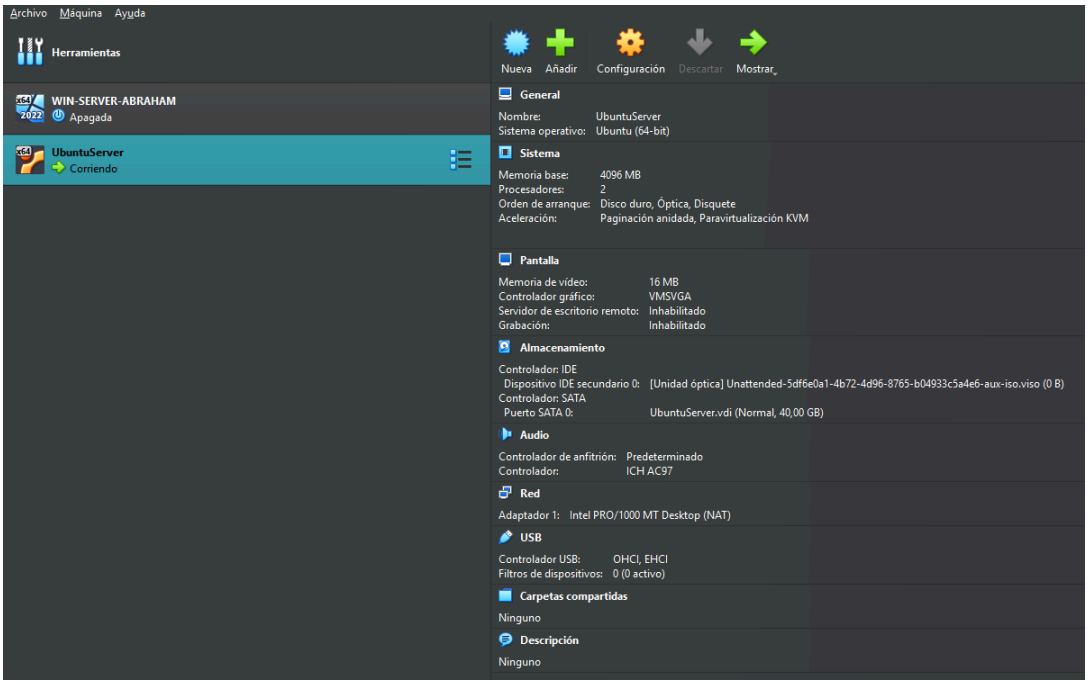
🏆 *Reto Día 7: Despliegue y Preparación de un Servidor Linux para Producción* 🐧🔧📡

Abraham Caamaño Martínez

🚀 Fase 1: Instalación del Entorno de Pruebas



🚀 Fase 2: Configuración del Servidor

✅Configurar el sistema con un **usuario administrador seguro**:

El mismo Virtualbox crea un usuario administrador.

✅ Asignar una **IP estática** en la red local y probar conectividad con ping:

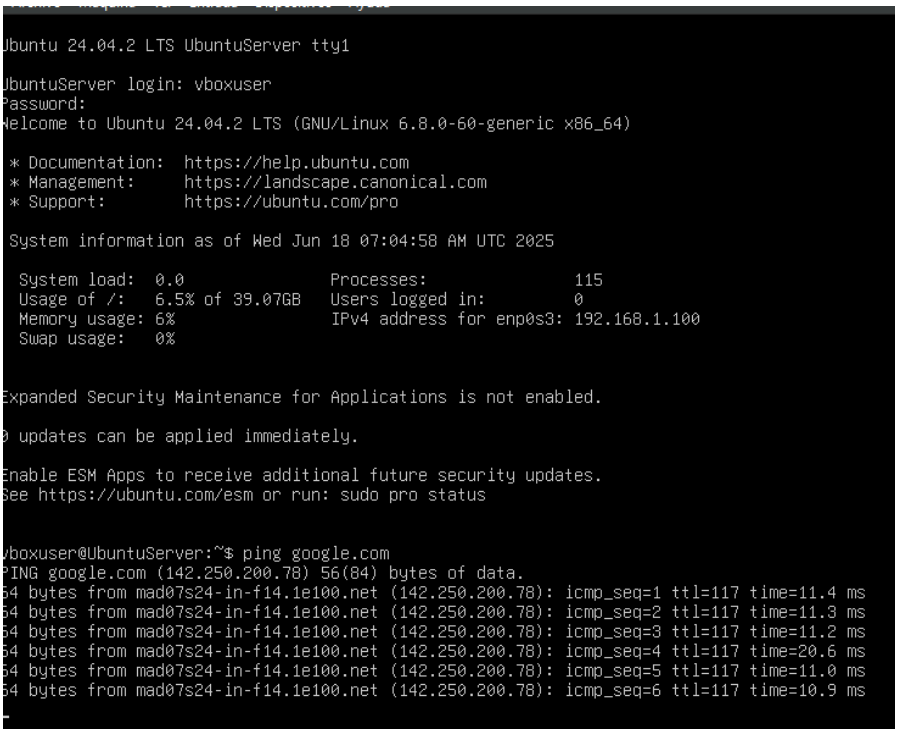
1. Editamos el archivo de configuración en **/etc/netplan/** de la siguiente manera con el comando:

sudo nano /etc/netplan/"nombre archivo de configuracion".yaml

```
network:
  version: 2
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [192.168.1.100/24]
      routes:
        - to: default
          via: 192.168.1.1
      nameservers:
        addresses:
          - 8.8.8.8
          - 8.8.4.4
```

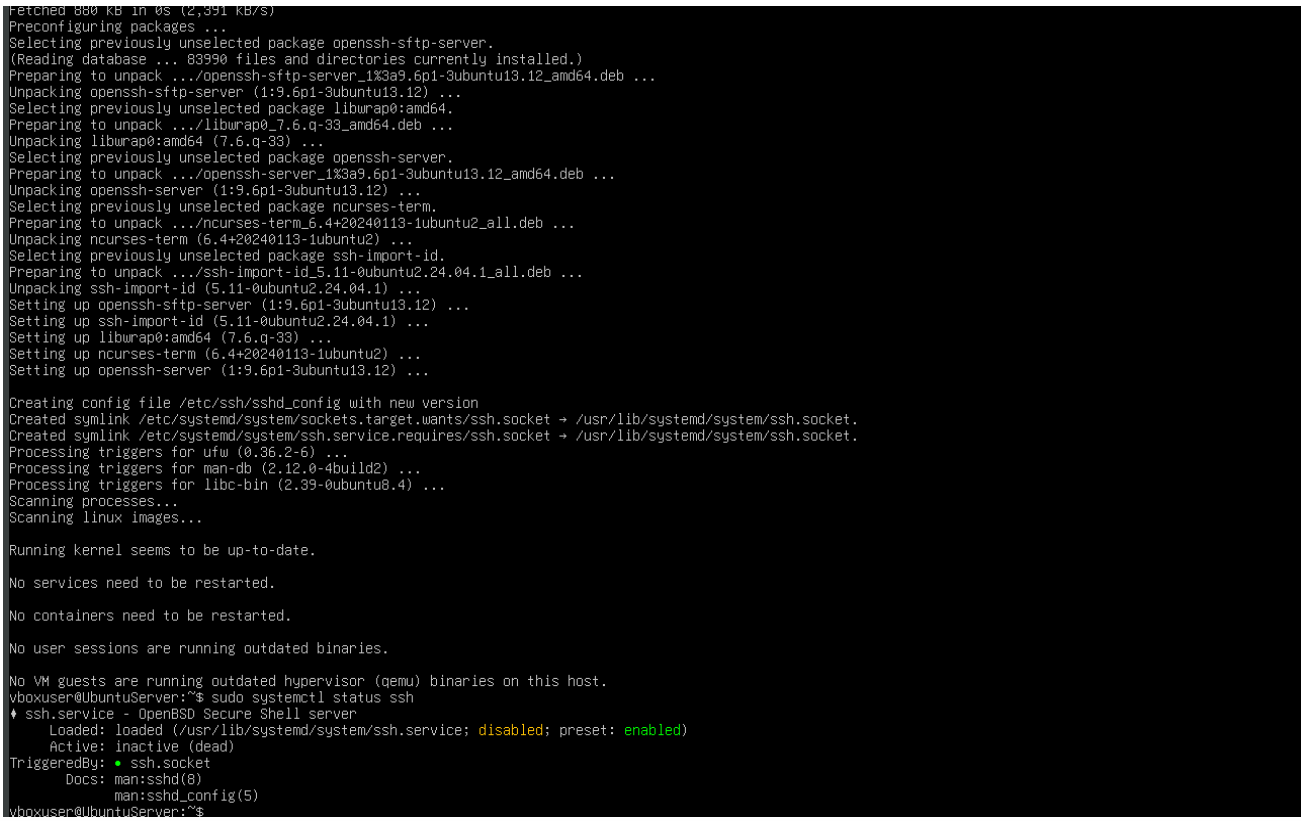
2. Aplicamos con:
   sudo netplan apply

3. Hacemos ping con:
   ping google.com



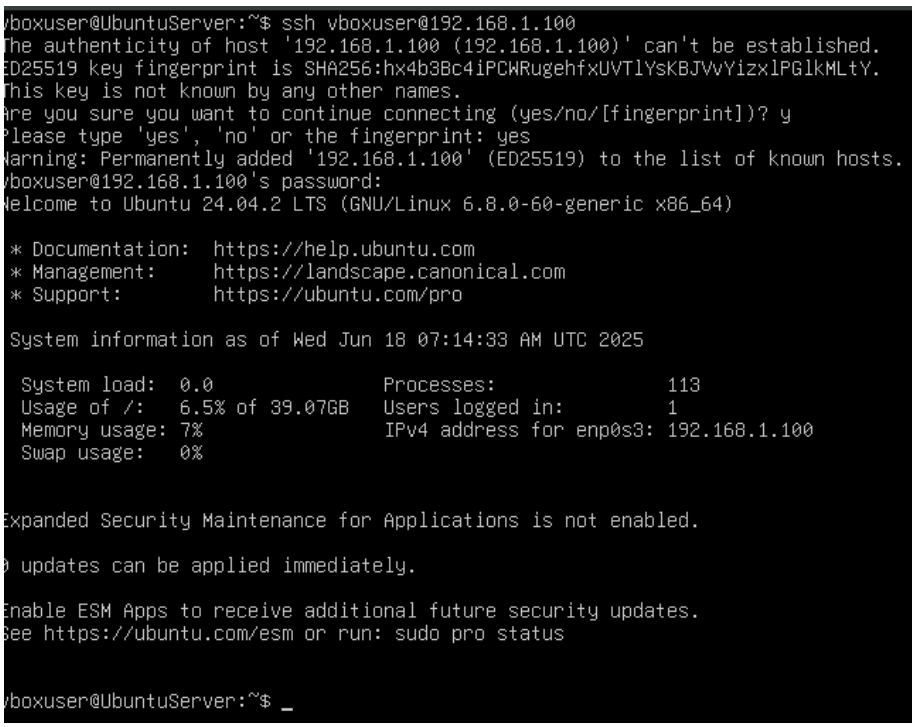
✅ Habilitar el acceso remoto mediante SSH y verificar conexión.

1. Instalar el servidor SSH con el comando:
   sudo apt install openssh-server



2. Ver si el servicio esta activo:
   sudo systemctl status ssh

3. Nos conectamos con el comando:
   ssh vboxuser@192.168.1.100

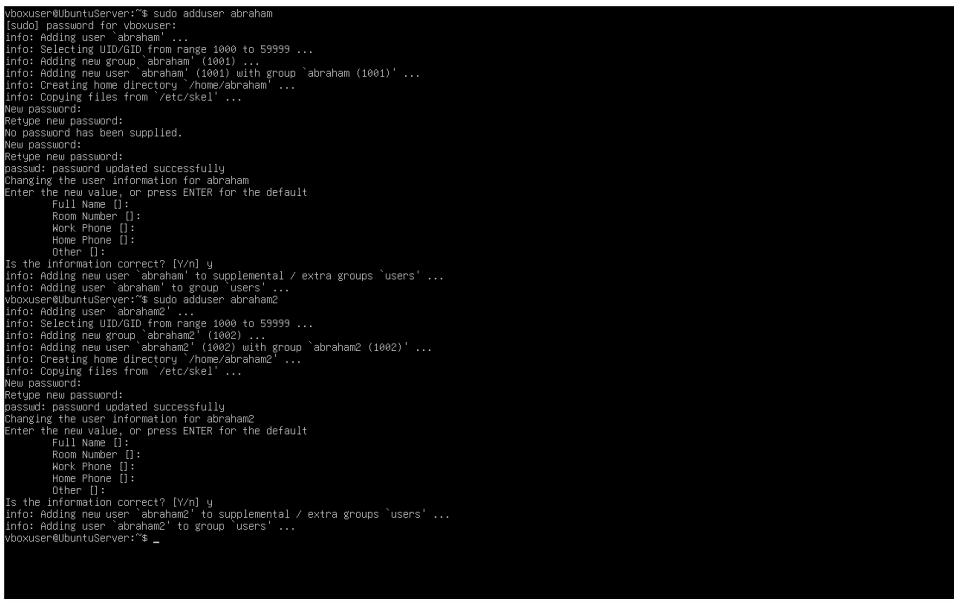


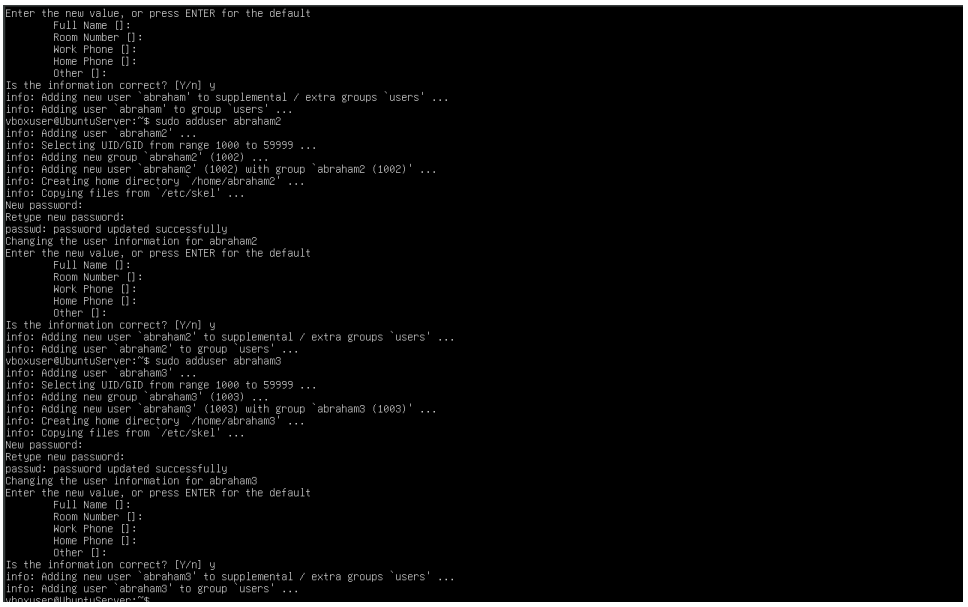
🚀 Fase 3: Gestión de usuarios y seguridad

✅ Crear tres usuarios con diferentes permisos en el servidor.

1. Creamos los usuarios de la siguiente forma:

sudo adduser abraham
sudo adduser abraham2
sudo adduser abraham3





✅ Asignar permisos específicos a cada usuario y probar accesos.

(Los permisos se explican en la segunda parte del reto)

Conectamos el usuario abraham2 por ejemplo con:

ssh abraham2@192.168.1.100

✅ Configurar el firewall con UFW y permitir solo las conexiones necesarias.

1. Verificamos el estado del firewall con:

sudo ufw status

2. Denegamos todo el tráfico entrante y permitimos todo el saliente con:

sudo ufw default deny incoming
sudo ufw default allow outgoing

3. Permitimos el acceso SSH:

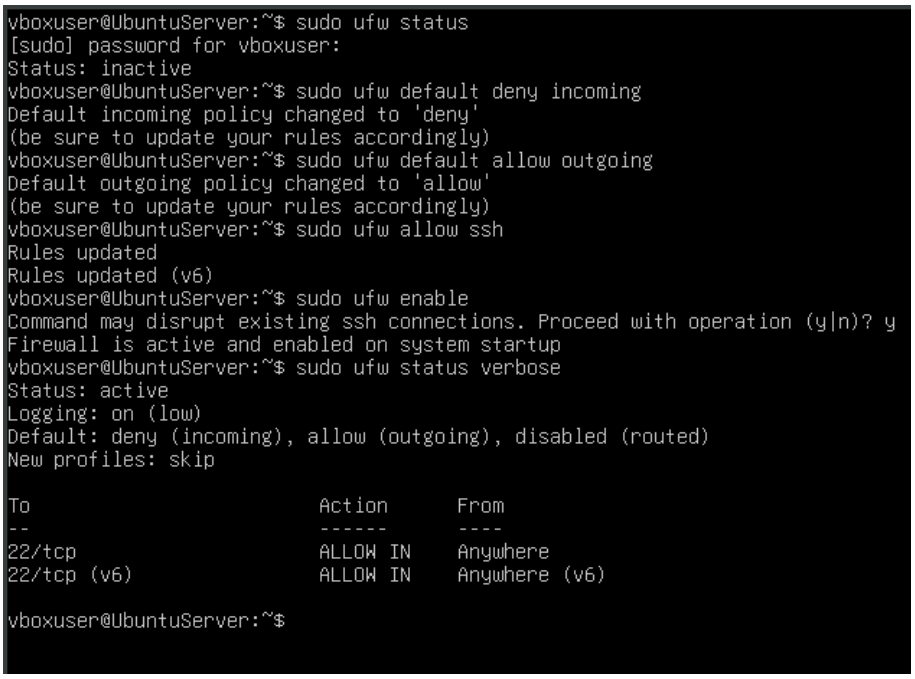
sudo ufw allow ssh (Permite el puerto 22)
sudo ufw allow http (Permite el puerto 80)
sudo ufw allow https (Permite el puerto 443)

4. Activamos el firewall:

sudo ufw enable

5. Verificamos las reglas que están activas:

sudo ufw status verbose

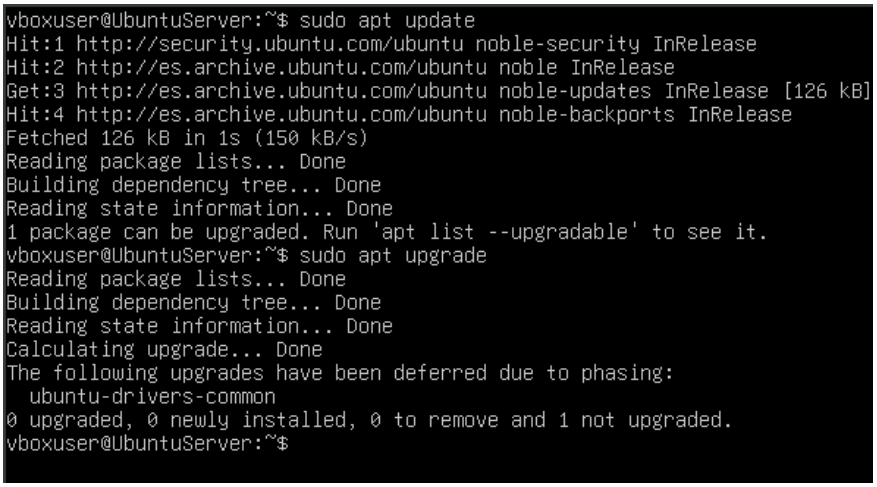


Parte 2

🚀 Fase 1: Instalación del sistema base

✅ Verificar que el sistema arranque sin errores y actualiza sus paquetes (apt update & & apt upgrade).

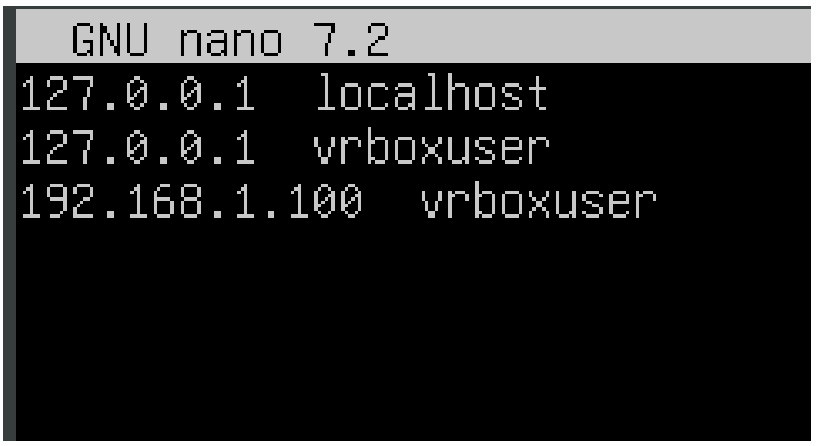
Actualizamos Ubuntu Server con:
sudo apt update && sudo apt upgrade -y



🚀 Fase 2: Configuración de red y acceso remoto

✅ Configurar el archivo /etc/hosts correctamente con el nombre del servidor.

Edite el archivo /etc/hosts con:
sudo nano /etc/hosts



Añadimos la linea:
192.168.1.100 vrboxuser

✅ Instalar y habilitar el servicio SSH.

(Ya realizado antes)

✅ Verificar la conexión remota desde otro sistema con ssh.

(Ya realizado antes)

🚀 Fase 3: Seguridad mínima obligatoria

✅ Configurar UFW

Permitiremos solo el trafico por puerto 22 (SSH) y puerto 80 (HTTP):

sudo ufw allow 22/tcp

sudo ufw allow 80/tcp

```
vboxuser@UbuntuServer:~$ sudo ufw allow 22/tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
vboxuser@UbuntuServer:~$ sudo ufw allow 80/tcp
Rule added
Rule added (v6)
vboxuser@UbuntuServer:~$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
22/tcp                     ALLOW       Anywhere
80/tcp                     ALLOW       Anywhere
22/tcp (v6)                ALLOW       Anywhere (v6)
80/tcp (v6)                ALLOW       Anywhere (v6)

vboxuser@UbuntuServer:~$ _
```

✅ Crear un nuevo usuario llamado desarrollador, con acceso limitado y sin permisos de superusuario.

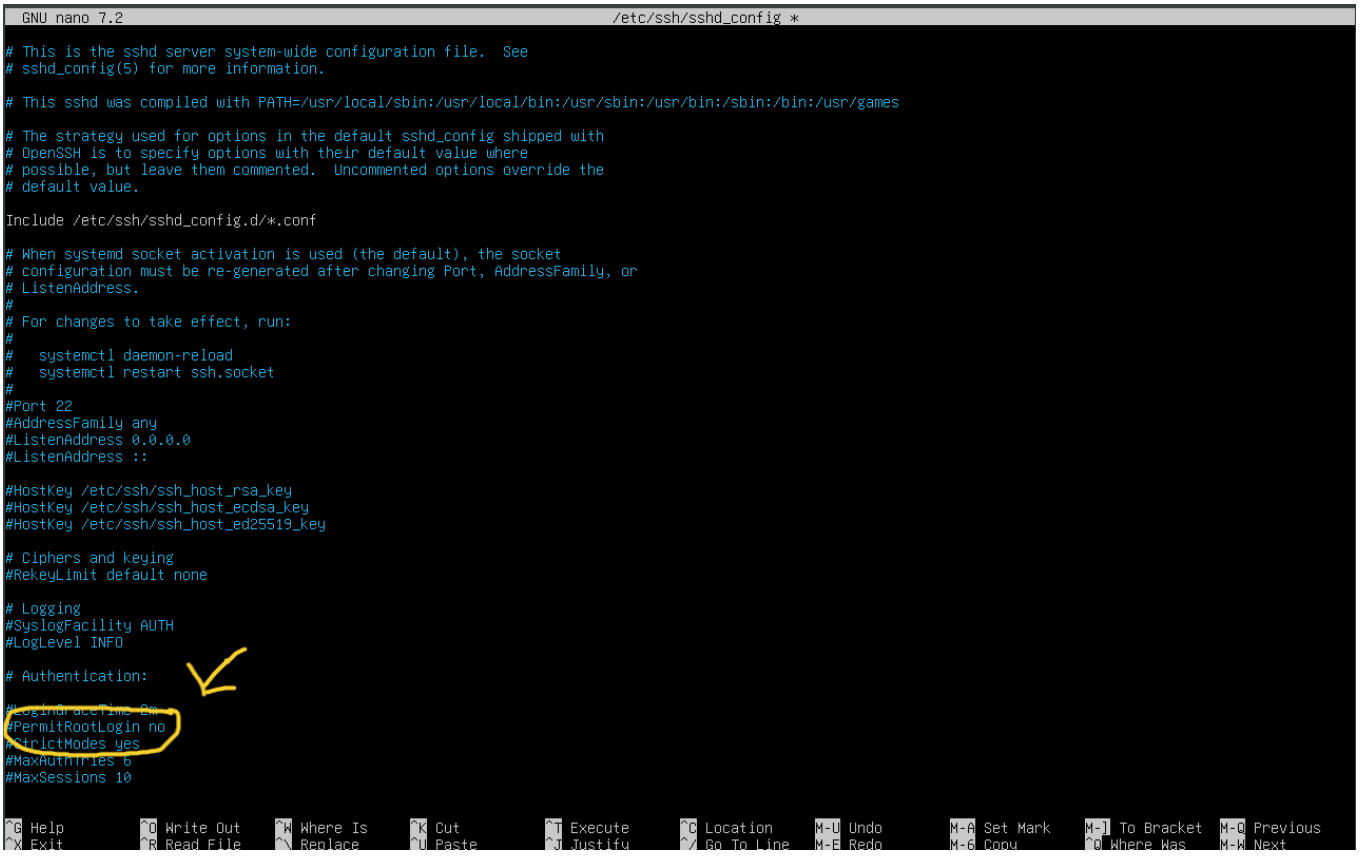
Creamos el usuario:
sudo adduser desarrollador

```
vboxuser@UbuntuServer:~$ sudo adduser desarrollador
info: Adding user `desarrollador' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `desarrollador' (1004) ...
info: Adding new user `desarrollador' (1004) with group `desarrollador (1004)' ..
info: Creating home directory `/home/desarrollador' ...
info: Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for desarrollador
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
info: Adding new user `desarrollador' to supplemental / extra groups `users' ...
info: Adding user `desarrollador' to group `users' ...
vboxuser@UbuntuServer:~$ _
```

✅ Cambiar el puerto por defecto de SSH a 2222 y reforzar la configuración (/etc/ssh/sshd_config).
✅ Desactivar el acceso SSH del usuario root.

Editamos el archivo de configuracion de SSH con:
sudo nano /etc/ssh/sshd_config

En el archivo,cambiamos donde dice ***#Port 22*** a ***#Port 2222***, y desactivaremos el acceso root poniendo ***"no"*** en ***PermitRootLogin***.
Quedaria algo asi:



Luego es importante que despues de que hagamos todo, permitamos el puerto 2222 con el siguiente comando:
sudo ufw allow 2222/tcp

```
vboxuser@UbuntuServer:~$ sudo ufw allow 2222/tcp
Rule added
Rule added (v6)
vboxuser@UbuntuServer:~$
```

Reiniciamos con:
sudo systemctl restart sshd

Y probamos que funcione correctamente:
ssh “usuario”@192.168.1.100 -p 2222

```
vboxuser@UbuntuServer:~$ ssh vboxuser@192.168.1.100 -p 2222
ssh: connect to host 192.168.1.100 port 2222: Connection refused
vboxuser@UbuntuServer:~$ _
```

🚀 Fase 4: Estructura de carpetas y servicios iniciales

✅ Crear una estructura de carpetas en /srv/

Creamos los directorios con:
sudo mkdir -p /srv/www /srv/repositorios /srv/docs

```
vboxuser@UbuntuServer:~$ sudo mkdir -p /srv/www /srv/repositorios /srv/docs
```

✅ Establecer permisos específicos:

Vamos a dar acceso de escritura para “desarrollador” con:

sudo chown -R desarrollador /srv/www

sudo chmod -R 775 /srv/www

Y acceso exclusivo a nuestro usuario administrador con:

sudo chown -R vboxuser /srv/repositorios

sudo chmod -R 770 /srv/repositorios

```
vboxuser@UbuntuServer:~$ sudo chown -R desarrollador /srv/www
vboxuser@UbuntuServer:~$ sudo chmod -R 775 /srv/www
vboxuser@UbuntuServer:~$ sudo chown -R vboxuser /srv/www
vboxuser@UbuntuServer:~$ sudo chown -R vboxuser /srv/repositorios
vboxuser@UbuntuServer:~$ sudo chmod -R 770 /srv/repositorios
vboxuser@UbuntuServer:~$
```

✅ Instalar el servidor web Apache2 o NGINX (a elegir) y colocar una página de prueba en /srv/www.

Instalamos NGINX:

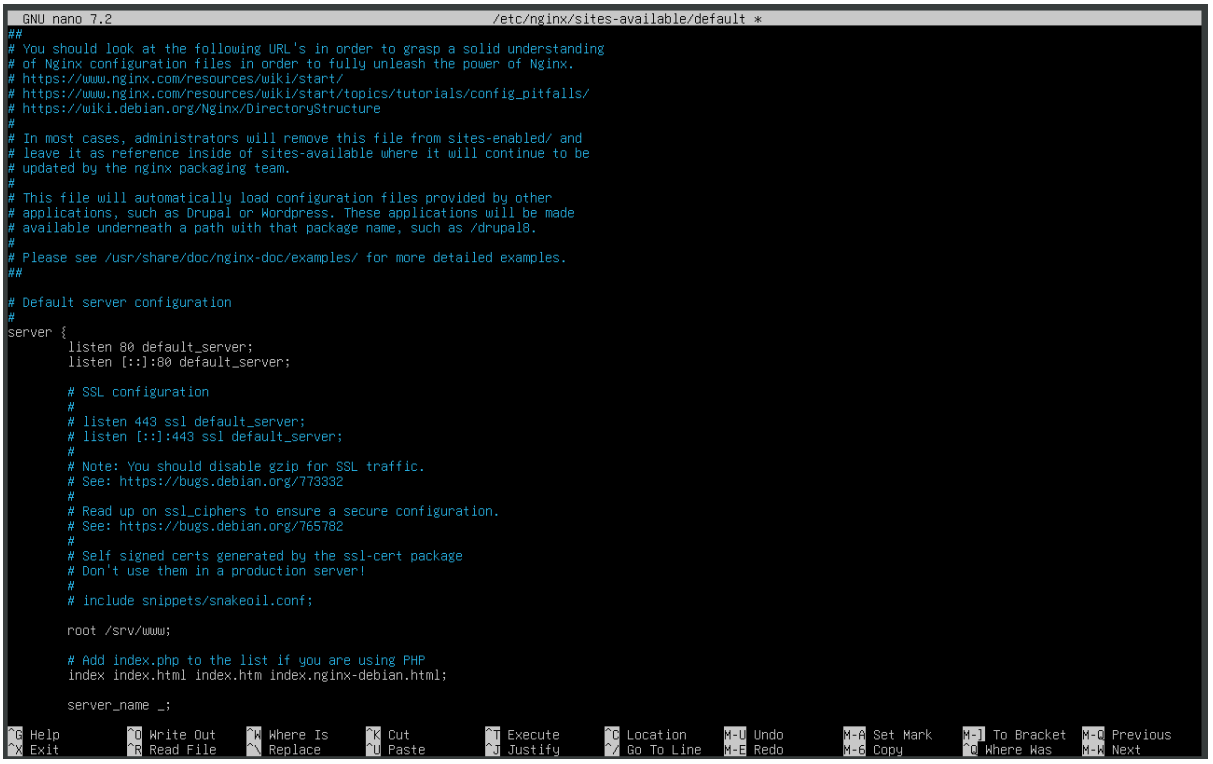
sudo apt install nginx -y

```
vboxuser@UbuntuServer:~$ sudo apt install nginx -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  nginx-common
Suggested packages:
  fcgiwrap nginx-doc ssl-cert
The following NEW packages will be installed:
  nginx nginx-common
0 upgraded, 2 newly installed, 0 to remove and 1 not upgraded.
Need to get 551 kB of archives.
After this operation, 1,596 kB of additional disk space will be used.
Get:1 http://es.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx-common all 1.24.0-2ubuntu7.3 [31.2 kB]
Get:2 http://es.archive.ubuntu.com/ubuntu noble-updates/main amd64 nginx amd64 1.24.0-2ubuntu7.3 [520 kB]
Fetched 551 kB in 0s (1,682 kB/s)
Preconfiguring packages ...
Selecting previously unselected package nginx-common.
(Reading database ... 86905 files and directories currently installed.)
Preparing to unpack .../nginx-common_1.24.0-2ubuntu7.3_all.deb ...
Unpacking nginx-common (1.24.0-2ubuntu7.3) ...
Selecting previously unselected package nginx.
Preparing to unpack .../nginx_1.24.0-2ubuntu7.3_amd64.deb ...
Unpacking nginx (1.24.0-2ubuntu7.3) ...
Setting up nginx (1.24.0-2ubuntu7.3) ...
Setting up nginx-common (1.24.0-2ubuntu7.3) ...
Created symlink /etc/systemd/system/multi-user.target.wants/nginx.service → /usr/lib/systemd/system/nginx.service.
Processing triggers for ufw (0.36.2-6) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
vboxuser@UbuntuServer:~$
```

Editamos el archivo de configuración de NGINX

sudo nano /etc/nginx/sites-available/default

y cambiamos la linea que dice “root /var/www/html;” borrando solo “html”



Verificamos que este todo correcto y reiniciamos NGINX con estos comandos:

sudo nginx -t

sudo systemctl restart nginx

```
vboxuser@UbuntuServer:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
vboxuser@UbuntuServer:~$ sudo systemctl restart nginx
vboxuser@UbuntuServer:~$
```

Ahora crearemos un archivo index.html con:

sudo tee /srv/www/index.html

```
vboxuser@UbuntuServer:~$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successf
vboxuser@UbuntuServer:~$ sudo systemctl restart nginx
vboxuser@UbuntuServer:~$ sudo tee /srv/www/index.html
```

El comando “tee” es para escribir en una ubicación protegida.

Para comprobar que funciona correctamente, vamos a un navegador web, en mi caso el de mi máquina local, y al escribir la dirección IP del servidor nos debería de aparecer la pagina de prueba.

Hola, soy una pagina de prueba :)