UD 05: Administración remota del sistema. Actividades

CUESTIONES

1.- Indica diferentes métodos de acceso y administración remota de sistemas.

SSH (Secure Shell):

Proporciona cifrado de extremo a extremo y autenticación sólida, lo que lo convierte en una opción segura para acceder a sistemas remotos, especialmente en entornos de servidores Linux y Unix. Esta no permite interfaz gráfica.

RDP (Remote Desktop Protocol):

Permite acceder y controlar un ordenador Windows con interfaz gráfica de forma remota. Es comúnmente utilizado en entornos empresariales para administrar servidores y estaciones de trabajo Windows.

VNC (Virtual Network Computing): VNC permite controlar y ver la interfaz gráfica de un sistema de forma remota. Se puede utilizar para sistemas operativos que tienen entorno gráfico, como Windows, Linux o macOS.

Otras Aplicaciones Remote Desktop:

Como podría ser TeamViewer, AnyDesk, Iperius remote. Que se basan en protocolos propietario, permiten la conexión, control remoto de dispositivos, compartir ficheros, tener un chat etc.

Telnet:

Telnet permite la administración remota a través de la línea de comandos, pero debido a problemas de seguridad, se recomienda utilizar SSH en su lugar.

Consolas web de administración:

Muchos dispositivos de red y servidores ofrecen interfaces web gráficas para la administración remota. Estas consolas web permiten realizar configuraciones y monitoreo a través de un navegador.

2.- Concepto de sesión. Diferencia entre servicios orientados a sesión y los no orientados a sesión. Ejemplo de cada uno de ellos.

Concepto sesión

Una sesión es un periodo de interacción entre dos dispositivos durante el cual se establece una conexión lógica para intercambiar datos.

Diferencia entre servicios orientados a sesión y los no orientados a sesión

Servicio orientado a sesión: Tiene la función de iniciar sesión remotamente en un equipo con usuario local o del dominio.

Ejemplo orientado a sesión

Un ejemplo sería el control remoto de Windows Server, que al iniciar sesión remotamente se cierra la sesión del equipo al que se ha accedido.

Servicio no orientado a sesión: Este servicio se conecta remotamente al equipo al que queremos administrar, para ello se utiliza la sesión que está abierta en ese momento.

Ejemplo no orientado a sesión

Un ejemplo seria la utilización de Teamweaver

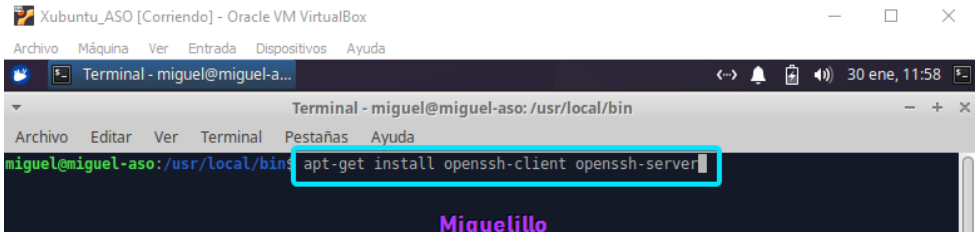
UD 05: Administración remota del sistema. Actividades

PRÁCTICAS

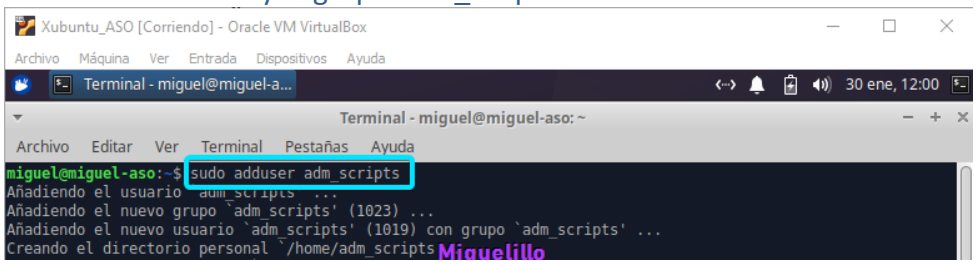
Práctica 1ª: Instala ***Visual Studio Code*** en el Ubuntu cliente y configúralo para que el usuario **adm_scripts** pueda administrar lo almacenado en /usr/local/bin.

El usuario adm_scripts no debería poder acceder a otros directorios.´

Instalamos el servidor para conectarnos ssh

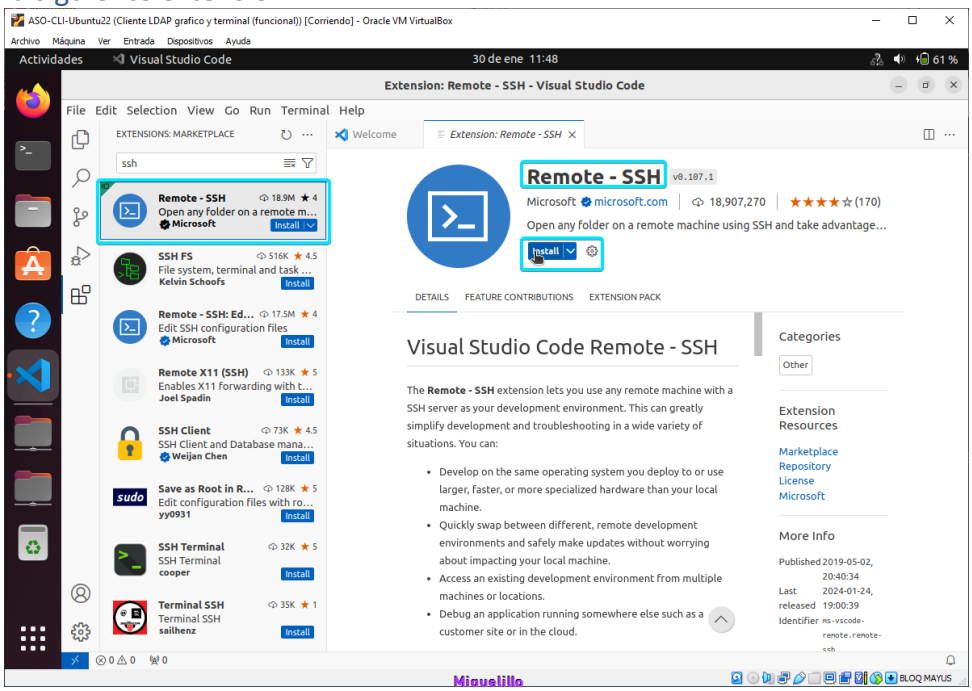


Creamos el usuario y el grupo adm_scripts

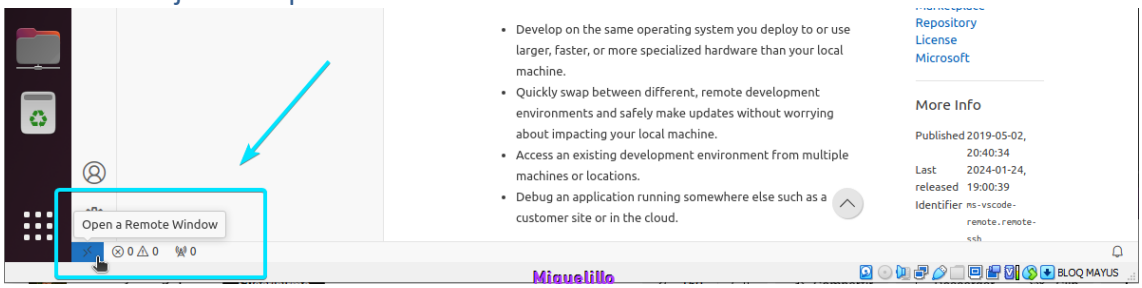


Vamos a la tienda de Ubuntu y buscamos Visual Studio Code y lo instalamos

Abrimos Visual Studio Code, vamos a extensiones y buscamos **SSH** e instalamos la siguiente extensión

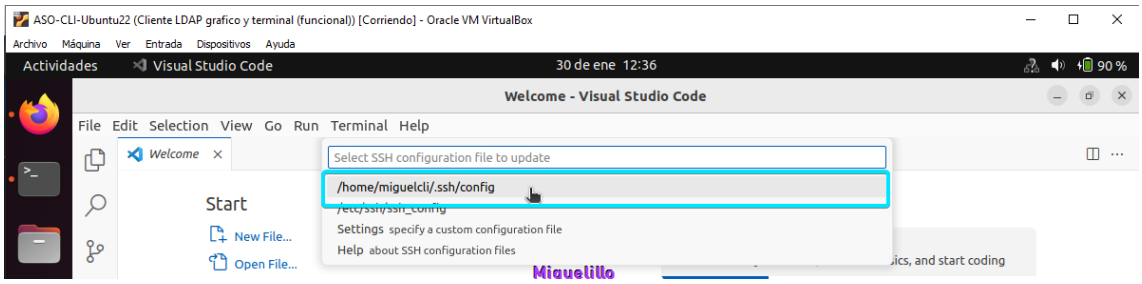


Clicamos abajo a la izquierda

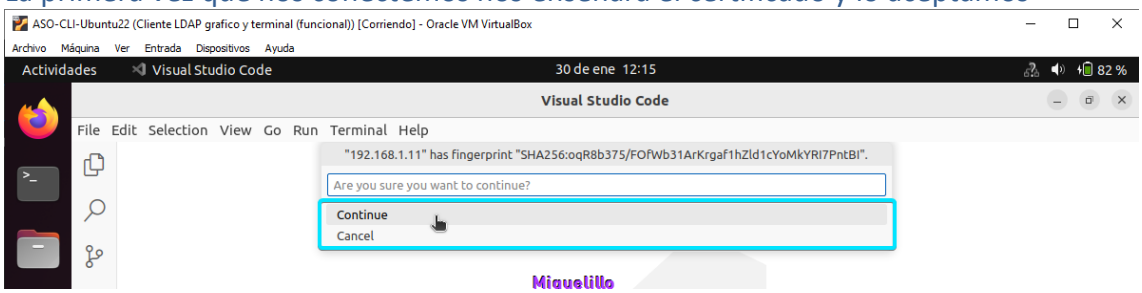


Clicasmo en añadir nuevo host

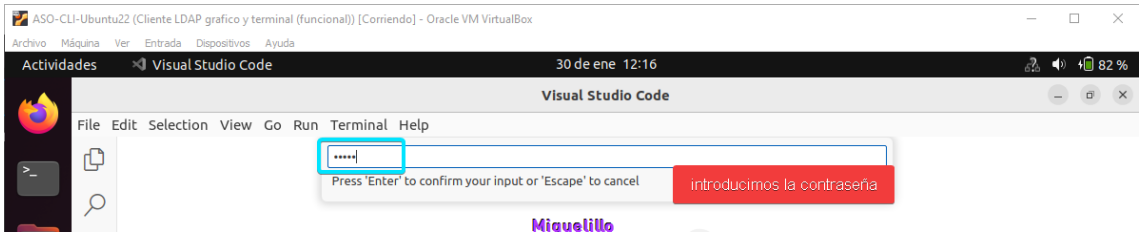
Guardamos el usuairo



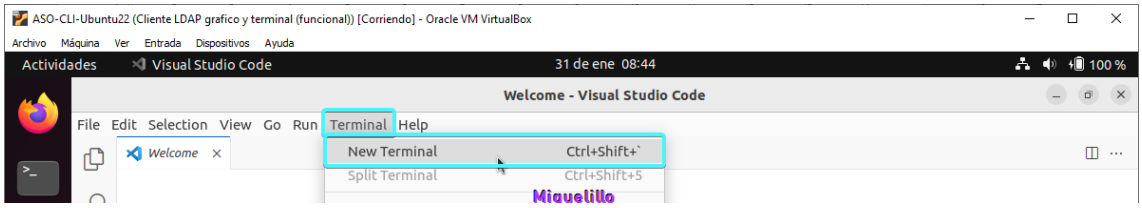
La primera vez que nos conectemos nos enseñará el certificado y lo aceptamos



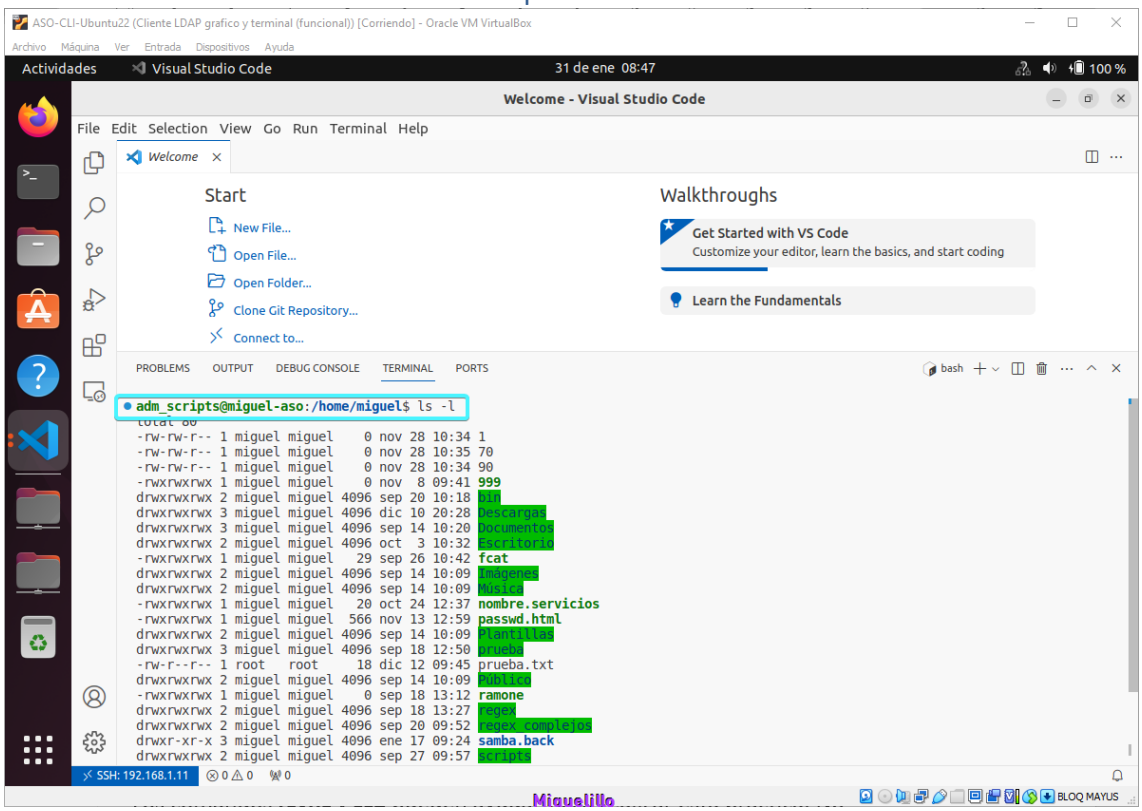
Introducimos la contraseña

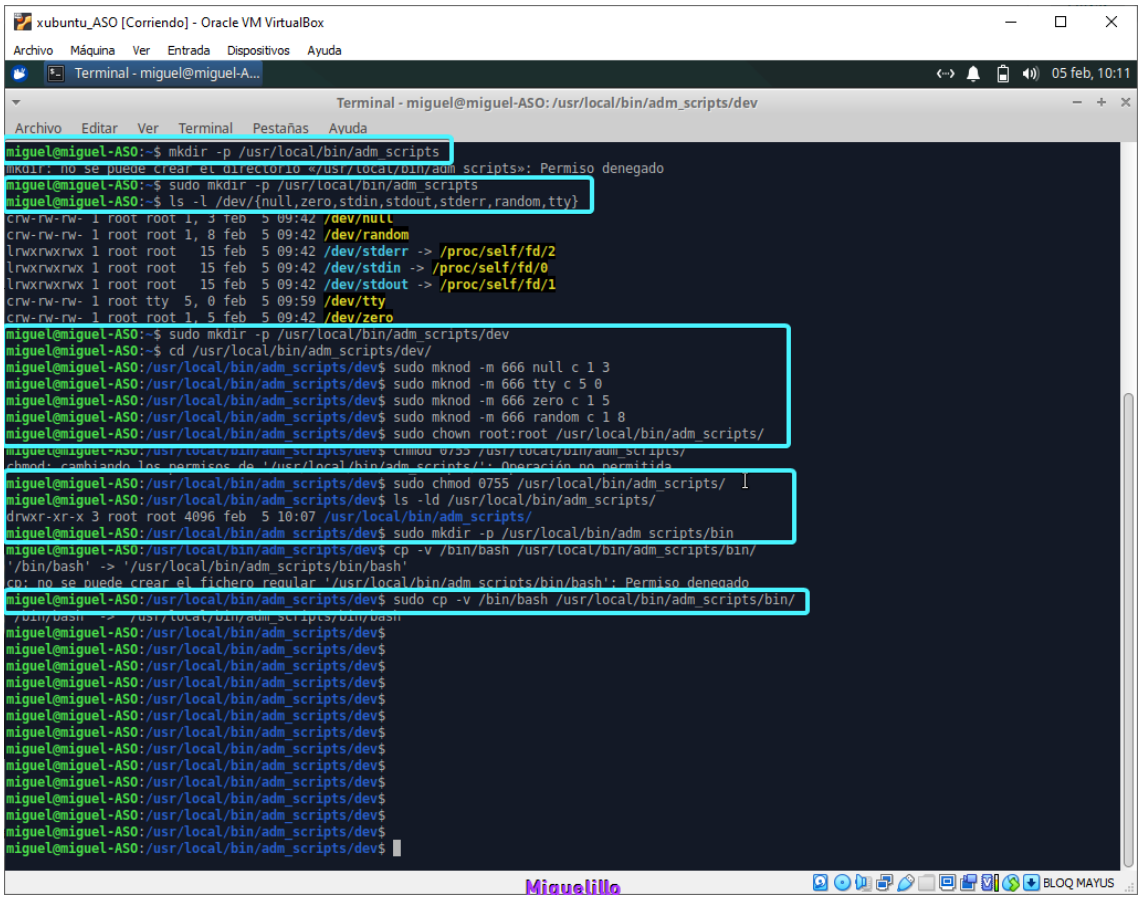


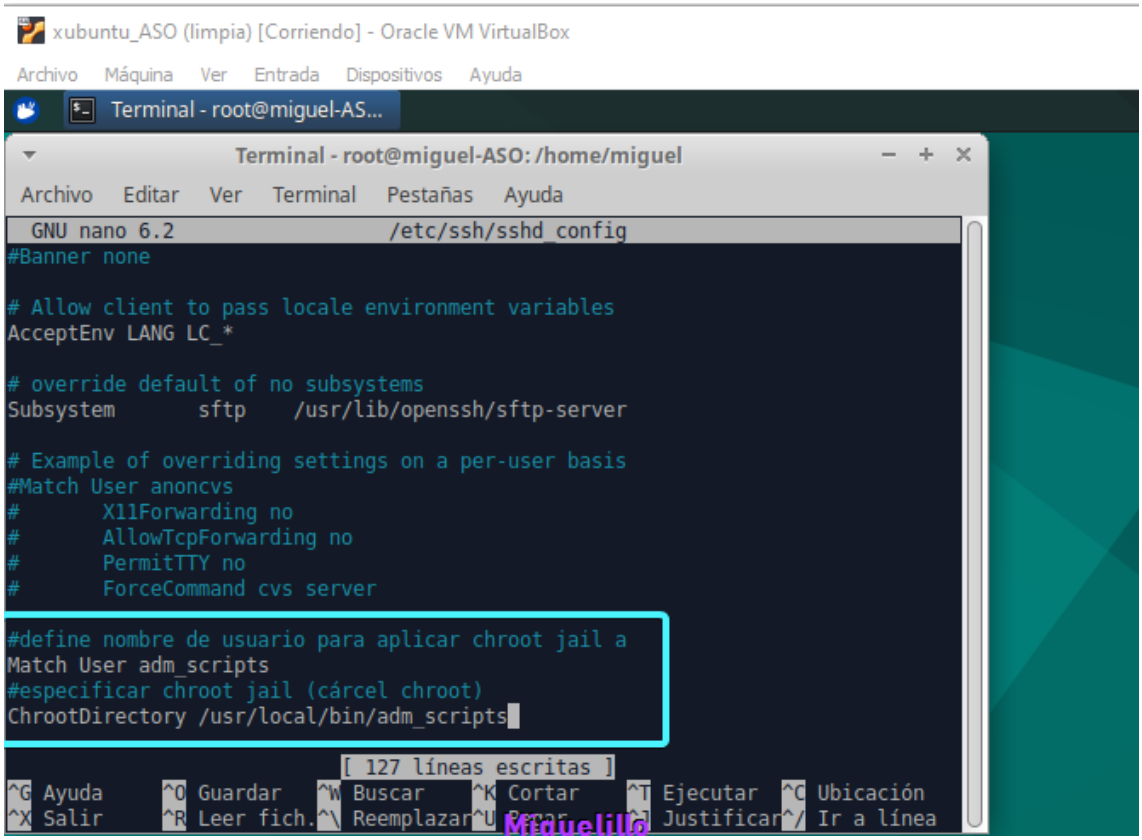
Vamos a **Terminal → New Terminal**



Nos muestra la consola de nuestro lnux podemos introducir comandos







Práctica 2ª: instala algún software que te permita administrar, de forma grafica remota, el servidor.

Práctica 3ª: Crea un script Linux que te permita hacer copias de seguridad de determinados directorios. Estas copias se deberán mover a otro equipo (utilizamos el

cliente).

Los comandos **rsync** y **scp** pueden ayudarnos a realizar esta práctica. No obstante, puedes utilizar otros.

Directorios

Nfs

Samba

Nombre copia$(date +"%d-%m-&Y")

Práctica 4ª: crea los usuarios adm_samba, adm_users y adm_nfs. Lo debes hacer conectándote al servidor mediante ssh con el usuario que se creó en el servidor durante la instalación (antonio en mi caso). De momento, vamos a configurar de forma adecuada la cuenta adm_users para que pueda utilizar comandos relacionados con los usuarios y los grupos. **Sudo visudo**

Recordemos ssh

Conexión a una estación remota con SSH

La función más común para SSH es establecer una sesión de trabajo remota en uso de técnicas criptográficas para transmitir la información. El uso del cliente SSH es bastante sencillo: **$ssh user@host**

user: es el usuario que se conectará a la máquina remota.

host: representa la IP o el nombre de dominio del servidor SSH al que nos queremos conectar.

Es todo lo que tenemos que escribir para iniciar una sesión remota como usuario (user) en el equipo anfitrión (host). Al ejecutar la orden nos pedirá la contraseña del usuario en el equipo remoto y, si la escribimos de manera correcta, podremos acceder a la sesión de trabajo remota para escribir órdenes.

$ssh antonio@192.168.56.10

antonio@192.168.56.10 's password: ***

antonio@us-01: ~ $

Para finalizar la conexión escribiremos exit .

Si no se especifica el nombre de usuario a la hora de invocar la orden SSH, intentará hacer la conexión con el usuario con el que estamos conectados al terminal de GNU / Linux.

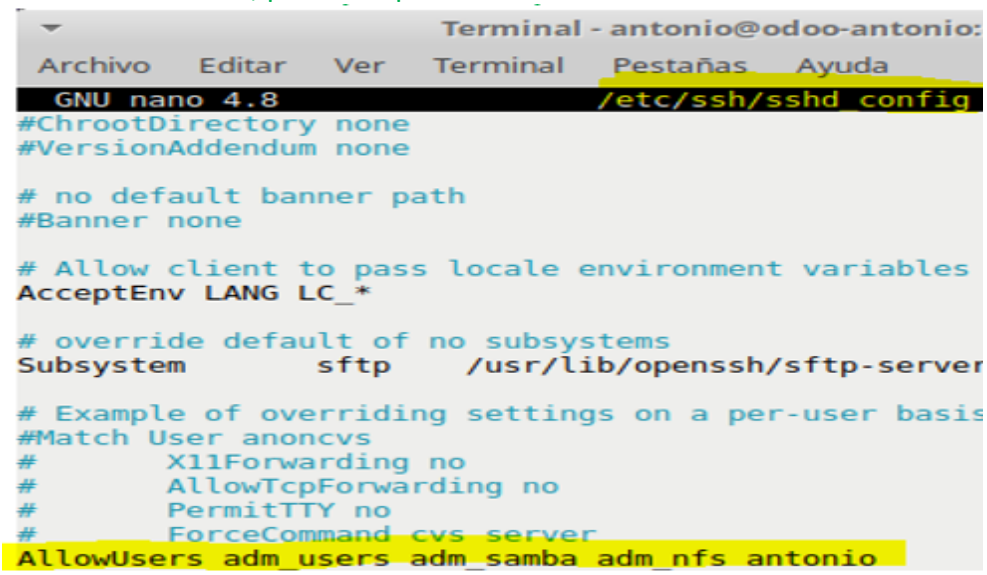
usuario1@ud-01$ssh 192.168.56.10

usuario1@192.168.56.10 's password:

En este caso intenta conectarse como usuario1, ya que no se ha especificado con qué usuario debe conectarse.

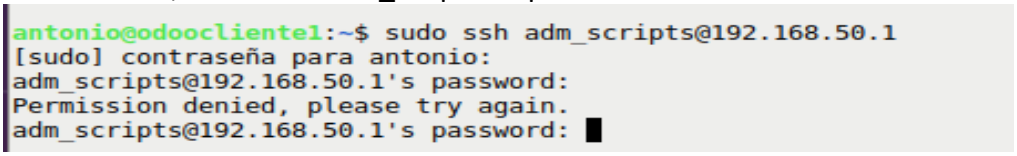
Configuración acceso mediante ssh

Si fuera necesario, podemos permitir el acceso a determinados usuarios.



$sudo systemctl restart ssh

De esta forma, el usuario adm_scripts no podrá acceder



Observación: puedes añadir aspectos de ssh que consideres oportunos**.**

Recordemos la configuración de /etc/sudoers

- Este archivo se proporciona de forma básica y permite al usuario root hacer cualquier cosa como si fuera otro usuario y permite a los miembros del grupo admin hacerse root (uno de ellos es el usuario que se crea durante la instalación).

- Modificar **/etc/sudoers**: a través de la herramienta proporcionada por **visudo**:

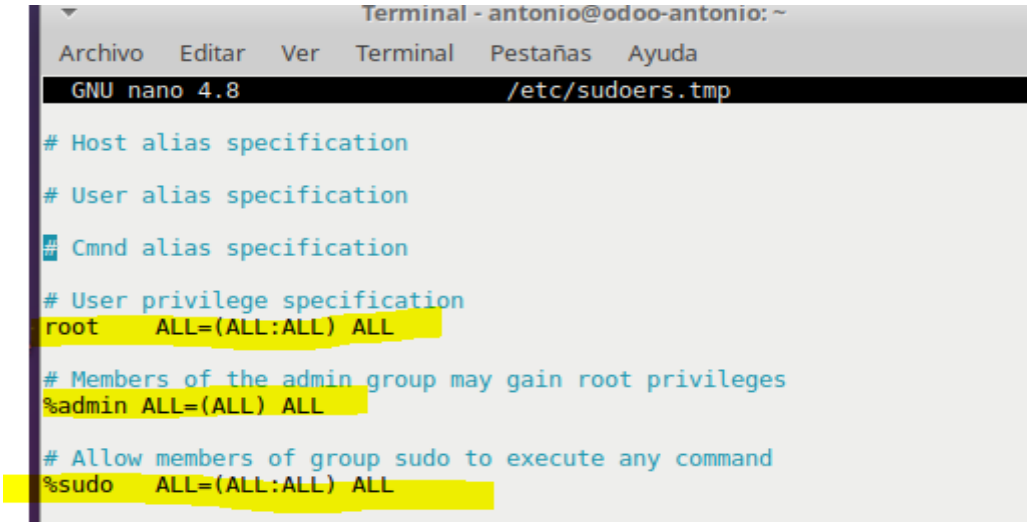
$sudo visudo → estamos editando /etc/sudoers. Además de permitirnos editar este archivo, visudo sirve como chequeo de errores ya que un sólo error en este archivo podría incluso bloquear nuestro acceso como root. **Si hay problemas, nos pregunta: "qué hacemos ahora?" (What now?). Podemos contestar con la letra e si queremos editar de nuevo el archivo, x para salir sin guardar o q para salir guardando los cambios.**

- El archivo por defecto tiene estas reglas (depende de la versión):

root ALL = (ALL) ALL

%admin ALL = (ALL) ALL

%sudo ALL = (ALL) ALL



La primera columna define a **qué usuario** o **grupo** se le aplica la regla. Podemos referirnos a un usuario poniendo su nombre (por ejemplo, root). Podemos poner varios separados por comas o un grupo añadiendo % delante del grupo (por ejemplo %admin).

La segunda columna, antes del signo =, define a **qué equipos (hosts)** se le aplica la regla. En el ejemplo vemos que aparece ALL, es decir, a todos los hosts.

Después del signo =, dentro de los paréntesis, se especifica el **cómo qué nombre de usuario se ejecutarán el o los comandos** que marque esta regla sudo En este ejemplo se ha marcado ALL, es decir, que se podrá ejecutar como cualquier usuario.

La última columna define **qué comandos podrá ejecutar el usuario**. En este ejemplo es ALL, es decir, se podrá ejecutar cualquier comando.

Nota: root ALL = (ALL) ALL se refiere a que, root puede hacer de todo como cualquier usuario en cualquier equipo. Como sabemos, root está deshabilitado por defecto.

IMPORTANTE: Si edito /etc/sudores con nano y cometo errores, puedo dejar al sistema inaccesible. Lo podemos solucionar así:

$pkexec /usr/sbin/visudo → clave → ya puedo acceder

Ejemplos teóricos para entender el funcionamiento

Ejemplo 1: Nuestra organización dispone, entre otros, de un servidor Web llamado *web1*. Tenemos un administrador cuyo nombre de usuario *es jorge*, el cual gestiona este servidor Web. **No queremos darle acceso completo como root** a la máquina por temas de seguridad. Todo lo que queremos darle es la posibilidad de utilizar **apache2ctl** de modo que pueda recargar y reiniciar Apache en este servidor. La regla resultante, en web1, será:

jorge web1 = (root) /usr/sbin/apache2ctl

En este caso, sólo estará esta regla en /etc/sudoers. En el resto de servidores, no aparecerá esta regla.

Importante: por temas de seguridad siempre hay que poner la ruta completa del o de los comandos.

Tal como está esta regla, cada vez que jorge ejecute apache2ctl le pedirá clave. Si queremos que no la pida:

jorge web1 = (root) NOPASSWD: /usr/sbin/apache2ctl

No es aconsejable esta medida por razones de seguridad.

Ejemplo 2

%directores principal = (director1) /usr/facturacion, (root) /var/log/*

Los usuarios que pertenezcan al grupo del sistema llamado 'directores' pueden en el equipo llamado 'principal' ejecutar como si fuera el usuario 'director1' la aplicación llamada 'facturacion'. Además, como usuarios 'root' pueden ver el contenido de lo que contenga el directorio /var/log.

Ejemplo 3 (continuación de ejemplo1): En lugar de sólo tener el usuario jorge, tenemos también enrique y aitor, los cuales trabajan en nuestro clúster de servidores Web: web1, web2 y web3. Además del acceso a apache2ctl, queremos proporcionar acceso a a2enmod, a2dismod, a2ensite y a2dissite.

Vamos a proponer dos soluciones, una sin alias y otra con alias.

Solución sin alias

jorge,enrique,aitor web1,web2,web3 = (root) /usr/sbin/apache2ctl, (root) /usr/sbin/a2enmod, (root) /usr/sbin/a2dismod, (root) /usr/sbin/a2ensite, (root) /usr/sbin/a2dissite

Los alias sudo

Un alias se refiere a un usuario, un comando o a un equipo. El alias engloba bajo un solo nombre (nombre del alias) una serie de elementos que después en la parte de definición de reglas serán referidos aplicados bajo ciertos criterios.

La sintaxis básica de cada uno de ellos es:

User_Alias - define alias de usuarios normales.

Ejemplo: **User_Alias NOMBREALIAS = user1, user2, user3**

Host_Alias - define alias de hosts o equipos.

Ejemplo: **Host_Alias NOMBREALIAS = host1, host2, host3**

Cmnd_Alias - define alias de comandos.

Ejemplo: **Cmnd_Alias NOMBREALIAS = /bin/comando1, /sbin/comando2**

Observación: NOMBREALIAS debe estar formado por letras mayúsculas, dígitos, del 0 al 9 y el carácter _.

Solución con alias

```
Host_Alias CLUSTERWEB = web1, web2, web3
User_Alias USERSAPACHE = jorge, enrique, aitor
Cmnd_Alias CMDSAPACHE = /usr/sbin/apache2ctl, \
                        /usr/sbin/a2enmod, \
                        /usr/sbin/a2dismod, \
                        /usr/sbin/a2ensite, \
                        /usr/sbin/a2dissite
# Regla
USERSAPACHE CLUSTERWEB = (root) CMDSAPACHE
```

Observación: la barra \ es para poder tener varias líneas. Se facilita así la lectura.