

Ruben Apablaza Muñoz
-OZonE-

Configuración y Hardening de Servidores Debian con WireGuard y GeolIP



OZonE
CIBERSECURITY

INTRODUCCIÓN

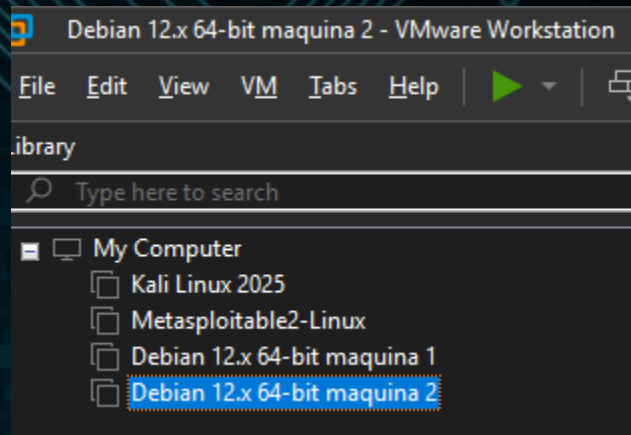
Este documento detalla el proceso de configuración y endurecimiento de dos máquinas virtuales Debian 12.x de 64 bits utilizando VMware Workstation. El objetivo principal es establecer una conexión VPN segura entre ambas máquinas usando WireGuard y reforzar su seguridad mediante la implementación de reglas de firewall con Iptables, incluyendo filtrado geográfico con GeoIP. Además, se aborda el endurecimiento del servicio SSH para mejorar la autenticación de usuarios y proteger el acceso remoto. Todo el proceso se realiza paso a paso, desde la clonación de las máquinas y la configuración de red, hasta la generación de claves, la instalación de servicios y la aplicación de políticas de seguridad.

TABLA DE CONTENIDO

- Configuración inicial del laboratorio.
- Instalación de WireGuard VPN.
 - Generación de llaves privada y pública.
- Configuración de archivos para la VPN.
- Configuración de IPTables.
- Apertura del puerto de WireGuard (51820 UDP).
 - Aclaración sobre la seguridad de puertos y servicios.
- Levantar interfaz WireGuard.
 - Comprobación de funcionamiento con ping y mtr.
- Instalación de GeoIP.
 - Descarga y preparación de la base de datos GeoIP
 - Copiar reglas básicas a archivo de configuración de iptable.
 - Bloquear tráfico con GeoIP
- Practicas adicionales de hardening
 - Autenticación de usuario con llave publica y contraseña.
 - Cambiar número de puerto ssh.
 - Creación de nuevos usuarios y permisos asociados.
 - Prohibir el acceso vía ssh como usuario root.

1. CONFIGURACIÓN INICIAL DEL LABORATORIO

Clonar maquinas en el virtualizador, en este caso [vmware](#).



importante que estén en modo **BRIDGE**

Cabe destacar que toda la configuración se hace en los dos equipos.

A continuación, encendemos nuestra "maquina1" y vemos la dirección IPv4 con `ip a`, en este caso corresponden a

maquina1 = 192.168.0.35/24

maquina2 = 192.168.0.36/24

```
Debian GNU/Linux 12 maquina1 tty1
maquina1 login: ruben1
Password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul 25 10:03:49 -04 2025 on tty1
ruben1@maquina1:~$ ip 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 noprefixroute
        valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:00:00:00 brd ff:ff:ff:ff:ff:ff
    altname enp2s1
    inet 192.168.0.35/24 brd 192.168.0.255 scope global dynamic ens33
        valid_lft 604770sec preferred_lft 604770sec
    inet6 fe80::208:1ff:fe00:0000/64 scope global dynamic mngtmpaddr
        valid_lft 3598sec preferred_lft 3598sec
    inet6 fe80::208:1ff:fe00:0000/64 scope link
        valid_lft forever preferred_lft forever
ruben1@maquina1:~$
```

Con esta dirección podemos acceder por terminal ssh, en este caso utilizaré vscode

Con el comando `ssh ruben1@192.168.0.35` nos solicitará las credenciales de acceso

```
$ ssh ruben1@192.168.0.35
ruben1@192.168.0.35's password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Fri Jul 25 21:37:45 2025
ruben1@maquina1:~$
```

Para llevar a cabo todas las configuraciones vamos a cambiar a usuario root con

`su - root`

```
ruben1@maquina1:~$ su - root
Contraseña:
root@maquina1:~#
```

Para cambiar nombre a la maquina ejecutar

`hostnamectl set-hostname santiago`

En este caso lo volveré a dejar como `maquina1`, básicamente se puede realizar este cambio las veces que sean necesarias.

```
root@maquina1:~# hostnamectl set-hostname santiago
root@maquina1:~# bash
root@santiago:~# hostnamectl set-hostname maquina1
root@santiago:~# bash
root@maquina1:~#
```


2. INSTALACIÓN DE WIREGUARD VPN

actualizar el sistema con `apt update && apt upgrade`

```
root@maquina1:~# apt update && apt upgrade
Obj:1 http://deb.debian.org/debian bookworm InRelease
Des:2 http://deb.debian.org/debian bookworm-updates InRelease [55,4 kB]
Des:3 http://security.debian.org/debian-security bookworm-security InRelease [48,0 kB]
Descargados 103 kB en 0s (292 kB/s)
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Todos los paquetes están actualizados.
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Calculando la actualización... Hecho
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
root@maquina1:~#
```

Una vez actualizado el sistema instalamos en los dos servidores la vpn wireguard con

`apt install wireguard -y`

```
root@maquina1:~# apt install wireguard -y
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes adicionales:
  wireguard-tools
Paquetes sugeridos:
  openresolv | resolvconf
Se instalarán los siguientes paquetes NUEVOS:
  wireguard wireguard-tools
0 actualizados, 2 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
Se necesita descargar 95,8 kB de archivos.
Se utilizarán 346 kB de espacio de disco adicional después de esta operación.
Des:1 http://deb.debian.org/debian bookworm/main amd64 wireguard-tools amd64 1.0.20210914-1+b1 [87,6 kB]
Des:2 http://deb.debian.org/debian bookworm/main amd64 wireguard all 1.0.20210914-1 [8.216 B]
Descargados 95,8 kB en 0s (673 kB/s)
Seleccionando el paquete wireguard-tools previamente no seleccionado.
(Leyendo la base de datos ... 34587 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../wireguard-tools_1.0.20210914-1+b1_amd64.deb ...
Desempaquetando wireguard-tools (1.0.20210914-1+b1) ...
Seleccionando el paquete wireguard previamente no seleccionado.
Preparando para desempaquetar .../wireguard_1.0.20210914-1_all.deb ...
Desempaquetando wireguard (1.0.20210914-1) ...
Configurando wireguard-tools (1.0.20210914-1+b1) ...
wg-quick.target is a disabled or a static unit, not starting it.
Configurando wireguard (1.0.20210914-1) ...
Procesando disparadores para man-db (2.11.2-2) ...
root@maquina1:~#
```

Ahora generamos la llave privada y pública para vpn EN LOS DOS EQUIPOS

`cd /etc/wireguard/` con esto se cambia al directorio wireguard

`wg genkey | tee privatekey | wg pubkey > publickey` con esto se generan las llaves

`ls` muestra las llaves generadas o dicho de otro modo verifica que las llaves fueron creadas

con `cat privatekey` puedo ver la llave privada

con `cat publickey` puedo ver la llave publica

OJO MUY IMPORTANTE,,, LA LLAVE PRIVADA NO SE DEBE COMPARTIR NUNCA

En este caso

```
root@maquina2:/etc/wireguard# wg genkey | tee privatekey | wg pubkey > publickey
root@maquina2:/etc/wireguard# ls
privatekey  publickey
root@maquina2:/etc/wireguard# cat privatekey
0PnzWT6>
root@maquina2:/etc/wireguard# cat publickey
JEzsScJTSdVK2BgV5dzR3pDjHio1X09plz7euM6nBj0=
root@maquina2:/etc/wireguard#
```


3. CONFIGURACIÓN DE LOS ARCHIVOS PARA LA VPN

Esto se realiza desde cualquier editor de texto como vi, vim o nano; para esta ocasión se trabaja con nano

`nano /etc/wireguard/wg0.conf`

cómo se trata de un laboratorio, no tenemos ip publica pero con la ip del modo bridge nos sirve en este caso.

Básicamente, en la configuración de la maquina1,

en **Interface** , **PrivateKey** colocamos la llave privada del **servidor 1** (para eso era el `cat privatekey`)

en **Peer**, colocamos los datos del **servidor 2**, donde dice **Endpoint** y en **PublicKey**

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  PORTS  TERMINAL
GNU nano 7.2 /etc/wireguard/wg0.conf *
[Interface]
PrivateKey = LLAVE_PRIVADA_DE_LA_MAUQUINA1
Address = 10.0.0.1/24
ListenPort = 51820

[Peer]
PublicKey = LLAVE_PUBLICA_DE_MAUQUINA2_O_CLIENTE
Endpoint = IP_PUBLICA_O_LOCAL_DE:MAQUINA2_O_CLIENTE:51820
AllowedIPs = 10.0.0.2/32
PersistentKeepalive = 25
```

En mi caso la configuración queda así:

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  PORTS  TERMINAL
GNU nano 7.2 /etc/wireguard/wg0.conf *
[Interface]
PrivateKey = +G7KG7 [REDACTED]
Address = 10.0.0.1/24
ListenPort = 51820

[Peer]
PublicKey = JEzsScJTSdVK2BgV5dzR3pDjHio1X09plz7euM6nBj0=
Endpoint = 192.168.0.36:51820
AllowedIPs = 10.0.0.2/32
PersistentKeepalive = 25
```


4. CONFIGURACIÓN DE IPTABLES

iptables: (habilitar las reglas en el firewall... iptables es el firewall de Debian, ver alternativa **nftables** que es su reemplazo más moderno)

se ejecuta

```
iptables -vnL
```

pero no funciona porque no está instalado. Por lo tanto instalamos iptables porque por defecto no viene en Debian

```
apt install iptables-persistent
```

y ahora si lo ejecutamos con

```
iptables -vnL
```

por defecto se puede ver que está permitiendo todo el trafico

```
root@maquina1:/etc/wireguard# iptables -vnL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
 pkts bytes target    prot opt in     out     source            destination
root@maquina1:/etc/wireguard#
```

con

```
cat /etc/iptables/rules.v4
```

da cuenta que la carpeta de configuración se encuentra vacía y por eso estaría permitiendo todo el trafico

lo mismo sucede con

```
cat /etc/iptables/rules.v6
```

5. APERTURA DEL PUERTO DE WIREGUARD (51820 UDP)

Necesitamos abrir el puerto en el que está escuchando el servicio de wireguard

`cat wg0.conf`

podemos ver la configuración de la interfaz de wireguard y vemos el número de puerto en el que está escuchando el servicio, en este caso el **51820**

configuración **maquina1**

```
root@maquina1:/etc/wireguard# cat wg0.conf
[Interface]
PrivateKey = +G7KGZJL1
Address = 10.0.0.1/24
ListenPort = 51820

[Peer]
PublicKey = JEzsScJTSdVK2BgV5dzR3pDjHio1X09plz7euM6nBj0=
Endpoint = 192.168.0.36:51820
AllowedIPs = 10.0.0.2/32
PersistentKeepalive = 25
root@maquina1:/etc/wireguard#
```

Configuración **maquina2**

```
root@maquina2:/etc/wireguard# cat wg0.conf
[Interface]
PrivateKey = 0PnzK
Address = 10.0.0.2/24
ListenPort = 51820

[Peer]
PublicKey = 9eCxN9HwqWpAk6zKRpuZFqBOHIS8JZsg9abj8+RazM=
Endpoint = 192.168.0.35:51820
AllowedIPs = 10.0.0.1/32
PersistentKeepalive = 25
root@maquina2:/etc/wireguard#
```

entonces abrimos ese puerto con:

`iptables -A INPUT -p udp --dport 51820 -j ACCEPT`

con

`iptables -vnL`

podemos ver como ya está habilitado el puerto.


```
root@maquina1:/etc/wireguard# iptables -A INPUT -p udp --dport 51820 -j ACCEPT
root@maquina1:/etc/wireguard# iptables -vnl
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
    0    0 ACCEPT     17    --    *      *       0.0.0.0/0            0.0.0.0/0            udp dpt:51820

Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination

Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source               destination
root@maquina1:/etc/wireguard#
```

OJO... SIEMPRE POR SEGURIDAD SIEMPRE SE DEBE CAMBIAR LOS NUMEROS DE PUERTO POR DEFECTO.

ACLARACION: "Lo vulnerable no es el puerto, sino el servicio"

El puerto en sí no tiene vulnerabilidades.

Lo que puede ser vulnerable es el programa (servicio) que está escuchando en ese puerto.

Ejemplo claro:

El puerto 80 es el estándar para servidores web (HTTP).

Ese puerto en sí no tiene fallas.

Pero si el servicio que escucha en el puerto 80 es, por ejemplo, un Apache viejo o mal configurado, ese servicio sí puede ser vulnerable.

Entonces, aunque el puerto 80 esté abierto, el riesgo real depende del servicio detrás.

¿Qué significa esto en ciberseguridad?

No se trata solo de escanear puertos.

Lo importante es identificar qué servicio está corriendo y su versión.

6. LEVANTAR INTERFAZ WIREGUARD Y HABILITAR AL INICIO

Levantamos el servicio con `wg-quick up wg0`

```
root@maquina1:/etc/wireguard# wg-quick up wg0
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.0.0.1/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
root@maquina1:/etc/wireguard#
```

y con `systemctl enable wg-quick@wg0` hacemos que la interfaz se inicie cada vez que reiniciamos el servidor

```
root@maquina1:/etc/wireguard# systemctl enable wg-quick@wg0
Created symlink /etc/systemd/system/multi-user.target.wants/wg-quick@wg0.service → /lib/systemd/system/wg-quick@.service.
root@maquina1:/etc/wireguard#
```

con el comando `ip route` verificamos que nuestra vpn ya está habilitada ya que la red 10.0.0.0/24 se enruta a través de la interfaz `wg0`, **que es la VPN de WireGuard.**

```
root@maquina1:/etc/wireguard# ip route
default via 192.168.0.1 dev ens33
10.0.0.0/24 dev wg0 proto kernel scope link src 10.0.0.1
192.168.0.0/24 dev ens33 proto kernel scope link src 192.168.0.35
root@maquina1:/etc/wireguard#
```

Eso significa que cualquier tráfico hacia esa red va por la VPN, lo cual es señal de que está funcionando.

Para comprobar que la vpn está funcionando se usa `ping`

Se hace ping a la ip del otro servidor a la que aparece en el campo `AllowedIPs = 10.0.0.2/32`

`ping -c 4 10.0.0.2`

```
root@maquina1:/etc/wireguard# ping -c 4 10.0.0.2
PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.
64 bytes from 10.0.0.2: icmp_seq=1 ttl=64 time=0.345 ms
64 bytes from 10.0.0.2: icmp_seq=2 ttl=64 time=1.53 ms
64 bytes from 10.0.0.2: icmp_seq=3 ttl=64 time=1.69 ms
64 bytes from 10.0.0.2: icmp_seq=4 ttl=64 time=1.47 ms

--- 10.0.0.2 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3033ms
rtt min/avg/max/mdev = 0.345/1.258/1.693/0.533 ms
root@maquina1:/etc/wireguard#
```


`mtr` (comando que permite ver el camino completo que sigue un paquete, es como `tracert` pero más completo y en tiempo real)

se instala con

`apt install mtr`

y se llama con

`mtr 10.0.0.2`

```
My traceroute [v0.95]
maquina1 (10.0.0.1) -> 10.0.0.2 (10.0.0.2) 2025-07-26T13:32:02-0400
Keys: Help Display mode Restart statistics Order of fields quit
      Packets
Host Loss% Snt Last Avg Best Wrst StDev
1. 10.0.0.2 0.0% 13 1.8 1.0 0.3 2.0 0.6
```

CON ESTO SE MUESTRA LA VPN YA CONFIGURADA!!!

`wg show`

muestra el estado actual de las interfaces VPN de wireguard

```
root@maquina1:/etc/wireguard# wg show
interface: wg0
  public key: 9eCxN9HwqWvpAk6zKRpuZFqBOHIS8JZsg9abj8+RazM=
  private key: (hidden)
  listening port: 51820

peer: JZzsScJTSdVK2BgV5dzR3pDjHio1X09plz7euM6nBj0=
  endpoint: 192.168.0.36:51820
  allowed ips: 10.0.0.2/32
  latest handshake: 2 minutes, 17 seconds ago
  transfer: 22.01 KiB received, 44.25 KiB sent
  persistent keepalive: every 25 seconds
root@maquina1:/etc/wireguard#
```

7. INSTALACIÓN DE GeoIP

lo primero es

```
cat <<EOF> /etc/apt/sources.list
```

```
deb http://deb.debian.org/debian/ bookworm main contrib non-free non-free-firmware
```

```
deb-src http://deb.debian.org/debian/ bookworm main contrib non-free non-free-firmware
```

```
deb http://security.debian.org/debian-security bookworm-security main contrib non-free non-free-firmware
```

```
deb-src http://security.debian.org/debian-security bookworm-security main contrib non-free non-free-firmware
```

```
deb http://deb.debian.org/debian/ bookworm-updates main contrib non-free non-free-firmware
```

```
deb-src http://deb.debian.org/debian/ bookworm-updates main contrib non-free non-free-firmware
```

EOF

Después ejecutamos

```
apt update
```

y después instalamos los paquetes

```
apt install -y xtables-addons-common xtables-addons-source build-essential module-assistant iptables iptables-persistent linux-headers-$(uname -r) libtext-csv-xs-perl wget curl unzip gzip
```

y a continuación

```
apt install -y dh-dkms debhelper dkms
```

después

```
m-a prepare
```

y ahora

```
m-a build xtables-addons
```


y por último

m-a install xtables-addons

```
root@maquina1:/etc/wireguard# m-a build xtables-addons
Extracting the package tarball, /usr/src/xtables-addons.tar.bz2, please wait...
Hecho con /usr/src/xtables-addons-modules-6.1.0-37-amd64_3.23-1_amd64.deb .
root@maquina1:/etc/wireguard# m-a install xtables-addons
Seleccionando el paquete xtables-addons-modules-6.1.0-37-amd64:amd64 previamente no seleccionado.
(Leyendo la base de datos ... 66919 ficheros o directorios instalados actualmente.)
Preparando para desempaquetar .../xtables-addons-modules-6.1.0-37-amd64_3.23-1_amd64.deb ...
Desempaquetando xtables-addons-modules-6.1.0-37-amd64:amd64 (3.23-1) ...
Configurando xtables-addons-modules-6.1.0-37-amd64:amd64 (3.23-1) ...
fijado xtables-addons-modules-6.1.0-37-amd64 como instalado automáticamente.
root@maquina1:/etc/wireguard#
```

8. DESCARGA Y PREPARACIÓN DE LA BASE DE DATOS GEOIP

antes de continuar se recomienda cambiar al directorio root

`cd /root`

Desde <https://db-ip.com/db/download/ip-to-country-lite> copiamos la dirección del vínculo

The screenshot shows the website db-ip.com/db/download/ip-to-country-lite. It displays two download options: CSV and MMDb. Both options show a release date of July 2025, supported languages (English for CSV, and English, French, German, Portuguese, Chinese, Japanese, Russian, Persian, Korean for MMDb), and a number of records of 593,488. The CSV file size is 23.5 MB, while the MMDb file size is 7.1 MB. The MD5SUM for CSV is 6e180273f86b8beab38907397c6692eb and for MMDb is ff2b15f26abd93fdb98ee7faa3c. The SHA1SUM for CSV is 9063a39521c602df1b126a6192a5fb83054a0679 and for MMDb is a714b380335c48c8d4dad30b. A context menu is open over the CSV download button, showing options: 'Abrir enlace en una pestaña nueva', 'Abrir enlace en una ventana nueva', 'Abrir el enlace en una ventana de incógnito', 'Guardar enlace como...', 'Copiar dirección de enlace', and 'Inspeccionar'.

Formato	Release	Supported language(s)	Number of records	File size	MD5SUM	SHA1SUM
CSV	July 2025	English	593,488	23.5 MB	6e180273f86b8beab38907397c6692eb	9063a39521c602df1b126a6192a5fb83054a0679
MMDb	July 2025	English, French, German, Portuguese, Chinese, Japanese, Russian, Persian, Korean	593,488	7.1 MB	ff2b15f26abd93fdb98ee7faa3c	a714b380335c48c8d4dad30b

y con

`wget https://download.db-ip.com/free/dbip-country-lite-2025-07.csv.gz`

instalamos en nuestros 2 servidores

```
root@maquina1:~# wget https://download.db-ip.com/free/dbip-country-lite-2025-07.csv.gz
--2025-07-26 13:54:57-- https://download.db-ip.com/free/dbip-country-lite-2025-07.csv.gz
Resolviendo download.db-ip.com (download.db-ip.com)... 172.67.75.166, 104.26.5.15, 104.26.4.15
Conectando con download.db-ip.com (download.db-ip.com)[172.67.75.166]:443... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 3780668 (3,6M) [application/octet-stream]
Grabando a: «dbip-country-lite-2025-07.csv.gz»

dbip-country-lite-2025-07. 100%[=====>] 3,61M 18,8MB/s en 0,2s

2025-07-26 13:54:57 (18,8 MB/s) - «dbip-country-lite-2025-07.csv.gz» guardado [3780668/3780668]

root@maquina1:~#
```

como el archivo llega comprimido debemos descomprimirlo con

`gunzip dbip-country-lite-2025-07.csv.gz`

y con

`mv dbip-country-lite-2025-07.csv dbip-country-lite.csv`

cambiamos el nombre al archivo y lo dejamos como `dbip-country-lite.csv`

EL NOMBRE SE CAMBIA PARA EVITAR PROBLEMAS CON COMANDOS AUTOMATIZADOS ENTRE OTRAS RAZONES

Por ejemplo:

si tienes un script o módulo que carga la base así:

`xt_geoiip_build -D /usr/share/xt_geoiip dbip-country-lite.csv`

y el archivo se llama distinto cada vez (con fecha), el comando fallaría o tendría que ser modificado cada mes.

9. CAMBIAR EL FORMATO DEL ARCHIVO QUE VIENE ILEGIBLE

Al realizar el siguiente comando sobre el archivo nos damos cuenta que el formato es ilegible

`tail -2 dbip-country-lite.csv`

```
root@maquina2:~# tail -2 dbip-country-lite.csv
fe80::,febf:ffff:ffff:ffff:ffff:ffff:ffff:ffff,ZZ
fec0::,ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff,CH
root@maquina2:~#
```

Entonces con

`awk -F, '{ gsub(/"/, "", $0); print $1 "," $2 "," $3 }' dbip-country-lite.csv
> dbip_xt_geoip.csv`

```
root@maquina1:~# awk -F, '{ gsub(/"/, "", $0); print $1 "," $2 "," $3 }' dbip-country-lite.csv > dbip_xt_g
eoip.csv
root@maquina1:~#
```

¿Qué hace este comando?

`-F,`: le dice a `awk` que el separador de campos es la coma (,).

`gsub(/"/, "", $0);`: elimina todas las comillas dobles (") de la línea completa (\$0).

`print $1 "," $2 "," $3;`: imprime los tres primeros campos separados por coma.

`> dbip_xt_geoip.csv`: guarda el resultado en un nuevo archivo.

¿Por qué es útil?

Porque la base `dbip-country-lite.csv` suele venir así:

`"1.0.0.0","1.0.0.255","AU" "1.0.1.0","1.0.3.255","CN" ...`

Y necesitas que quede en formato sin comillas para `xt_geoip_build`:

`1.0.0.0,1.0.0.255,AU 1.0.1.0,1.0.3.255,CN ...`

10. CREAR DIRECTORIO DONDE SE GUARDARÁN LOS ARCHIVOS PROCESADOS POR XT_GEOIP_BUILD (GeoIP en formato binario que iptables puede usar).

1. crear directorio

```
mkdir /usr/share/xt_geoip
```

Esto crea un directorio llamado xt_geoip en la ruta /usr/share/.

¿Para qué?

Ese directorio se usará para guardar los archivos de datos geográficos que el módulo xt_geoip necesita para funcionar. Este módulo se utiliza en conjunto con iptables para aplicar reglas basadas en geolocalización IP

2. buscar el ejecutable

y ahora buscamos en todo el sistema la ruta donde está el archivo xt_geoip con

```
find / -name xt_geoip_build
```

¿Para qué?

Necesitamos encontrar la ubicación exacta del programa xt_geoip_build, que se usa para convertir una base de datos de IPs geográficas (como la que descargamos: dbip_xt_geoip.csv) en el formato binario que xt_geoip entiende.

Nota: Este comando puede demorar varios segundos o minutos, ya que busca desde la raíz / de todo el sistema.

3. Ejecutar el convertidor

y ahora ejecutamos con

```
/usr/libexec/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip  
dbip_xt_geoip.csv
```

¿Qué hace esto?

Transforma la base de datos de geolocalización en binarios que el módulo xt_geoip puede consultar cuando se utiliza una regla de iptables del tipo:

```
iptables -A INPUT -m geoip --src-cc CN,RU -j DROP
```

Lo que significa: bloquea conexiones entrantes que vienen de China y Rusia.

```
root@maquina1:~# mkdir /usr/share/xt_geoip
root@maquina1:~# find / -name xt_geoip_build
/usr/libexec/xtables-addons/xt_geoip_build
root@maquina1:~# /usr/libexec/xtables-addons/xt_geoip_build -D /usr/share/xt_geoip dbip_xt_geoip.csv
593488 entries total
  85 IPv4 ranges for AD
  82 IPv6 ranges for AD
1049 IPv4 ranges for AE
 825 IPv6 ranges for AE
 190 IPv4 ranges for AF
 175 IPv6 ranges for AF
```


11. COPIAR REGLAS BÁSICAS A ARCHIVO DE CONFIGURACIÓN DE IPTABLES

Desde <https://wiki.debian.org/iptables>

copiamos las siguientes reglas **BÁSICAS** y las pegaremos en nuestro archivo de configuración que hasta ahora está vacío.

Con cualquier editor, yo uso nano, pegamos estas líneas en:

`nano /etc/iptables/rules.v4`

con el archivo abierto pegamos lo siguiente:

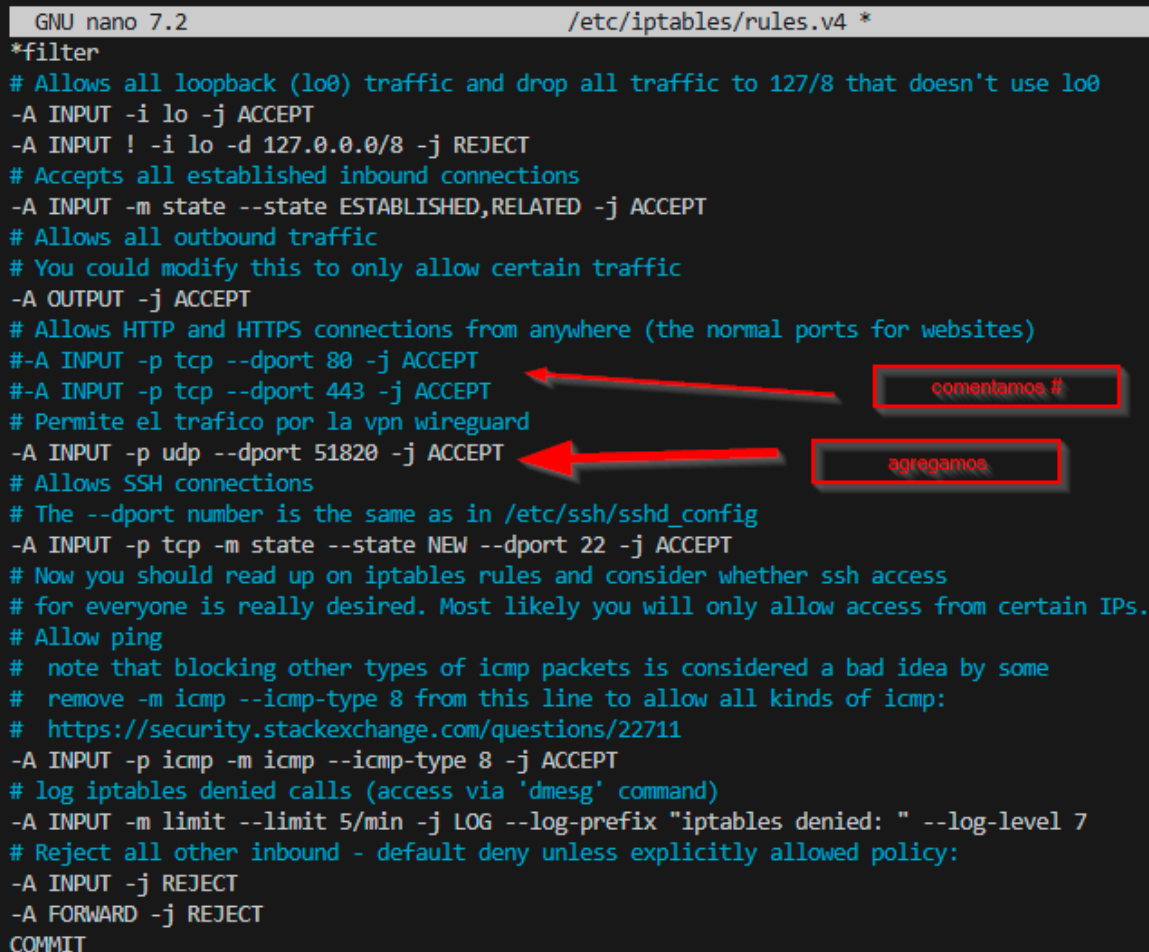
```
GNU nano 7.2 /etc/iptables/rules.v4 *
*filter
# Allows all loopback (lo0) traffic and drop all traffic to 127/8 that doesn't use lo0
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT
# Accepts all established inbound connections
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Allows all outbound traffic
# You could modify this to only allow certain traffic
-A OUTPUT -j ACCEPT
# Allows HTTP and HTTPS connections from anywhere (the normal ports for websites)
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
# Allows SSH connections
# The --dport number is the same as in /etc/ssh/sshd_config
-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
# Now you should read up on iptables rules and consider whether ssh access
# for everyone is really desired. Most likely you will only allow access from certain IPs.
# Allow ping
# note that blocking other types of icmp packets is considered a bad idea by some
# remove -m icmp --icmp-type 8 from this line to allow all kinds of icmp:
# https://security.stackexchange.com/questions/22711
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
# log iptables denied calls (access via 'dmesg' command)
-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --log-level 7
# Reject all other inbound - default deny unless explicitly allowed policy:
-A INPUT -j REJECT
-A FORWARD -j REJECT
COMMIT
█
```

```
*filter
# Allows all loopback (lo0) traffic and drop all traffic to 127/8 that doesn't
use lo0
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT
# Accepts all established inbound connections
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Allows all outbound traffic
# You could modify this to only allow certain traffic
-A OUTPUT -j ACCEPT
# Allows HTTP and HTTPS connections from anywhere (the normal ports
for websites)
-A INPUT -p tcp --dport 80 -j ACCEPT
-A INPUT -p tcp --dport 443 -j ACCEPT
# Allows SSH connections
# The --dport number is the same as in /etc/ssh/sshd_config
-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
# Now you should read up on iptables rules and consider whether ssh
access
# for everyone is really desired. Most likely you will only allow access from
certain IPs.
# Allow ping
# note that blocking other types of icmp packets is considered a bad idea
by some
# remove -m icmp --icmp-type 8 from this line to allow all kinds of icmp:
# https://security.stackexchange.com/questions/22711
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
# log iptables denied calls (access via 'dmesg' command)
-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --
log-level 7
# Reject all other inbound - default deny unless explicitly allowed policy:
-A INPUT -j REJECT
-A FORWARD -j REJECT
COMMIT
```

para que los cambios tomen efecto hay que reiniciar con
systemctl restart iptables

sí nos fijamos en el archivo abierto con nano, podemos ver que tenemos conexiones y puertos abiertos (específicamente el 80 y 443 por lo que comentamos esas líneas con #) que vamos a cerrar y vamos a permitir solo el tráfico por la vpn agregando al archivo la siguiente línea:

```
-A INPUT -p udp --dport 51820 -j ACCEPT
```



```
GNU nano 7.2 /etc/iptables/rules.v4 *
*filter
# Allows all loopback (lo0) traffic and drop all traffic to 127/8 that doesn't use lo0
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT
# Accepts all established inbound connections
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
# Allows all outbound traffic
# You could modify this to only allow certain traffic
-A OUTPUT -j ACCEPT
# Allows HTTP and HTTPS connections from anywhere (the normal ports for websites)
#-A INPUT -p tcp --dport 80 -j ACCEPT
#-A INPUT -p tcp --dport 443 -j ACCEPT
# Permite el trafico por la vpn wireguard
-A INPUT -p udp --dport 51820 -j ACCEPT
# Allows SSH connections
# The --dport number is the same as in /etc/ssh/sshd_config
-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT
# Now you should read up on iptables rules and consider whether ssh access
# for everyone is really desired. Most likely you will only allow access from certain IPs.
# Allow ping
# note that blocking other types of icmp packets is considered a bad idea by some
# remove -m icmp --icmp-type 8 from this line to allow all kinds of icmp:
# https://security.stackexchange.com/questions/22711
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
# log iptables denied calls (access via 'dmesg' command)
-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --log-level 7
# Reject all other inbound - default deny unless explicitly allowed policy:
-A INPUT -j REJECT
-A FORWARD -j REJECT
COMMIT
```

y volvemos a reiniciar iptables con:

```
systemctl restart iptables
```

12. BLOQUEAR TRÁFICO CON GEOIP

Primero abriremos el archivo con

`nano /etc/iptables/rules.v4`

Agregamos la siguiente línea en el archivo de configuración de iptables

por ejemplo si queremos bloquear todo el tráfico de Rusia agregamos la siguiente línea

`-A INPUT -m geoip --src-cc RU -j REJECT`

```
# Reject all other inbound - default deny u
-A INPUT -j REJECT
-A FORWARD -j REJECT
-A INPUT -m geoip --src-cc RU -j REJECT
COMMIT
```

Reiniciamos iptables para que los cambios surjan efecto con

`systemctl restart iptables`

y verificamos el estado de iptables con

`systemctl status iptables`

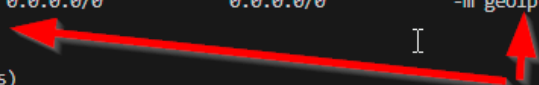
```
root@maquina1:~# systemctl status iptables
● netfilter-persistent.service - netfilter persistent configuration
   Loaded: loaded (/lib/systemd/system/netfilter-persistent.service; enabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/netfilter-persistent.service.d
            └─iptables.conf
   Active: active (exited) since Sat 2025-07-26 14:39:02 -04; 15s ago
     Docs: man:netfilter-persistent(8)
   Process: 8769 ExecStart=/usr/sbin/netfilter-persistent start (code=exited, status=0/SUCCESS)
   Main PID: 8769 (code=exited, status=0/SUCCESS)
      CPU: 16ms

jul 26 14:39:02 maquina1 systemd[1]: Starting netfilter-persistent.service - netfilter persistent configu>
jul 26 14:39:02 maquina1 netfilter-persistent[8771]: run-parts: executing /usr/share/netfilter-persistent>
jul 26 14:39:02 maquina1 netfilter-persistent[8771]: run-parts: executing /usr/share/netfilter-persistent>
jul 26 14:39:02 maquina1 systemd[1]: Finished netfilter-persistent.service - netfilter persistent configu>
lines 1-14/14 (END)
```


y verificamos que se está bloqueando el tráfico en este caso el de Rusia con

`iptables -vnL`

```
root@maquina1:~# iptables -vnL
Chain INPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
    0    0 ACCEPT     0    --  lo     *        0.0.0.0/0         0.0.0.0/0
    0    0 REJECT     0    --  !lo    *        0.0.0.0/0         127.0.0.0/8      reject-with icmp-po
rt-unreachable
  106  5896 ACCEPT     0    --  *      *        0.0.0.0/0         0.0.0.0/0      state RELATED,ESTAB
LISHED
    0    0 ACCEPT     17   --  *      *        0.0.0.0/0         0.0.0.0/0      udp dpt:51820
    0    0 ACCEPT     6    --  *      *        0.0.0.0/0         0.0.0.0/0      state NEW tcp dpt:2
2
    0    0 ACCEPT     1    --  *      *        0.0.0.0/0         0.0.0.0/0      icmp type 8
    1   36 LOG      0    --  *      *        0.0.0.0/0         0.0.0.0/0      limit: avg 5/min bu
rst 5 LOG flags 0 level 7 prefix "iptables denied: "
    1   36 REJECT     0    --  *      *        0.0.0.0/0         0.0.0.0/0      reject-with icmp-po
rt-unreachable
    0    0 REJECT     0    --  *      *        0.0.0.0/0         0.0.0.0/0      -m geoip --source-c
ountry RU reject-with icmp-port-unreachable
Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
    0    0 REJECT     0    --  *      *        0.0.0.0/0         0.0.0.0/0      reject-with icmp-po
rt-unreachable
Chain OUTPUT (policy ACCEPT 0 packets, 0 bytes)
  pkts bytes target     prot opt in     out     source            destination
   79  7596 ACCEPT     0    --  *      *        0.0.0.0/0         0.0.0.0/0
root@maquina1:~#
```



13. HARDENING DE SSH

AUTENTICACIÓN DE USUARIO CON LLAVE PÚBLICA Y CONTRASEÑA

Paso 1

Generar claves SSH en ambas máquinas

Ejecutar como usuario normal (**ruben1**) en ambas máquinas:

`ssh-keygen -t rsa -b 4096` (opcionalmente pide ingresar una frase para la contraseña)

```
ruben1@maquina2:~$ ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (/home/ruben1/.ssh/id_rsa):
Created directory '/home/ruben1/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ruben1/.ssh/id_rsa
Your public key has been saved in /home/ruben1/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:joEmr6oWxjx17CxuLJN7omN8AfWDFwUwUI18rDOF2YQ ruben1@maquina2
The key's randomart image is:
+---[RSA 4096]-----+
|  .oo@.                |
|  ..Eo*               |
|   . +=               |
|  . .oo o             |
|   + + O S            |
|  . O . *             |
| o o B o .            |
|  * B =               |
| *o=.B               |
+----[SHA256]-----+
ruben1@maquina2:~$
```

Paso 2

Preparar el directorio `.ssh` en ambas máquinas

<code>mkdir -p ~/.ssh</code>	# Crear directorio si no existe
<code>touch ~/.ssh/authorized_keys</code>	# Crear archivo vacío
<code>chmod 700 ~/.ssh</code>	# Permisos solo para el usuario
<code>chmod 600 ~/.ssh/authorized_keys</code>	# Permisos restrictivos

```
ruben1@maquina1:~$ mkdir -p ~/.ssh
ruben1@maquina1:~$ touch ~/.ssh/authorized_keys
ruben1@maquina1:~$ chmod 700 ~/.ssh
ruben1@maquina1:~$ chmod 600 ~/.ssh/authorized_keys
```


Paso 3

Copiar la clave pública de maquina1 a maquina2 y viceversa

En maquina1 (como ruben1):

ssh-copy-id ruben1@192.168.0.36

```
ruben1@maquina1:~$ ssh-copy-id ruben1@192.168.0.36
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ruben1/.ssh/id_rsa.pub"
The authenticity of host '192.168.0.36 (192.168.0.36)' can't be established.
ED25519 key fingerprint is SHA256:aNB0D8vVs2FpiJf+sLNau5Ctng59xD6XD3QRivogJHw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/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 prompted now it is to install the new keys
ruben1@192.168.0.36's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ruben1@192.168.0.36'"
and check to make sure that only the key(s) you wanted were added.

ruben1@maquina1:~$
```

En maquina2 (como ruben1):

ssh-copy-id ruben1@192.168.0.35

```
ruben1@maquina2:~$ ssh-copy-id ruben1@192.168.0.35
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/ruben1/.ssh/id_rsa.pub"
The authenticity of host '192.168.0.35 (192.168.0.35)' can't be established.
ED25519 key fingerprint is SHA256:aNB0D8vVs2FpiJf+sLNau5Ctng59xD6XD3QRivogJHw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
/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 prompted now it is to install the new keys
ruben1@192.168.0.35's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'ruben1@192.168.0.35'"
and check to make sure that only the key(s) you wanted were added.

ruben1@maquina2:~$
```

Paso 4

Configurar el servidor SSH (sshd)

Cambiar a root y editar el archivo con un editor de texto, en este caso con nano ejecutamos `nano /etc/ssh/sshd_config`

```
ruben1@maquina1:~$ su - root
Contraseña:
root@maquina1:~# nano /etc/ssh/sshd_config
```

Asegurarse de que estas líneas estén en el archivo

`PubkeyAuthentication yes` # Habilitar autenticación por clave

`PasswordAuthentication yes` # Habilitar contraseña (opcional)

`AuthenticationMethods publickey,password` # Requerir AMBOS

```
# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes
AuthenticationMethods publickey,password

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
```

DESCOMENTAR

AGREGAR

DESCOMENTAR

Guardar los cambios y reiniciamos el servicio con:
`systemctl restart sshd`

Paso 5

Probar la conexión

Cambiamos a usuario nuevamente, en este caso **ruben1** y nos conectamos por ssh donde nos solicita la frase de contraseña de la clave privada (que en mi caso si configure en el paso 1) y ademas la contraseña del usuario en la maquina.

```
ssh -i ~/.ssh/id_rsa ruben1@192.168.0.36
```

Y podemos ver que tenemos conexión por ssh de forma bidireccional.

Desde maquina1 hacia maquina2

```
ruben1@maquina1:~$ ssh -i ~/.ssh/id_rsa ruben1@192.168.0.36
Enter passphrase for key '/home/ruben1/.ssh/id_rsa':
ruben1@192.168.0.36's password:
Linux maquina2 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Jul 26 15:56:18 2025 from 192.168.0.31
ruben1@maquina2:~$
```

Y desde maquina2 hacia maquina1

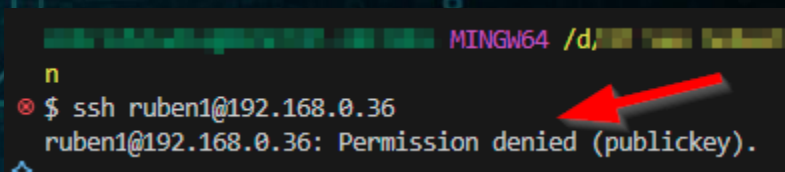
```
ruben1@maquina2:~$ ssh -i ~/.ssh/id_rsa ruben1@192.168.0.35
Enter passphrase for key '/home/ruben1/.ssh/id_rsa':
ruben1@192.168.0.35's password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sat Jul 26 15:55:48 2025 from 192.168.0.31
ruben1@maquina1:~$
```

AGREGAR OTROS EQUIPOS, EJEMPLO WINDOWS ANFITRION

Lo realizado anteriormente conecta ambas maquinas, pero al cerrar el laboratorio y al volver a intentar conectar por SSH, lo cierto es que no fue posible porque de acuerdo a la configuracion realizada, la autenticacion para acceder a las maquinas quedo con la autenticacion por password y llave publica la cual en el equipo anfitrión no ha sido configurada.



```
MINGW64 /d/...
n
$ ssh ruben1@192.168.0.36
ruben1@192.168.0.36: Permission denied (publickey).
```

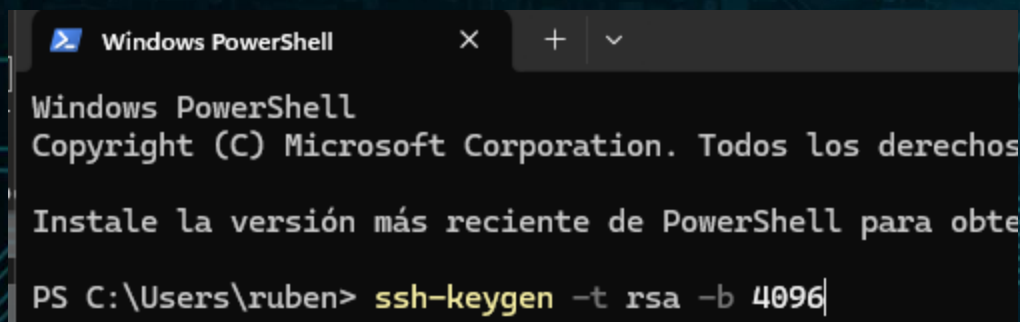
Por lo tanto, para poder conectar el equipo windows anfitrión via ssh a las maquinas se debe hacer lo siguiente:

Paso 1.

Generar llave publica para ssh en windows

En caso de que no tengamos una llave publica, generamos una en Powershell con

```
ssh-keygen -t rsa -b 4096
```



```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos los derechos reservados.

Instale la versión más reciente de PowerShell para obtener las mejores
funcionalidades.

PS C:\Users\ruben> ssh-keygen -t rsa -b 4096
```


En caso de que exista una preguntará si se quiere sobrescribir el archivo existente, en este caso le colocare que sí.

```
PS C:\Users\ruben> ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\ruben/.ssh/id_rsa):
C:\Users\ruben/.ssh/id_rsa already exists.
Overwrite (y/n)? y|
```

Opcionalmente (y más seguridad) te solicita que ingreses una frase que acompañara al inicio de sesión.

```
PS C:\Users\ruben> ssh-keygen -t rsa -b 4096
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\ruben/.ssh/id_rsa):
C:\Users\ruben/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\ruben/.ssh/id_rsa
Your public key has been saved in C:\Users\ruben/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:28jjeBjLLu...
The key's randomart image is:
+---[RSA 4096]---+
|  .                |
|  . o              |
|  = .              |
|  o.=              |
|+oo+ . S          |
|*X*   .. +        |
|%=+. . += .       |
|B*ooo +o..         |
|OBEo.+o..         |
+---[SHA256]-----+
PS C:\Users\ruben> ^C
PS C:\Users\ruben> |
```

Y con esto se acaba de generar la llave pública que es básicamente lo que deberías hacer en cualquier equipo Windows para generar una.

Paso 2.

Agregar la clave publica a las máquinas virtuales Debian

Antes de continuar, debemos entrar a las máquinas virtuales por vmware (o el virtualizador que prefieras), ya encendidas previamente y como usuario **root** vamos a cambiar temporalmente la configuración del archivo **sshd_config**, para esto con el editor de texto en mi caso nano hacemos lo siguiente:

```
nano /etc/ssh/sshd_config
```

y en el archivo vamos a comentar las líneas que solicita la autenticación por llave publica y método de autenticación, como muestra la imagen siguiente:

```
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
#AuthenticationMethods publickey,password

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
KbdInteractiveAuthentication no
```

comentamos esas 2

esta la dejamos

Una vez realizados los cambios, guardamos y cerramos.

Esto lo hacemos para poder ingresar via ssh, en mi caso desde vscode, que es mucho más sencillo de manejar que la terminal de las maquinas en vmware, sobre todo para copiar y pegar comandos.

Antes de continuar abriremos powershell en el equipo Windows anfitrión y con

`cat C:\Users\“reemplaza_tu_usuario\.ssh\id_rsa.pub`

ya podemos ver la llave publica que la copiaremos más adelante.

```
PS C:\Users\ruben> cat C:\Users\ruben\.ssh\id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCNMrDaB334IC/LYzP4VpmMxY3//3aoiy1EdByQxWoIksiBpD+tG0kGG2Md3nDKcgNaCCrGxXLE2jv45M9P
ZQ== hibridstudio@DESKTOP-28FJ8UL
PS C:\Users\ruben>
```

Paso 3.

Conexión por ssh

A continuación ya podemos conectarnos vía ssh; como ya mencioné en mi caso lo hago desde la **terminal de vscode**, aunque si quieres puedes hacerlo desde cualquier otra aplicación, como **powershell** o **mobaxterm** por citar un ejemplo.

Ya en la **maquina1** (después hay que hacer lo mismo en la 2) **y como usuario** (en mi caso **ruben1**), insisto no como **root**, vamos a hacer lo que viene a continuación.

Solo en el caso de que no existe (en este caso el directorio si existe porque lo habíamos creado anteriormente en el paso 2 del punto 13.1 de este informe) deberías crear el directorio:

```
mkdir -p ~/.ssh
```

y a continuación agregamos de forma manual la llave publica que generamos en Windows con

```
echo "ssh-rsa AAAAB3NzaC1yc2E... usuario@windows" >>
```

```
~/.ssh/authorized_keys #obvio que hay que reemplazar por tu llave.
```

```
ruben1@maquina1:~$ echo ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCNrDaB334IC/LYzP4VpmMxY3//3aoiy1EdByQxwoIksIB
pD+ti  T7EAuD/y+WoGKRsv
G9dB  LD0g214oBwGRZ9Py
n50h  qCiPvaxlGn0FcTsr
h5rZ  3Z6RDUL+/96rmor9
hrdw  oge8Lp8a/2wH76UB
nzaH  SN63ZQ== hibrids
tudic@DESKTOP-28FJ8UL >> ~/.ssh/authorized_keys
ruben1@maquina1:~$
```


Con `cat ~/.ssh/authorized_keys` podemos ver como se ha agregado la llave generada en Windows al archivo de llaves

```
ruben1@maquina1:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQADV7r4bIvyVG3Kex+WPazASWRwJECg2clw8StywjDomf174C0I4sXo+ZQeH9kcIjSiqzw+Yp
h5BFs... i/tc08prvs
oqDnL... 2VZRM90+bP
9uCDs... RYtx/gotwR
AukIK... K9zakIXf5e
qLrc+... 6sNqS8ln4d
uTKwdw0Foo/eQddyDH4ZVDWx+Qhk0qos1whEPCaNTl61dFVDXls4ylbhaTBwT6hjKVL0If95ZQ== ruben1@maquina2
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQACnMrDaB334IC/LYzP4VpmMxY3//3aoiy1EdByQxwoIksibPd+tg0kGG2Md3nDKcgNaCCrG
xX1E... zQxfEP2bbNLiV8z+
sSzy... CaZUeuYGyXzf3b/a
PSwi... /0tWZRuS77ShU8
8i9Z... Ztyd60JdyKIQC6x
SGDYH/NHEXSSwQ0+3/XLCC/F4CD14H/dababLPexCqD4nZCK3b1R0n9paxS280T/Pc49oge8Lp8a/ZwH/oubnzaHegSnmC... aw6ZETZkyC1
gXC9yMXTZ8x38bBdJNIBqv78Jyq3EnXFLlq5wM4QaUTWjxIqR87P319+xHf6aPnQoF1ZSN63ZQ== hibridstudio@DESKTOP-28FJ8UL
ruben1@maquina1:~$
```

Ahora cambiamos a usuario **root** nuevamente y vamos a editar nuevamente nuestro archivo de configuración de sshd

`nano /etc/ssh/sshd_config`

y dejamos la configuración como lo muestra la imagen a continuación:

```
#Port 22
#AddressFamily any
ListenAddress 0.0.0.0
#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

PubkeyAuthentication yes
AuthenticationMethods publickey,password

# Expect .ssh/authorized_keys2 to be disregarded by default in future.
#AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody

# For this to work you will also need host keys in /etc/ssh/ssh_known_hosts
#HostbasedAuthentication no
# Change to yes if you don't trust ~/.ssh/known_hosts for
# HostbasedAuthentication
#IgnoreUserKnownHosts no
# Don't read the user's ~/.rhosts and ~/.shosts files
#IgnoreRhosts yes

# To disable tunneled clear text passwords, change to no here!
PasswordAuthentication yes
#PermitEmptyPasswords no
```

Guardamos y cerramos... y reiniciamos la maquina con reboot.

Ahora desde **VSCode** podemos iniciar sesión nuevamente, situación que al principio no permitía y eso es todo, ya podemos seguir testeando desde una terminal “más amigable” que la de Debian.



```
MINGW64 /d/
$ ssh ruben1@192.168.0.35
Enter passphrase for key '/c/Users/ruben/.ssh/id_rsa':
ruben1@192.168.0.35's password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Jul 27 15:27:59 2025 from 192.168.0.31
ruben1@maquina1:~$
```

Para la maquina2 hay que hacer los mismos pasos.

MEDIDAS DE HARDENING ADICIONALES

1. Cambiar puerto ssh (ejemplo 2222)

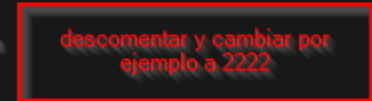
Como root

`nano /etc/ssh/sshd_config`

Agregamos al archivo

Port 202 (puede ser cualquiera hasta 65536)

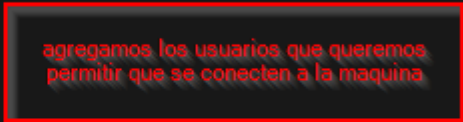
```
# possible, but leave them commented. Uncommented or  
# default value.  
  
Include /etc/ssh/sshd_config.d/*.conf  
  
Port 2222  
#AddressFamily any  
ListenAddress 0.0.0.0  
#ListenAddress ::  
  
#HostKey /etc/ssh/ssh_host_rsa_key  
#HostKey /etc/ssh/ssh_host_ecdsa_key  
#HostKey /etc/ssh/ssh_host_ed25519_key
```



2. Permitir solo ciertos usuarios

AllowUsers ruben1

```
#UseDNS no  
#PidFile /run/sshd.pid  
#MaxStartups 10:30:100  
#PermitTunnel no  
#ChrootDirectory none  
#VersionAddendum none  
AllowUsers ruben1  
  
# no default banner path  
#Banner none  
  
# Allow client to pass locale environment variables  
AcceptEnv LANG LC_*  
  
# override default of no subsystems  
Subsystem sftp /usr/lib/openssh/sftp-server
```



3. Actualizar iptables:

En este punto quiero aclarar que lo mejor es hacer lo siguiente de forma manual debido que al realizarlo de forma automática, la regla de aceptar el tráfico por el puerto 2222 en iptables se agrega al final, lo que en este caso trae problemas porque las reglas de iptables, se procesan en orden de arriba hacia abajo. En cuanto un paquete coincide con una regla, esa regla se aplica y las demás no se evalúan para ese paquete. Por lo tanto, en este caso, cualquier intento de conexión al puerto 2222 es interceptado y rechazado por la regla `-A INPUT -j REJECT` antes de que se alcance la regla que lo permite.

Por lo tanto, como usuario `root`, editamos con `nano /etc/iptables/rules.v4`

Y agregamos la siguiente regla, antes de la regla que rechaza

`iptables -A INPUT -p tcp --dport 2222 -j ACCEPT`

```
ruben1@maquina1:~$ cat /etc/iptables/rules.v4
# Generated by iptables-save v1.8.9 (nf_tables) on Sun Jul 27 17:25:01 2025
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
-A INPUT -i lo -j ACCEPT
-A INPUT -d 127.0.0.0/8 ! -i lo -j REJECT --reject-with icmp-port-unreachable
-A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
-A INPUT -p udp -m udp --dport 51820 -j ACCEPT
-A INPUT -p tcp -m state --state NEW -m tcp --dport 22 -j ACCEPT
-A INPUT -p icmp -m icmp --icmp-type 8 -j ACCEPT
-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied: " --log-level 7
-A INPUT -p tcp -m tcp --dport 2222 -j ACCEPT
-A INPUT -p tcp -m tcp --dport 202 -j ACCEPT
-A INPUT -j REJECT --reject-with icmp-port-unreachable
-A INPUT -m geoip --source-country RU -j REJECT --reject-with icmp-port-unreachable
-A FORWARD -j REJECT --reject-with icmp-port-unreachable
-A OUTPUT -j ACCEPT
COMMIT
# Completed on Sun Jul 27 17:25:01 2025
ruben1@maquina1:~$
```

ESTA REGLA DEBE QUEDAR POR ENCIMA
DE LA QUE RECHAZA

4. Reiniciar servicios:

```
systemctl restart sshd iptables
```

```
root@maquina1:~# systemctl restart sshd iptables
root@maquina1:~#
```

5. Nos conectamos por ssh al nuevo puerto 2222

Podemos ver que ahora la conexión que hacíamos previamente falla y ahora solo nos permite conectarnos por solo por el puerto indicado (2222) y solo con el usuario indicado.

```

MINGW64 /C:/Users/ruben/.ssh
C:\Users\ruben> ssh ruben1@192.168.0.35
ssh: connect to host 192.168.0.35 port 22: Connection refused

C:\Users\ruben> ssh -p 2222 ruben1@192.168.0.35
Enter passphrase for key '/c/Users/ruben/.ssh/id_rsa':
ruben1@192.168.0.35's password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Sun Jul 27 17:36:53 2025 from 192.168.0.31
ruben1@maquina1:~$

```


Creación de nuevos usuarios

En este caso creare un nuevo usuario “**usuario_nuevo**” con permisos de root en **maquina1**. Para esto, voy a instalar sudo, que por defecto no viene instalado en Debian12.

Como usuario root

`su - root`

actualizamos primero la lista de paquetes con

`apt update`

y a continuación instalamos el paquete sudo con

`apt install sudo`

```
root@maquina1:~# apt install sudo
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias... Hecho
Leyendo la información de estado... Hecho
sudo ya está en su versión más reciente (1.9.13p3-1+deb12u2).
fijado sudo como instalado manualmente.
0 actualizados, 0 nuevos se instalarán, 0 para eliminar y 0 no actualizados.
root@maquina1:~#
```

Para crear el nuevo usuario ingresamos

`Adduser usuario_nuevo`

Nos pedirá ingresar contraseña y otra información, opcionalmente se pueden dejar todos los campos en blanco.

```
root@maquina1:~# adduser usuario_nuevo
Añadiendo el usuario `usuario_nuevo' ...
Añadiendo el nuevo grupo `usuario_nuevo' (1001) ...
Adding new user `usuario_nuevo' (1001) with group `usuario_nuevo (1001)' ...
Creando el directorio personal `/home/usuario_nuevo' ...
Copiando los ficheros desde `/etc/skel' ...
Nueva contraseña:
Vuelva a escribir la nueva contraseña:
passwd: contraseña actualizada correctamente
Cambiando la información de usuario para usuario_nuevo
Introduzca el nuevo valor, o pulse INTRO para usar el valor predeterminado
Nombre completo []:
Número de habitación []:
Teléfono del trabajo []:
Teléfono de casa []:
Otro []:
¿Es correcta la información? [S/n] s
Adding new user `usuario_nuevo' to supplemental / extra groups `users' ...
Añadiendo al usuario `usuario_nuevo' al grupo `users' ...
root@maquina1:~#
```

Para que "**usuario_nuevo**" pueda ejecutar comandos con privilegios de root usando sudo, debemos añadirlo al grupo sudo:

```
usermod -aG sudo usuario_nuevo
```

Se puede probar si el usuario tiene permisos de root cerrando la sesión de root y luego cambiando al nuevo usuario.

```
exit
```

```
su - usuario_nuevo
```

Una vez logueado como **usuario_nuevo**, intentamos ejecutar un comando con sudo:

```
sudo apt update
```

Nos pedirá la contraseña de **usuario_nuevo**. Si el comando se ejecuta sin errores de permiso, el usuario tiene privilegios de root correctamente.

```
ruben1@maquina1:~$ su - usuario_nuevo
Contraseña:
usuario_nuevo@maquina1:~$
```


Ahora debemos agregar nuestro **usuario_nuevo** a los usuarios permitidos en la configuración de ssh para poder iniciar sesión via ssh.

Como **root** ingresamos al archivo y en la línea que dice AllowUsers agregamos a **usuario_nuevo**

`nano /etc/ssh/sshd_config`

```
#ChrootDirectory none
#VersionAddendum none
AllowUsers ruben1 usuario_nuevo
```

A continuación nos cambiamos a "**usuario_nuevo**" y vamos a agregar la llave publica como lo hiciéramos en el **paso 3** del título "**AGREGAR OTROS EQUIPOS, EJEMPLO WINDOWS ANFITRION**".

Con `mkdir -p ~/.ssh` creamos el directorio

y a continuación agregamos de forma manual la llave publica que generamos en Windows con

```
echo "ssh-rsa AAAAB3NzaC1yc2E... usuario@windows" >>
```

```
~/.ssh/authorized_keys #obvio que hay que reemplazar por tu llave.
```

Si estas desde otro equipo Windows, los pasos para generar la llave ya fueron explicados previamente. Pero siempre nos podemos apoyar en la IA para buscar métodos sobre cómo hacerlo.

Con `cat ~/.ssh/authorized_keys` podemos ver como se ha agregado la llave generada en Windows al archivo de llaves

```
usuario_nuevo@maquina1:~$ mkdir -p ~/.ssh
usuario_nuevo@maquina1:~$ echo ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCNMrDaB334IC/LYzP4VpmMxY3//3aoiy1EdByQx
WoIh...
usuario_nuevo@maquina1:~$ cat ~/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQCNMrDaB334IC/LYzP4VpmMxY3//3aoiy1EdByQxKcgNaCCrG
xx1E2jv45M...
usuario_nuevo@maquina1:~$
```

Para comprobar que todo funciona reiniciamos la máquina. Y como se puede apreciar en la imagen el nuevo usuario se puede conectar por el puerto 2222 utilizando autenticación de llave publica con frase de seguridad y contraseña de usuario.

```
$ ssh -p 2222 usuario_nuevo@192.168.0.35
Enter passphrase for key '/c/Users/ruben/.ssh/id_rsa':
usuario_nuevo@192.168.0.35's password:
Linux maquina1 6.1.0-37-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.140-1 (2025-05-22) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Jul 28 17:15:54 2025 from 192.168.0.31
usuario_nuevo@maquina1:~$
```



Prohibir el acceso vía ssh como usuario root

Deshabilitar el inicio de sesión remoto para el usuario **root** es una practica de seguridad fundamental. Para esto vamos a editar como usuario **root** el archivo de configuración de ssh

```
sudo nano /etc/ssh/sshd_config
```

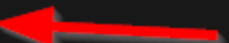
y en el interior vamos a cambiar esta línea

```
# Authentication:
#LoginGraceTime 2m
#PermitRootLogin prohibit-password
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10
■
```



Con este sencillo cambio **PermitRootLogin no**

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



Guardamos y reiniciamos el servicio ssh con

```
sudo systemctl restart sshd
```

Muy importante, ahora para realizar tareas administrativas se debe iniciar sesión como usuario normal (**ruben1** o **nuevo_usuario**) y utilizar **"sudo"** (instalado previamente) o utilizar **su - root**.