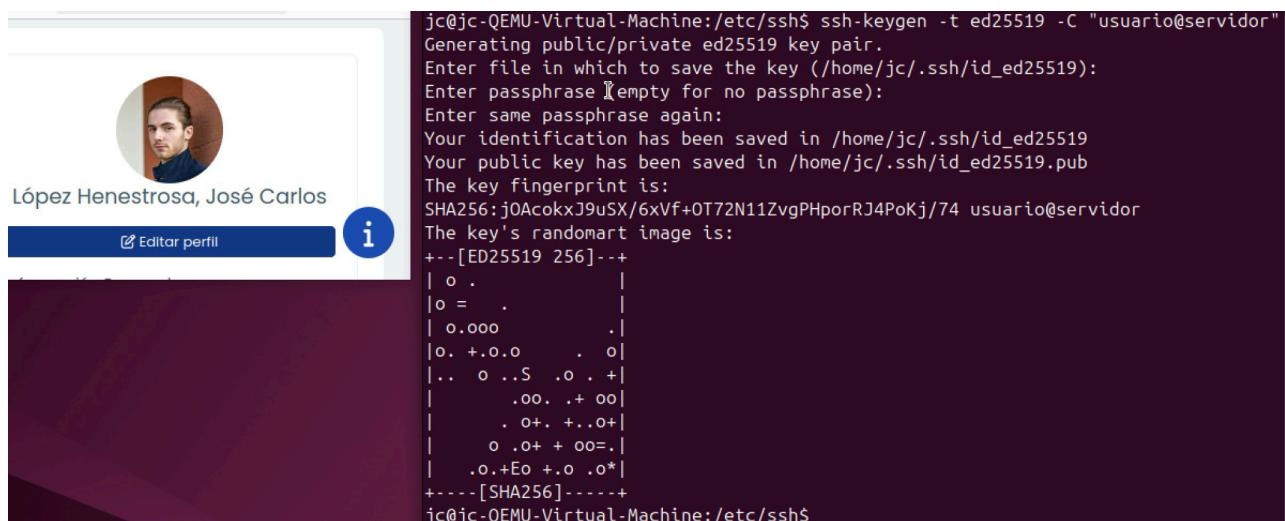
Una vez cambiado el valor de estas directivas, guardamos los cambios y salimos.

Ahora, creamos las claves públicas/privadas para cada usuario que vaya a acceder al servidor. Para ello, usamos el siguiente comando para el usuario jc:

```
ssh-keygen -t ed25519 -C "usuario@servidor"
```

Donde:

- ssh-keygen: Herramienta de Linux para generar claves SSH.
- -t ed25519: Especifica el tipo de clave. En este caso, es ED25519, un algoritmo moderno, seguro y rápido.
- -C "usuario@servidor": Añade un comentario a la clave pública para identificarla fácilmente. No afecta al funcionamiento.



```
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ ssh-keygen -t ed25519 -C "usuario@servidor"  
Generating public/private ed25519 key pair.  
Enter file in which to save the key (/home/jc/.ssh/id_ed25519):  
Enter passphrase (empty for no passphrase):  
Enter same passphrase again:  
Your identification has been saved in /home/jc/.ssh/id_ed25519  
Your public key has been saved in /home/jc/.ssh/id_ed25519.pub  
The key fingerprint is:  
SHA256:joAcokxJ9uSX/6xVf+OT72N11ZvgPHporRJ4PoKj/74 usuario@servidor  
The key's randomart image is:  
+--[ED25519 256]--+  
| o . |  
| o = . |  
| o.ooo .|  
| o. +o.o . o|  
| .. o ..S .o . +|  
| .oo. .+ oo|  
| . o+. +..o+|  
| o .o+ + oo=.|  
| .o.+Eo +.o .o*|  
+---[SHA256]---+  
jc@jc-QEMU-Virtual-Machine:/etc/ssh$
```

Captura del proceso de creación y configuración de las claves públicas/privadas

Al realizar el proceso, se generan tanto la clave privada (`~/.ssh/id_ed25519`) como la clave pública (`~/.ssh/id_ed25519.pub`).

Hecho esto, copiamos la clave pública al servidor con este comando:

```
ssh-copy-id "jc@192.168.100.111"
```

Esto copiará la clave pública al archivo:

```
/home/jc/.ssh/authorized_keys
```

Al ejecutarlo, encontraremos lo siguiente:



```
jc@jc-QEMU-Virtual-Machine:/etc/ssh$ ssh-copy-id "jc@192.168.100.111"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter
out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompt
ed now it is to install the new keys
jc@192.168.100.111's password:

Number of key(s) added: 1

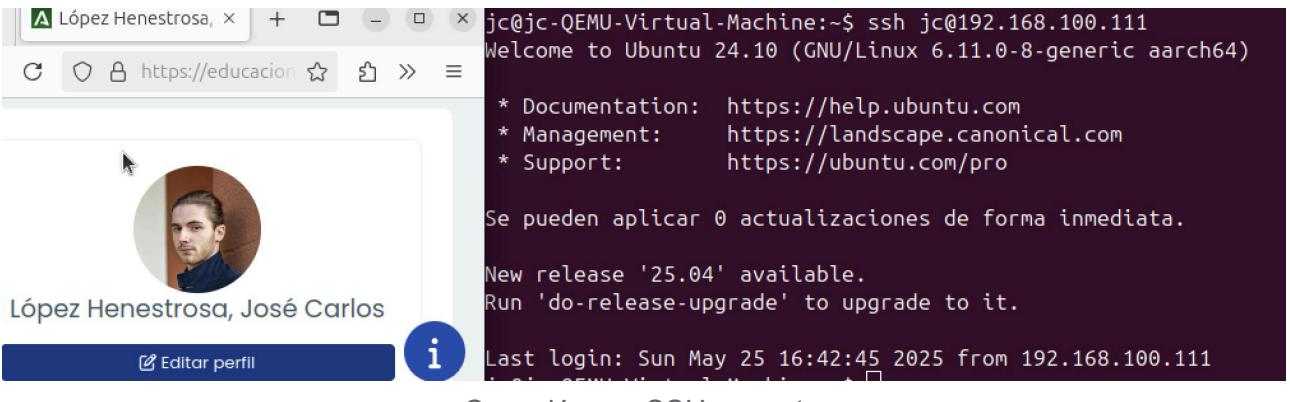
Now try logging into the machine, with:  "ssh 'jc@192.168.100.111'"
and check to make sure that only the key(s) you wanted were added.
```

Copia la clave pública al servidor

Ahora que tenemos todo configurado, reiniciamos el servicio SSH:

```
sudo systemctl restart ssh
```

Y probamos a conectarnos al usuario jc:



```
jc@jc-QEMU-Virtual-Machine:~$ ssh jc@192.168.100.111
Welcome to Ubuntu 24.10 (GNU/Linux 6.11.0-8-generic aarch64)

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

Se pueden aplicar 0 actualizaciones de forma inmediata.

New release '25.04' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun May 25 16:42:45 2025 from 192.168.100.111
```

Conexión por SSH correcta

Apartado 4

Limitar el acceso al servicio únicamente para un usuario.

Para ello, tenemos que modificar el archivo `/etc/ssh/sshd_config` y añadir la directiva `AllowUsers` con el nombre de usuario que queremos permitir. En este caso, será `jc`.

```
AllowUsers jc
```



López Henestrosa, José Carlos

Directiva añadida al archivo `/etc/ssh/sshd_config`

```
#      #  
#      Allow  
#      Permi  
#      Force  
  
AllowUsers jc
```

Tras esto, guardamos los cambios y salimos.

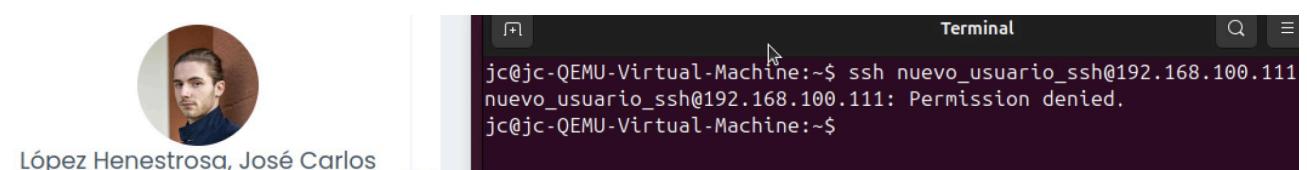
Para comprobar que los cambios se han aplicado correctamente, creamos un nuevo usuario en el sistema:

```
sudo adduser usuario_prueba_ssh
```

Reiniciamos el servicio para que los cambios surtan efecto:

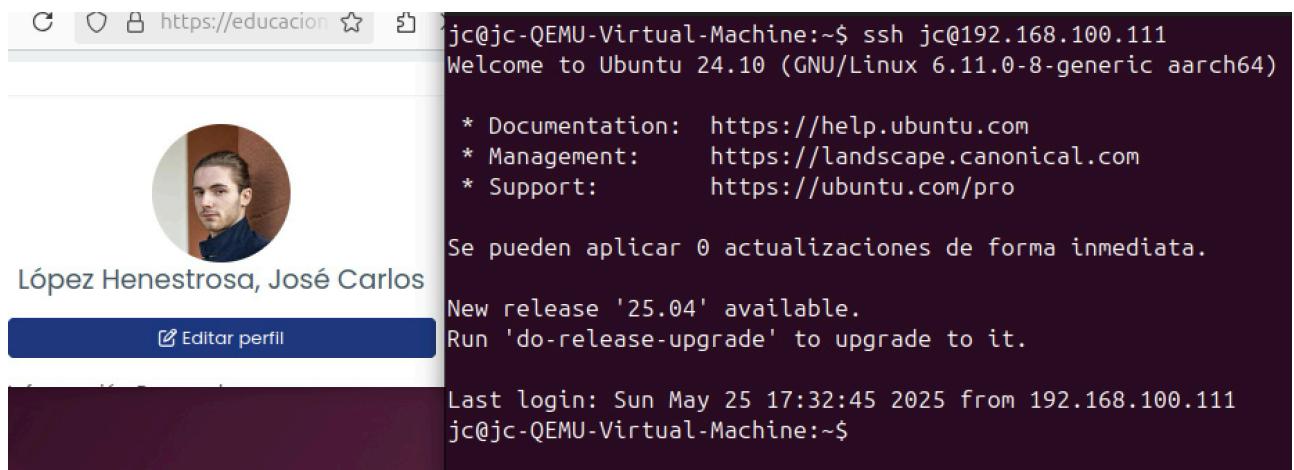
```
sudo systemctl restart ssh
```

Por último, intentamos establecer una conexión SSH con el usuario recién creado:



Prueba fallida de conexión SSH al nuevo usuario

Como podemos comprobar, la conexión es rechazada. En cambio, si intentamos establecer la conexión con el usuario `jc`, veremos que se lleva a cabo correctamente:



A screenshot of a web browser window. On the left, there's a sidebar with a circular profile picture of a man, the name "López Henestrosa, José Carlos", and a blue button labeled "Editar perfil". The main content area shows a terminal session titled "jc@jc-QEMU-Virtual-Machine:~\$ ssh jc@192.168.100.111". The terminal displays the following text:
Welcome to Ubuntu 24.10 (GNU/Linux 6.11.0-8-generic aarch64)

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

Se pueden aplicar 0 actualizaciones de forma inmediata.

New release '25.04' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Sun May 25 17:32:45 2025 from 192.168.100.111
jc@jc-QEMU-Virtual-Machine:~\$

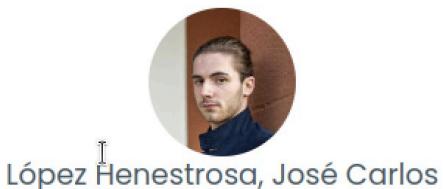
Conexión por SSH correcta

Apartado 5

Limitar el acceso al servicio para determinadas IPs.

Basar la confianza de los equipos en parámetros de filtrado host de SSH en lugar del fichero `.rhosts`.

Para ello, editamos el archivo `/etc/ssh/sshd_config` y añadimos las directivas `Match Address` para restringir el acceso SSH por IP o rango de IP y la directiva `DenyUsers` para bloquear todo por defecto.



```
# Bloquear todo por defecto
DenyUsers *

# Permitir solo conexiones desde IPs específicas
Match Address 192.168.100.111
AllowUsers jc
```

Directiva añadida al archivo `/etc/ssh/sshd_config`

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Para comprobar que la configuración funciona correctamente, probamos a conectarnos por SSH al usuario `jc` desde otra máquina conectada a la misma red. Cabe destacar que se han desactivado temporalmente los cambios realizados en el [apartado 3](#) para verificar esta directiva específicamente:



```
ssh jc@192.168.100.111
macbook@M1-Mac-JC ~ % ssh jc@192.168.100.111
jc@192.168.100.111's password:
Permission denied, please try again.
jc@192.168.100.111's password: █
```

Conexión por SSH desde otra máquina con una dirección IP no autorizada

Como podemos observar, no se permite realizar la conexión. En cambio, si probamos desde la máquina con la IP autorizada, veremos que, efectivamente, se efectúa la conexión correctamente:



```
Terminal
jc@jc-QEMU-Virtual-Machine:~$ ssh jc@192.168.100.111
Last login: Sun May 25 17:54:29 2025 from 192.168.100.111
jc@jc-QEMU-Virtual-Machine:~$ █
```

Conexión por SSH correcta con la dirección IP autorizada

Apartado 6

Implementar la versión segura del protocolo.

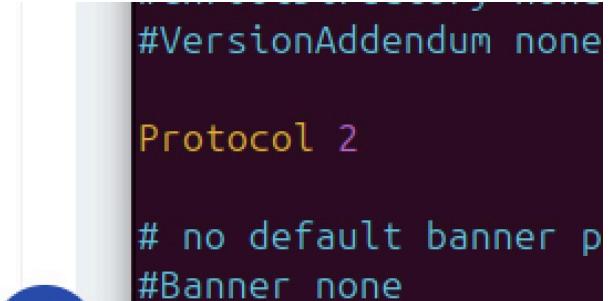
Para esto se debe usar la directiva `Protocol`.

SSH tiene estas dos versiones principales:

- SSH-1 es un protocolo antiguo y vulnerable. Ya no es seguro.
- SSH-2 es la versión moderna, segura y recomendada.

Para implementar la versión segura (SSH-2), tenemos que editar el archivo `/etc/ssh/sshd_config` y buscar la directiva `Protocol`, a la que le tendremos que asignar el valor 2. Esto indica que el servidor SSH solo aceptará conexiones usando el protocolo SSH versión 2.

López Henestrosa, José Carlos



```
#VersionAddendum none
Protocol 2
# no default banner p
#Banner none
```

Directiva añadida al archivo `/etc/ssh/sshd_config`

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Apartado 7

Bloquear “Port-forwarding” y “X11 Forwarding” a través del servicio.

Para probarlo, debe usarse el parámetro de `ssh -X`. Con algunos clientes de SSH como MobaXterm puede darnos idea de dicha configuración.

Para ello, tenemos que modificar el archivo de configuración `/etc/ssh/sshd_config` y añadir estas líneas para desactivar el reenvío:

```
AllowTcpForwarding no          # valor previo: línea comentada  
X11Forwarding no             # valor previo: yes
```



López Henestrosa, José Carlos

```
#AllowAgentForwarding yes  
AllowTcpForwarding no  
#GatewayPorts no  
X11Forwarding no  
#X11DisplayOffset 10
```

Directivas añadidas al archivo `/etc/ssh/sshd_config`

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Para comprobar que está bloqueado el X11 Forwarding, ejecutamos el siguiente comando:

```
ssh -X jc@192.168.100.111
```



López Henestrosa, José Carlos

```
jc@jc-QEMU-Virtual-Machine:~$ ssh -X jc@192.168.100.111  
X11 forwarding request failed on channel 0  
Last login: Sun May 25 18:50:26 2025 from 192.168.100.111  
jc@jc-QEMU-Virtual-Machine:~$ 
```

Conexión por SSH correcta con el aviso de que la petición X11 forwarding falló, lo cual es correcto

Apartado 8

Mostrar un mensaje a los usuarios en el inicio de sesión en el que se indique el tipo de sistema al que están accediendo y la obligación de cumplir con la normativa interna de la empresa.

Directiva `Banner`.

En primer lugar, crearemos un archivo (`/etc/ssh/ssh-banner.txt`) con el mensaje que queremos mostrar, que en este caso será el siguiente:

```
*****  
* Bienvenido al Servidor Ubuntu Corporativo      *  
*                                              *  
* Este sistema es propiedad de la empresa XYZ.    *  
* El acceso está restringido y regulado por la     *  
* normativa interna. Cualquier uso indebido será    *  
* sancionado.                                     *  
*****
```

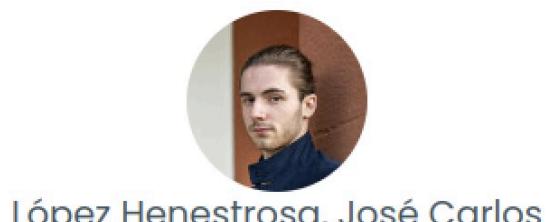


```
*****  
* Bienvenido al Servidor Ubuntu Corporativo      *  
*                                              *  
* Este sistema es propiedad de la empresa XYZ.    *  
* El acceso está restringido y regulado por la     *  
* normativa interna. Cualquier uso indebido será    *  
* sancionado.                                     *  
*****
```

Fichero `/etc/ssh/ssh-banner.txt` con el mensaje especificado anteriormente

Guardamos y cerramos el archivo.

A continuación, configuramos la directiva `Banner` en el fichero de configuración del servidor SSH `/etc/ssh/sshd_config`:

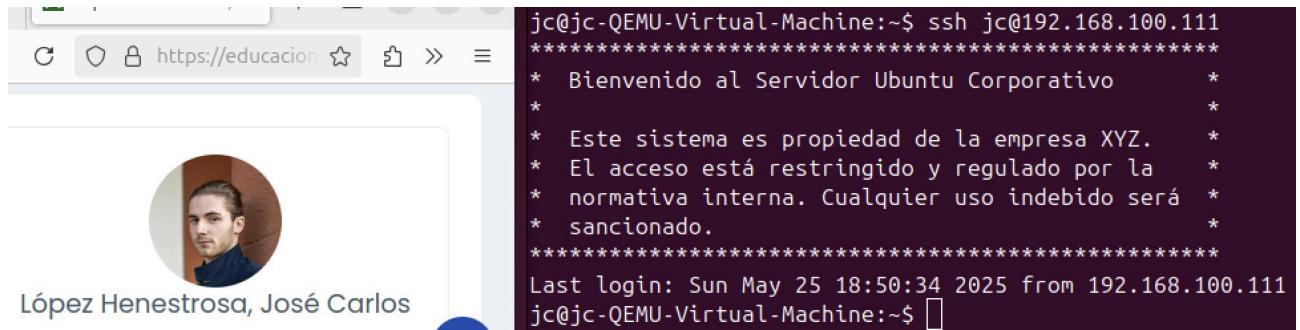


```
#versionAddendum none  
Protocol 2  
# no default banner path  
Banner /etc/ssh/ssh-banner.txt
```

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Para comprobar que aparece el mensaje especificado, iniciamos una sesión SSH:



```
jc@jc-QEMU-Virtual-Machine:~$ ssh jc@192.168.100.111
*****
* Bienvenido al Servidor Ubuntu Corporativo      *
*
* Este sistema es propiedad de la empresa XYZ.   *
* El acceso está restringido y regulado por la   *
* normativa interna. Cualquier uso indebido será  *
* sancionado.                                     *
*****
Last login: Sun May 25 18:50:34 2025 from 192.168.100.111
jc@jc-QEMU-Virtual-Machine:~$ 
```

Conexión SSH en el que aparece el mensaje configurado en la directiva Banner

Apartado 9

Establecer el protocolo de cifrado SHA2-512 o SHA2-256.

Para probar esto, podemos usar el comando `nmap --script ssh2-enum-algos -sV -p 22 localhost`.

Para usar uno de estos protocolos de cifrado, tenemos que configurar el archivo `/etc/ssh/sshd_config` y añadir las siguientes líneas para forzar el uso de algoritmos modernos con SHA2:

```
# Algoritmos de intercambio de claves
KexAlgorithms
curve25519-sha256@libssh.org,diffie-hellman-group-exchange-sha256

# Algoritmos de cifrado simétrico
Ciphers aes256-ctr

# Algoritmos de integridad con SHA2-512
MACs hmac-sha2-512
```

López Henestrosa, José Carlos

```
# Intercambio de claves
KexAlgorithms curve25519-sha256@libssh.org,diffie-hellman-group-exchange-sha256

# Algoritmos de cifrado simétrico (ciphers)
Ciphers aes256-ctr

# Algoritmos de integridad (MACs) con SHA2-512
MACs hmac-sha2-512
```

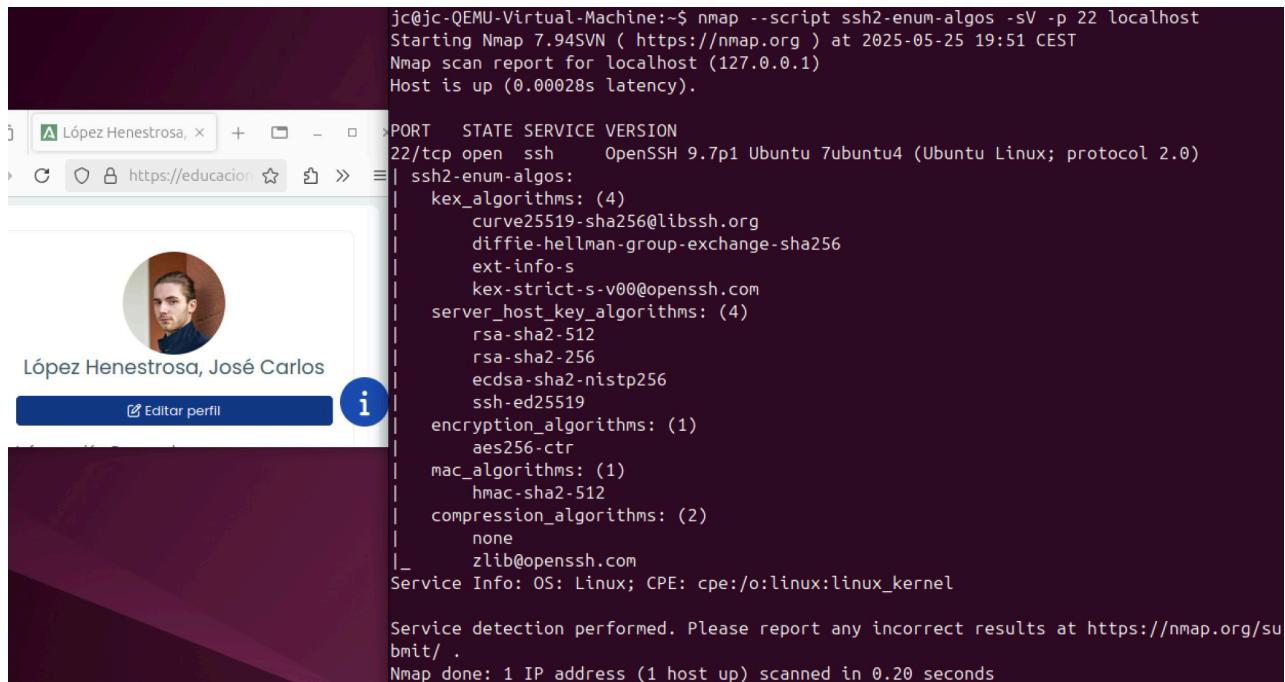
Directivas añadidas al archivo `/etc/ssh/sshd_config`

Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Por último, comprobamos los algoritmos soportados con `nmap` ejecutando este comando:

```
nmap --script ssh2-enum-algos -sV -p 22 localhost
```



```
jc@jc-QEMU-Virtual-Machine:~$ nmap --script ssh2-enum-algos -sV -p 22 localhost
Starting Nmap 7.94SVN ( https://nmap.org ) at 2025-05-25 19:51 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.00028s latency).

PORT      STATE SERVICE VERSION
22/tcp    open  ssh      OpenSSH 9.7p1 Ubuntu 7ubuntu4 (Ubuntu Linux; protocol 2.0)
| ssh2-enum-algos:
|   kex_algorithms: (4)
|     curve25519-sha256@libssh.org
|     diffie-hellman-group-exchange-sha256
|     ext-info-s
|     kex-strict-s-v00@openssh.com
|   server_host_key_algorithms: (4)
|     rsa-sha2-512
|     rsa-sha2-256
|     ecdsa-sha2-nistp256
|     ssh-ed25519
|   encryption_algorithms: (1)
|     aes256-ctr
|   mac_algorithms: (1)
|     hmac-sha2-512
|   compression_algorithms: (2)
|     none
|     zlib@openssh.com
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel

Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 0.20 seconds
```

Rastreo realizado con nmap

Con este comando, se listan los algoritmos soportados por el servidor SSH. Como podemos comprobar, todos los algoritmos configurados aparecen correctamente en el resumen del escaneo realizado por nmap.

Apartado 10

Activar los logs del servicio para poder enviárselos al SOC.

Para ello, configuraremos la directiva `LogLevel` en el archivo `/etc/ssh/sshd_config`. Hay que destacar que hay distintos niveles posibles de log, los cuales son los siguientes:

- QUIET
- FATAL
- ERROR
- INFO (por defecto)
- VERBOSE
- DEBUG

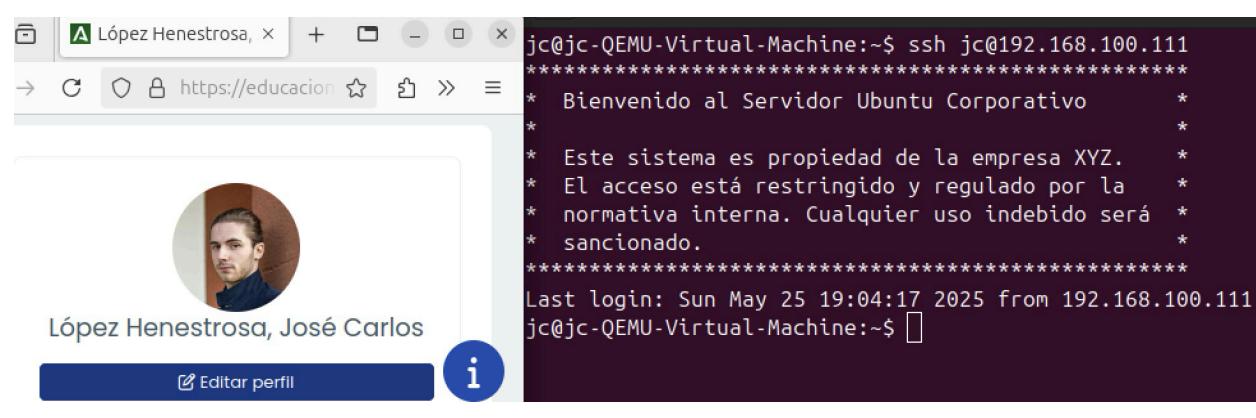
En este caso, vamos a configurarlo con el nivel **VERBOSE**:



Tras esto, guardamos los cambios, salimos y reiniciamos el servicio:

```
sudo systemctl restart ssh
```

Por defecto en Ubuntu, los logs de SSH se almacenan en el fichero `/var/log/auth.log`. Para comprobar que se están guardando correctamente, probamos a iniciar sesión por SSH:



A continuación, revisamos dicho fichero:



The screenshot shows a web browser window with a user profile for 'López Henestrosa, José Carlos'. Below the profile, there is a blue button labeled 'Editar perfil' and a small circular icon with the letter 'i'. To the right of the browser window is a terminal window displaying log entries from the file /var/log/auth.log. The log entries are as follows:

```
key for jc from 192.168.100.111 port 33996 ssh2 [preauth]
2025-05-25T20:02:52.268906+02:00 jc-QEMU-Virtual-Machine sshd[13059]: Accepted key ED25
519 SHA256:j0AcokxJ9uSX/6xVf+OT72N11ZvgPHporRJ4PoKj/74 found at /home/jc/.ssh/authorize
d_keys:1
2025-05-25T20:02:52.272474+02:00 jc-QEMU-Virtual-Machine sshd[13059]: Accepted publicke
y for jc from 192.168.100.111 port 33996 ssh2: ED25519 SHA256:j0AcokxJ9uSX/6xVf+OT72N11
ZvgPHporRJ4PoKj/74
2025-05-25T20:02:52.276451+02:00 jc-QEMU-Virtual-Machine sshd[13059]: User child is on
pid 13062
2025-05-25T20:02:52.319003+02:00 jc-QEMU-Virtual-Machine sshd[13062]: Starting session:
shell on pts/1 for jc from 192.168.100.111 port 33996 id 0
2025-05-25T20:04:07.820043+02:00 jc-QEMU-Virtual-Machine sudo:      jc : TTY=pts/1 ; P
WD=/home/jc ; USER=root ; COMMAND=/usr/bin/vim /var/log/auth.log
2025-05-25T20:04:07.825968+02:00 jc-QEMU-Virtual-Machine sudo: pam_unix(sudo:session):
session opened for user root(uid=0) by jc(uid=1000)
```

Contenido del archivo /var/log/auth.log

Como podemos comprobar, los logs se almacenan correctamente con el nivel indicado en la configuración del servicio SSH.