SSH_ACTIVITY

▼ Part 1: Basic SSH Configuration

1. Check SSH service status

Make sure that the SSH server is active:

```
sudo systemctl status ssh
```

```
root@server:/home/user2# sudo systemctl status ssh
● ssh.service - OpenBSD Secure Shell server
     Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2025-01-15 16:37:05 UTC; 16min ago
       Docs: man:sshd(8)
             man:sshd_config(5)
    Process: 605 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
   Main PID: 634 (sshd)
      Tasks: 1 (limit: 4540)
     Memory: 5.7M
        CPU: 439ms
     CGroup: /system.slice/ssh.service
             └─634 "sshd: /usr/sbin/sshd -D [listener] 0 of 10-100 startups"

Jan 15 16:42:19 server sshd[1046]: User user from 192.168.3.112 not allowed because none of user's grou>
Jan 15 16:42:19 server sshd[1046]: pam_unix(sshd:auth): authentication failure; logname= uid=0 euid=0 t>
Jan 15 16:42:21 server sshd[1046]: Failed password for invalid user user from 192.168.3.112 port 62091 >
Jan 15 16:42:22 server sshd[1046]: Failed password for invalid user user from 192.168.3.112 port 62091 >
Jan 15 16:42:31 server sshd[1046]: Failed password for invalid user user from 192.168.3.112 port 62091 >
Jan 15 16:42:32 server sshd[1046]: Received disconnect from 192.168.3.112 port 62091:11: Normal Shutdow>
Jan 15 16:42:32 server sshd[1046]: Disconnected from invalid user user 192.168.3.112 port 62091 [preaut>
Jan 15 16:42:32 server sshd[1046]: PAM 2 more authentication failures; logname= uid=0 euid=0 tty=ssh ru>
Jan 15 16:42:55 server sshd[1048]: Accepted password for user2 from 192.168.3.112 port 62092 ssh2
Jan 15 16:42:55 server sshd[1048]: pam_unix(sshd:session): session opened for user user2(uid=1001) by (>
lines 1-23/23 (END)
```

2. Configuring user access

- **Block root access**: Edit the configuration file:

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

- Add or modify the line:

  ```
  PermitRootLogin no
  ```

- Save the changes and restart the service:

  ```
  sudo systemctl restart ssh
  ```

```
  GNU nano 6.2                                    sshd_config
# Authentication:

#LoginGraceTime 2m
PermitRootLogin prohibit-password
```

```
user@server:~$ su root
Password:
su: Authentication failure
```

- Allow root access (optional):

  Change `PermitRootLogin` to `yes` if you want to enable it.

```
  GNU nano 6.2                                    sshd_config

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
```

```
user@server:~$ su root
Password:
root@server:/home/user#
```

- **Create users and group for SSH access:**

```
sudo adduser user2
sudo adduser user3
sudo groupadd ssh_users
```

- **Assign users to the group:**

```
sudo usermod -aG ssh_users user2
sudo usermod -aG ssh_users user3
```

```
sudo groupadd ssh_users
sudo usermod -aG ssh_users user2
sudo usermod -aG ssh_users user3
```

> The -aG parameter of the usermod command has the following functions:
>
> - a (append)
> - This parameter ensures that the user is added to the additional group(s) without removing the user from existing groups.
> - G (groups)
> - This parameter allows you to specify one or more additional groups to which you want to add the user.

- **Restrict SSH access to the `ssh_users` group only:**
  Edit the file:

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

  Add the line:

  ```
  AllowGroups ssh_users
  ```

  Restart the service:

  ```
  sudo systemctl restart ssh
  ```

```
  GNU nano 6.2                                    sshd_config
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes
AllowGroups ssh_users
```

```
usuario@user:~$ ssh user3@192.168.18.251
user3@192.168.18.251's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-130-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.


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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

3. Change the SSH listening port

Edit the SSH configuration:

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

Modify the line:

```
Port 22
```

by:

```
Port 10022
```

Restart the service:

```
sudo systemctl restart ssh
```

```
  GNU nano 6.2                                          sshd_config

# This is the sshd server system-wide configuration file.  Se
# sshd_config(5) for more information.

# This sshd was compiled with PATH=/usr/local/sbin:/usr/local

# The strategy used for options in the default sshd_config sh
# OpenSSH is to specify options with their default value wher
# possible, but leave them commented.  Uncommented options ov
# default value.

Include /etc/ssh/sshd_config.d/*.conf

Port 10022
```

```
usuario@user:~$ ssh -p 10022 user3@192.168.18.251
user3@192.168.18.251's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-130-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Mon Jan 13 15:33:06 2025 from 192.168.18.101
```

4. Hide last connection date and time

Edit the configuration file:

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

Add the line:

```
PrintLastLog no
```

Restart the service:

```
sudo systemctl restart ssh
```

```
  GNU nano 6.2                                    sshd_config *
# 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 without-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
X11Forwarding yes
#X11DisplayOffset 10
#X11UseLocalhost yes
#PermitTTY yes
PrintMotd no
PrintLastLog no
#TCPKeepAlive yes
```

```
user2@server: ~
File  Actions  Edit  View  Help
user2@server: ~
usuario@user:~$ ssh -p 10022 user2@192.168.18.251
user2@192.168.18.251's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-130-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

user2@server:~$
```

5. Enable X11 forwarding for remote graphic applications

- On the server, configure:

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

  Make sure these lines are present:

```
X11Forwarding yes
AllowTcpForwarding yes
```

Restart the service:

```
sudo systemctl restart ssh
```

```
#Match User anoncvs
        X11Forwarding yes
        AllowTcpForwarding yes
#       PermitTTY no
```

- On the client, test the forwarding:

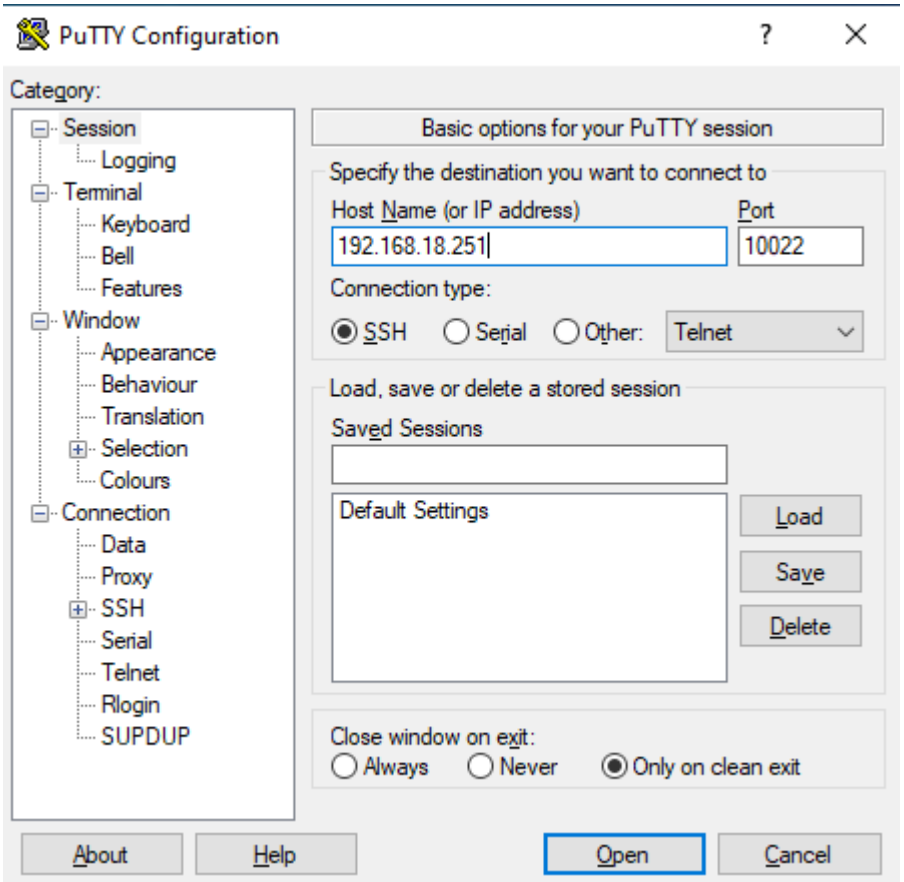
```
ssh -X user2@<ip_servidor>
firefox &
```

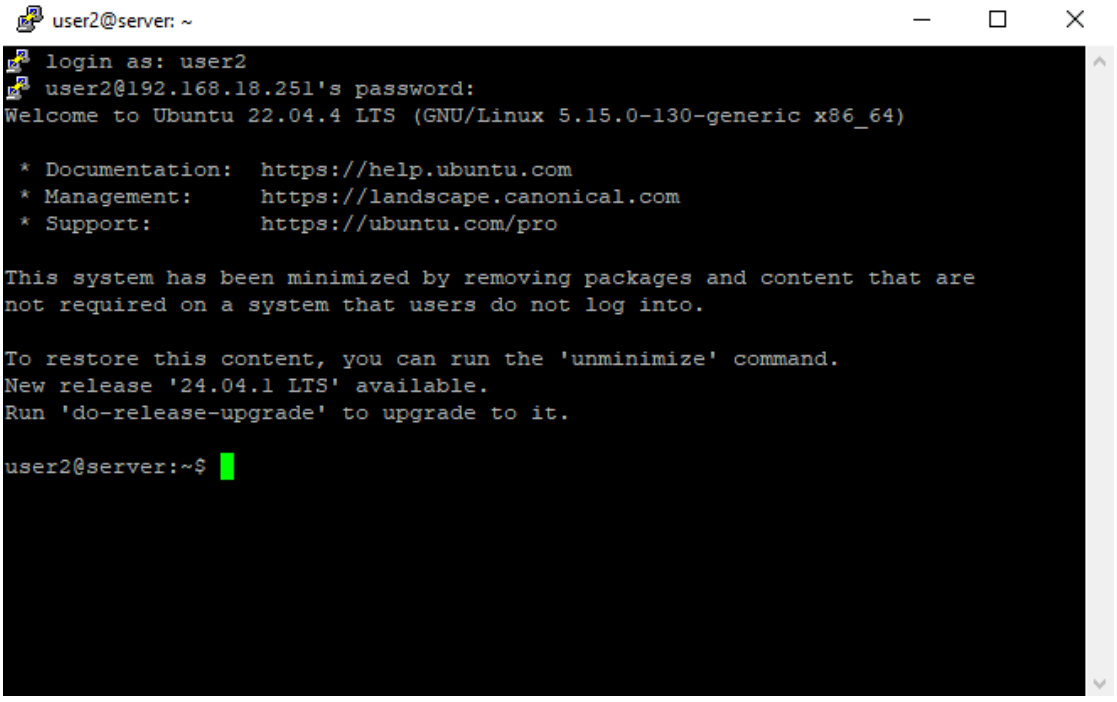
```
usuario@user:~$ ssh -x -p 10022 user2@192.168.18.251
user2@192.168.18.251's password:
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-130-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

user2@server:~$ firefox &
[1] 1331
user2@server:~$ -bash: firefox: command not found
```

> `Command not found` may appear if the server does not have a graphical interface.

6. Test connection from a Windows client

Use an SSH client such as **PuTTY** to connect to the GNU/Linux server and verify the connection.





▼ Part 2: Advanced Authentication

1. Password-based authentication

Make sure that on the server, the configuration file allows password authentication:

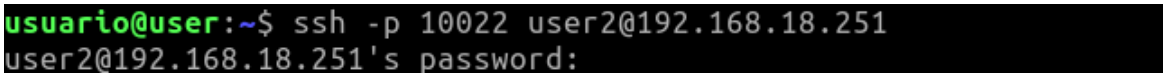
```
sudo nano /etc/ssh/sshd_config
```

Confirms that the next line is enabled:

```
PasswordAuthentication yes
```

Restart the service:

```
sudo systemctl restart ssh
```



2. Set up public key authentication (without passphrase).

- **Generate a key pair on the client machine:**

  ```
  ssh-keygen -t rsa -b 2048
  ```

```
usuario@user:~$ ssh-keygen -t rsa -b 2048
Generating public/private rsa key pair.
Enter file in which to save the key (/home/usuario/.ssh/id_rsa):
/home/usuario/.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 /home/usuario/.ssh/id_rsa
Your public key has been saved in /home/usuario/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:5imlaiad8r/Qs6t/nbDEuJWDVD+OP10ynAggrpGpQA4 usuario@user
The key's randomart image is:
+---[RSA 2048]----+
|                 |
|E.  . . .        |
|+  + . o .       |
|..+ . . . o      |
|.. o . +S= + .   |
|. .  .o=O.o = .  |
|   ...=+o* o +   |
|  o =o.+o = .    |
|   *++*+   .     |
+----[SHA256]-----+
```

- **Copy the public key to the server:**

```
ssh-copy-id -i ~/.ssh/id_rsa.pub usuario@<ip_servidor
>
```

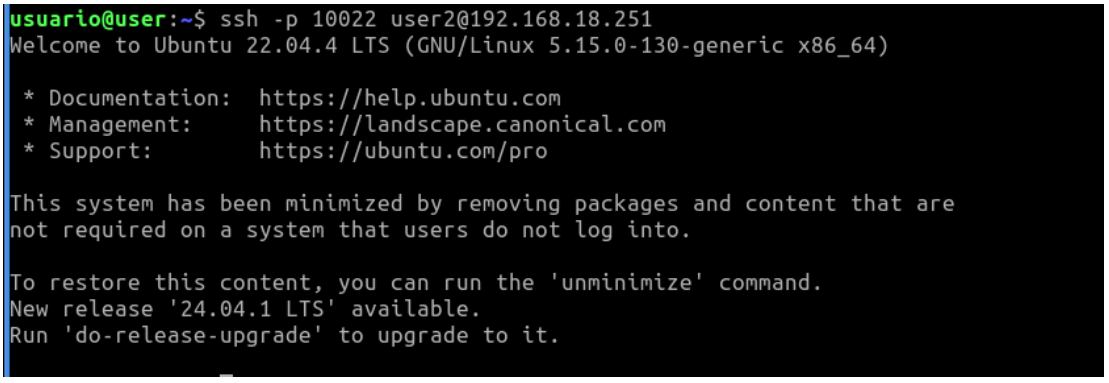
```
usuario@user:~$ ssh-copy-id -p 10022 -i ~/.ssh/id_rsa.pub user2@192.168.18.251
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed: "/home/usuario/.ssh/id_rsa.pub"
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already inst
alled
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the n
ew keys
user2@192.168.18.251's password:

Number of key(s) added: 1

Now try logging into the machine, with:   "ssh -p '10022' 'user2@192.168.18.251'"
and check to make sure that only the key(s) you wanted were added.
```

- **Uncomment** `AuthorizedKeysFile  .ssh/authorized_keys`

```
  GNU nano 6.2                                  sshd_config *
#MaxSessions 10

PubkeyAuthentication yes
AllowGroups ssh_users

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

- **Verify connection without password:**

```
ssh usuario@<ip_servidor>
```



▼ Parte 3: Transferencia de Archivos entre Cliente y Servidor

1. Prepare the environment on the client machine

- **Create a folder named `dir1` in the `HOME` directory:**

```
mkdir ~/dir1
```

- **Inside `dir1`, create another folder named `dir2`:**

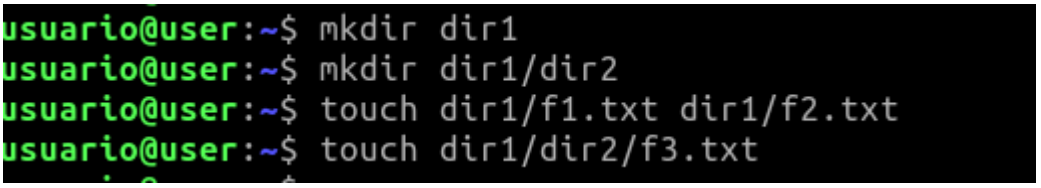
```
mkdir ~/dir1/dir2
```

- **Create two text files in `dir1` named `f1.txt` and `f2.txt`:**

```
touch ~/dir1/f1.txt ~/dir1/f2.txt
```

- **In `dir2`, create a file named `f3.txt`:**

```
touch ~/dir1/dir2/f3.txt
```



2. Transfer individual files to the server

- **Copy the files `f1.txt` and `f2.txt` from the client to the server:**

```
scp ~/dir1/f1.txt ~/dir1/f2.txt usuario@<ip_servidor
>:~/
```

```
usuario@user:~$ scp /home/usuario/dir1/f1.txt /home/usuario/dir1/f2.txt user2@192.168.18.251:~/
f1.txt                                                    100%    0     0.0KB/s   00:00
f2.txt                                                    100%    0     0.0KB/s   00:00
usuario@user:~$
```

3. Transfer an entire folder to the server

- **Copy the `dir2` folder (and its contents) to the server recursively:**

```
scp -r ~/dir1/dir2 usuario@<ip_servidor>:~/
```

```
usuario@user:~$ scp -r /home/usuario/dir1/dir2/ user2@192.168.18.251:~/
f3.txt                                                    100%    0     0.0KB/s   00:00
```

4. Modify a file on the server and send it back to the client

On the server:

- **Edit the file `f1.txt`:**
  Connect to the server and add content to the file:

```
ssh usuario@<ip_servidor> "echo 'Contenido modificad
o' >> ~/f1.txt"
```

From the client:

- **Copy the modified file `f1.txt` from the server back to the client:**

```
scp usuario@<ip_servidor>:~/f1.txt ~/dir1/f1.txt
```

```
root@server:/home/user2# scp /home/user2/f1.txt usuario@192.168.18.101:~/dir1/f1.txt
usuario@192.168.18.101's password:
f1.txt                                                    100%   36     7.3KB/s   00:00
```

▼ PART 4: TCP Port Forwarding with SSH

1. Configure TCP port forwarding using SSH

On the client machine, run the following command to set up an SSH tunnel that forwards port **53** on the server to port **53** on the client:

```
ssh -L 53:<ip_servidor>:53 usuario@<ip_servidor>
```

Explanation of the `L`: parameter.

- `L`: Configures a **local port forwarding**. This means that requests to the port specified on the client machine (in this case, 53) will be forwarded to the corresponding port on the server.

```
usuario@user:~$ ssh -L 53:192.168.18.251:53 user2@192.168.18.251
bind [127.0.0.1]:53: Permission denied
channel_setup_fwd_listener_tcpip: cannot listen to port: 53
Could not request local forwarding.
Welcome to Ubuntu 22.04.4 LTS (GNU/Linux 5.15.0-130-generic x86_64)

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

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

To restore this content, you can run the 'unminimize' command.
New release '24.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.
```

---

2. Perform a DNS query through the forwarded port.

Use the `dig` command to perform a DNS query using the forwarded port:

```
dig -p 53 @localhost google.com +tcp
```

Parameters of Interest:

- `p 53`:Specifies port **53** for the query.

- `@localhost`: Indicates that the query is made through the SSH tunnel.
- `+tcp`: Forces the use of TCP for the query.

```
user2@server:~$ dig -p 53 @localhost google.com +tcp

; <<>> DiG 9.18.28-0ubuntu0.22.04.1-Ubuntu <<>> -p 53 @localhost google.com +tcp
; (1 server found)
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 55334
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 0, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 1232
; COOKIE: 0eac15bfc8ce538f01000000678e70940a5a8cbb5b288166 (good)
;; QUESTION SECTION:
;google.com.                    IN      A

;; ANSWER SECTION:
google.com.             247     IN      A       142.250.184.14

;; Query time: 100 msec
;; SERVER: 127.0.0.1#53(localhost) (TCP)
;; WHEN: Mon Jan 20 15:49:40 UTC 2025
;; MSG SIZE  rcvd: 83
```

---

3. Performing a query with the `host` command using TCP

To perform another DNS query, use the `host` command as follows:

```
host -T google.com localhost
```

Explanation of the `T` parameter:

- `T:` Forces the `host` command to perform the query using the TCP protocol.

```
user2@server:~$ host -T google.com localhost
Using domain server:
Name: localhost
Address: 127.0.0.1#53
Aliases:

google.com has address 142.250.184.14
google.com has IPv6 address 2a00:1450:4003:807::200e
google.com mail is handled by 10 smtp.google.com.
```

▼ PART 5: SSH Tunnel Configuration

Preparations on the server

1. **Enable tunneling on the SSH server:**

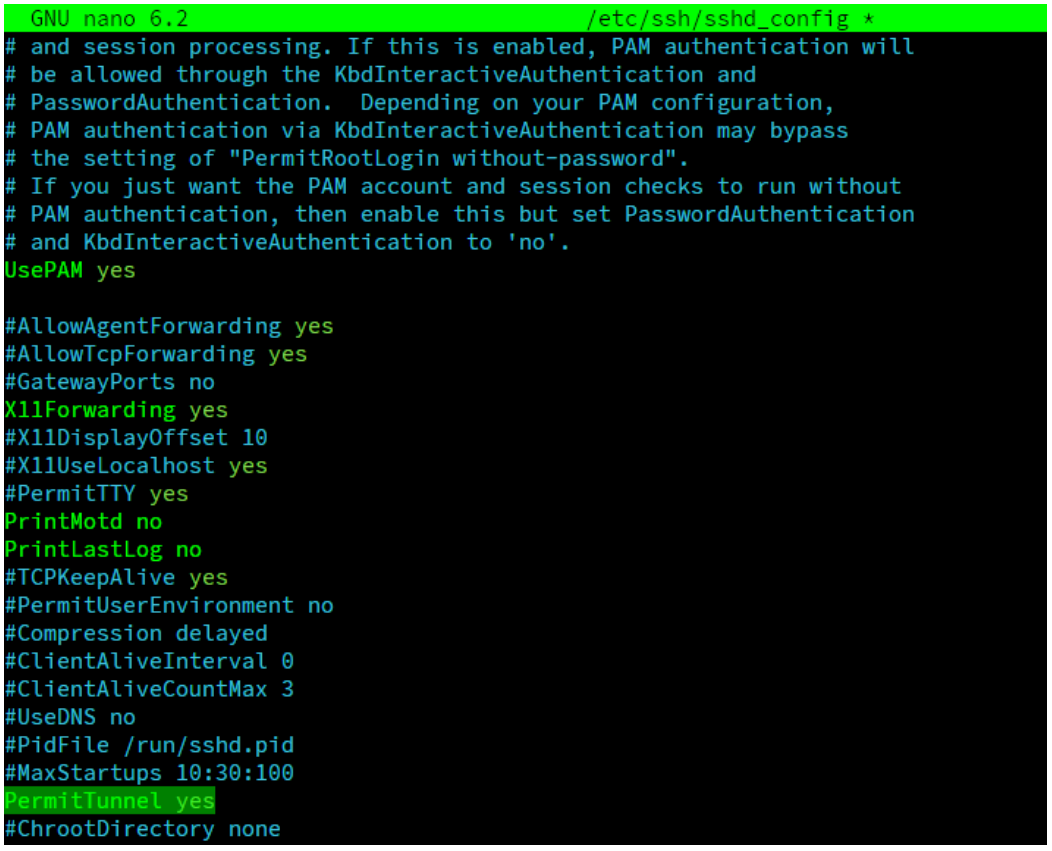
   - Open the SSH server configuration file:

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

   - Make sure the following lines are configured:

     ```
     PermitTunnel yes
     PermitRootLogin yes
     ```

   - Save and close the file.



2. **Restart the SSH service:**

   ```
   sudo systemctl restart ssh
   ```

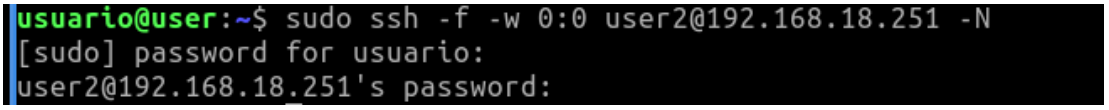
1. Configure the SSH tunnel

- **From the client:**
  - Set up the SSH tunnel:

    ```
    ssh -f -w 0:0 usuario@<ip_servidor> -N
    ```

    ### Explained parameters:

    - `f`: Run SSH in the background after authentication.
    - `w 0:0`: Creates a tunnel interface using the `tun0` device on both ends.
    - `N`: Does not execute remote commands, only establishes the tunnel.



2. Configure Network Interfaces

> WE COULD NOT COMPLETE THIS PART, BUT I WILL EXPLAIN IT.

On the Client:

- Assign an IP address and activate the tunnel interface:

  ```
  sudo ip addr add 10.1.1.2/30 dev tun0
  sudo ip link set tun0 up
  ```

On the Server:

- Assign an IP address and activate the tunnel interface:

  ```
  sudo ip addr add 10.1.1.1/30 dev tun0
  sudo ip link set tun0 up
  ```

3. Verify Tunnel Connectivity

- **From the Client:**

```
ping 10.1.1.1
```

- **From the Server:**

```
ping 10.1.1.2
```

4. Access the Web Server Through the Tunnel

- Open a browser on the client machine and navigate to the address: