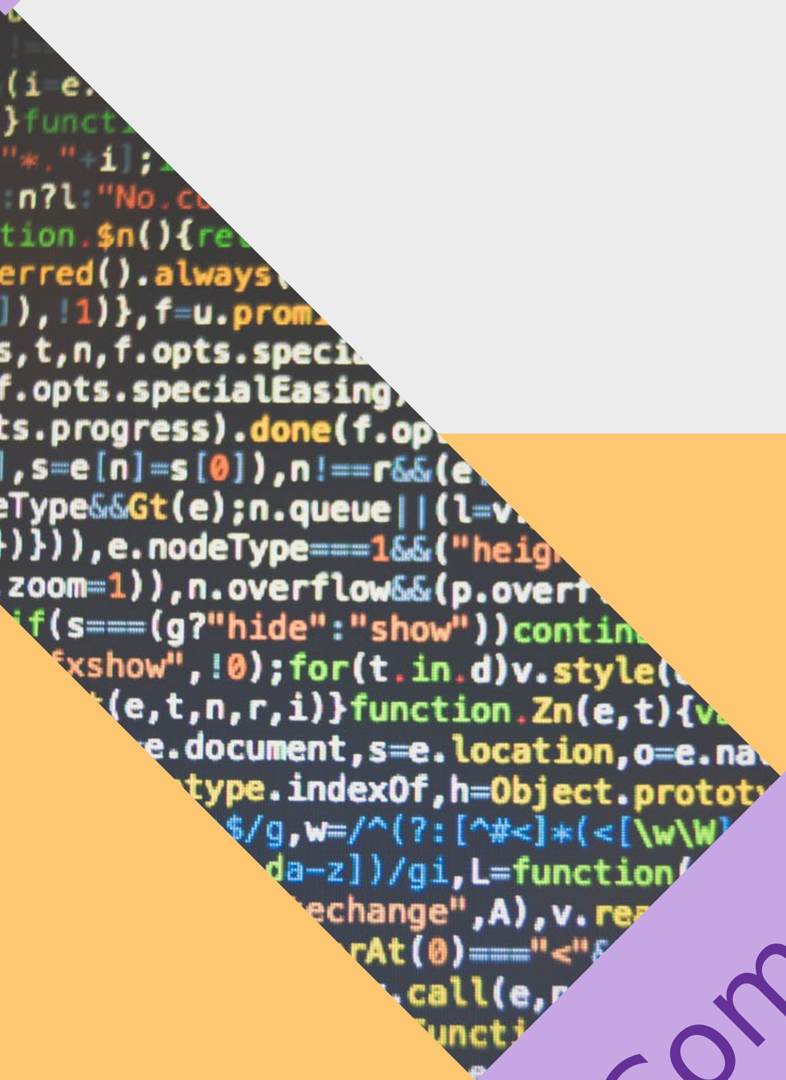
Curs 2019-2020



Computer Systems
Linux Server II

IES Jaume II El Just
Tavernes de la Valldigna

Contents

1 SSH. Manage Server Ubuntu 18.04 from different computer

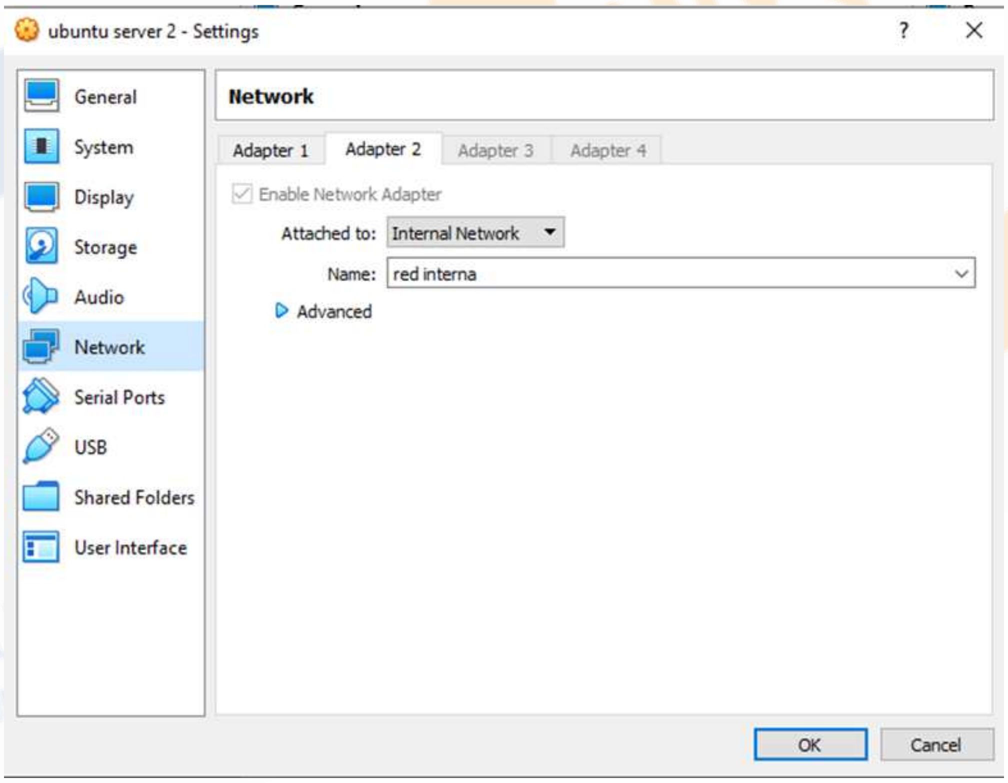
SSH protocol provides a simple and reasonable connection in order to manage the computer on line, through the network.

The main benefit is that all the information exchanged between both computers is encrypted. This feature is very important when the main aim is server remote control through the network. So in order to prepare the server, first of all it would be necessary to configure the network adapter

1.1 Preparing the server

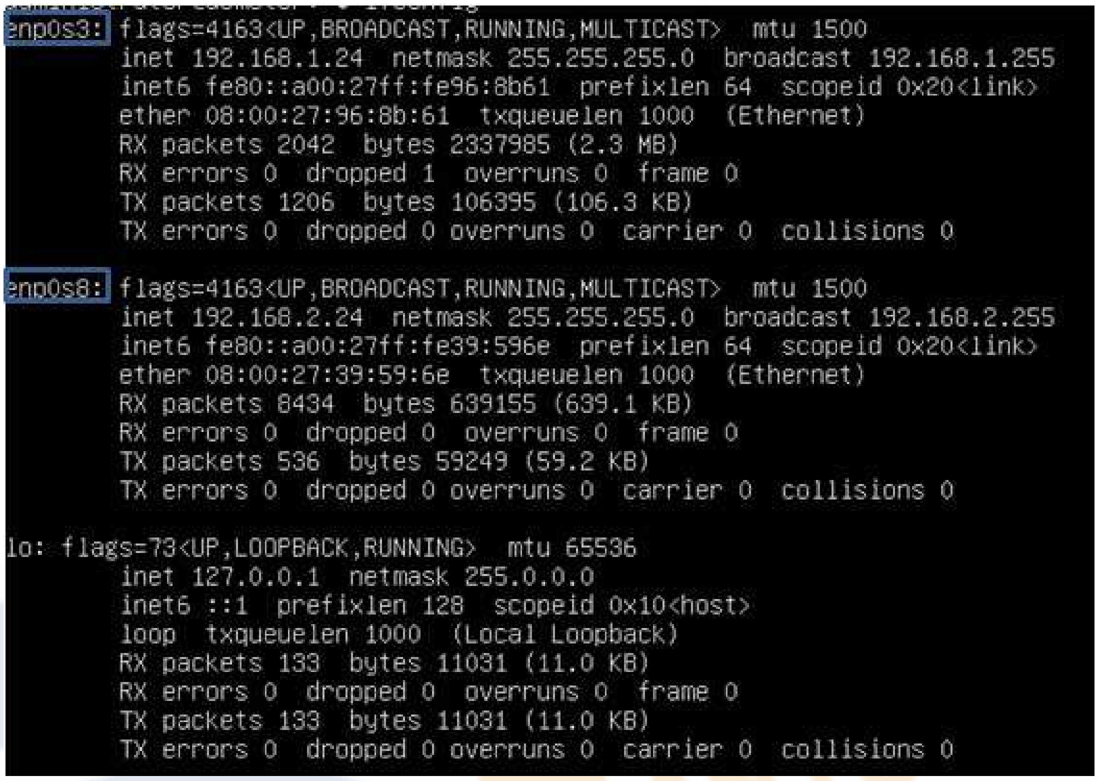
It is interesting to have 2 connections at the server, one for the Internet access and the other for communicating with the clients. So, it would be necessary for a suitable configuration of the network's cards:

- First of all, after turning off the server, another network adapter will be added to the virtual machine. This time, the adapter will be configured as an internal Network.



Enable Network Adapter ✓
Attached to: Internal Network
Name: red interna

- After turning on the machine, it would be necessary to configure the adapter. In order to know what name the adapter has, ifconfig command will be executed:

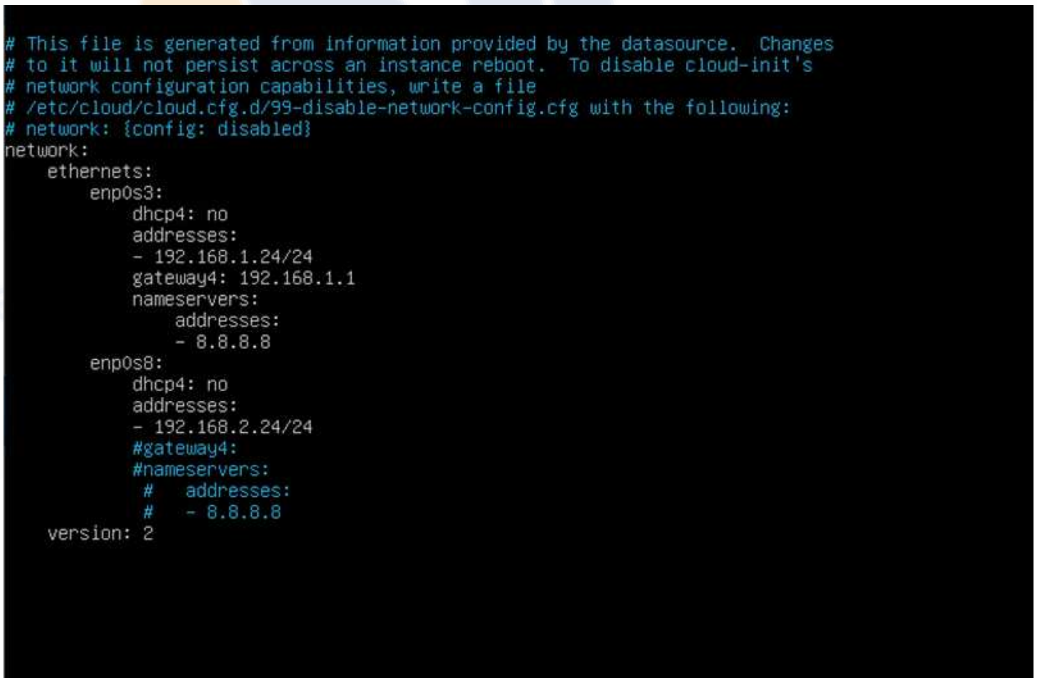


```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.1.24  netmask 255.255.255.0  broadcast 192.168.1.255
        inet6 fe80::a00:27ff:fe96:8b61  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:96:8b:61  txqueuelen 1000  (Ethernet)
        RX packets 2042  bytes 2337985 (2.3 MB)
        RX errors 0  dropped 1  overruns 0  frame 0
        TX packets 1206  bytes 106395 (106.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.24  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::a00:27ff:fe39:596e  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:39:59:6e  txqueuelen 1000  (Ethernet)
        RX packets 8434  bytes 639155 (639.1 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 536  bytes 59249 (59.2 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 133  bytes 11031 (11.0 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 133  bytes 11031 (11.0 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

It could be that the adapter would be disabled. In that case, it would be necessary to run:

```
sudo ifconfig name_adapter up
```

- The following step would be configuring the adapter with a suitable IP, to do this, the file /etc/netplan/50-cloud-init.yaml will be modified:



```
# This file is generated from information provided by the datasource.  Changes
# to it will not persist across an instance reboot.  To disable cloud-init's
# network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    ethernets:
        enp0s3:
            dhcp4: no
            addresses:
            - 192.168.1.24/24
            gateway4: 192.168.1.1
            nameservers:
                addresses:
                - 8.8.8.8
        enp0s8:
            dhcp4: no
            addresses:
            - 192.168.2.24/24
            #gateway4:
            #nameservers:
             #   addresses:
             #   - 8.8.8.8
    version: 2
```

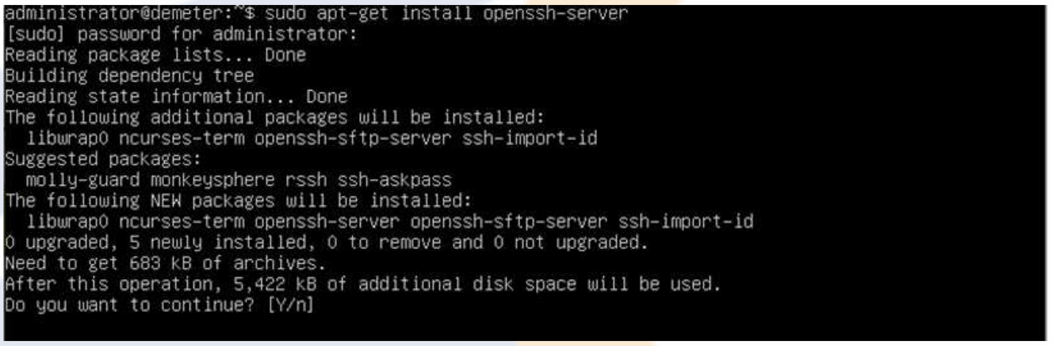
In that case, pay attention to the configuration for the enp0s8 adapter. This is the internal network adapter and it has been configured just with an IP address, In this case the network has been chosen, 192.168.2.0 with the IP 192.168.2.24. The reason is that the internal network should be in a different network.

- Remember updating the configuration by executing:

```
sudo netplan apply
        or
sudo systemctl status systemd-networkd
```

- In order to install OpenSSH, the following command will be executed:

```
sudo apt-get install openssh-server
```

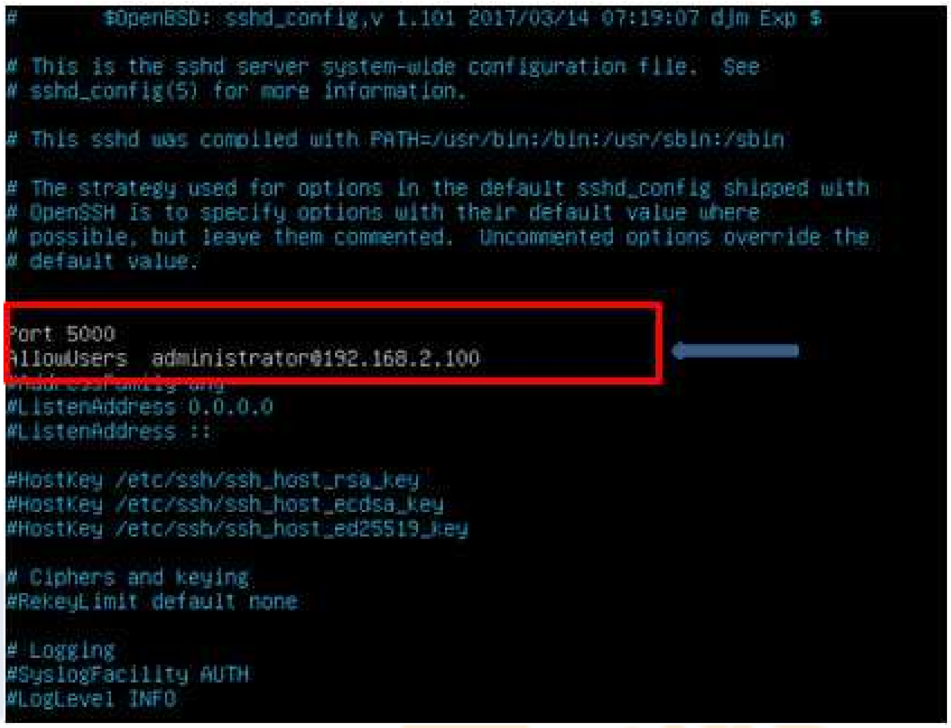


```
administrator@demeter:~$ sudo apt-get install openssh-server
[sudo] password for administrator:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  libwrap0 ncurses-term openssh-sftp-server ssh-import-id
Suggested packages:
  molly-guard monkeysphere rssh ssh-askpass
The following NEW packages will be installed:
  libwrap0 ncurses-term openssh-server openssh-sftp-server ssh-import-id
0 upgraded, 5 newly installed, 0 to remove and 0 not upgraded.
Need to get 683 kB of archives.
After this operation, 5,422 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

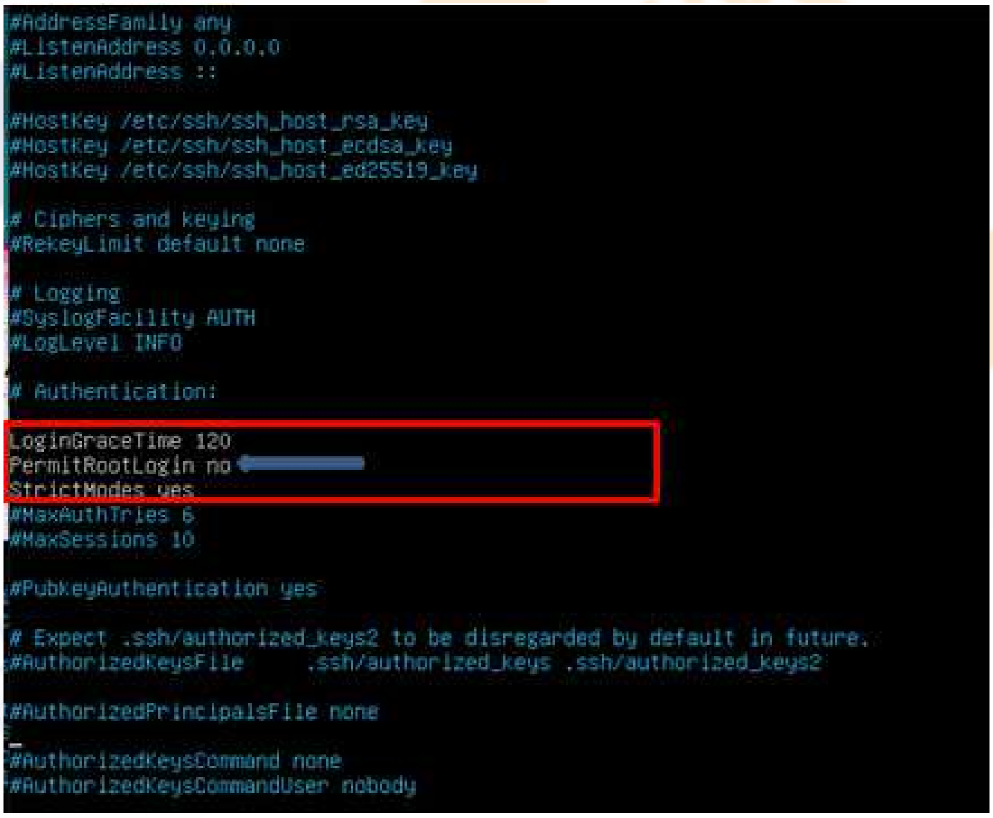
- Once the package is installed, it is necessary to make some adjusts in the configuration file in order to get more safety connection:

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

- Nano editor will open the file. The proposed changes are:
  - Port parameter: that parameter is the port which is going to be used for the connection. Normally 22 is used but in order to don't allow improper accesses, it is recommended to use some port between 1025 and 65536, for instance 5000.
  - The Allowusers parameter: this parameter restricts the remote connection just for the specified users. It is possible to add IP address of the computer from which will establish the connection.
  - PermitRootLogin parameter: it will be changed so as not to allow connection with root user. The following picture show the changes:



```
#       $OpenBSD: sshd_config,v 1.101 2017/03/14 07:19:07 djm Exp $

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

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

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

Port 5000
AllowUsers  administrator@192.168.2.100
#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
```



```
#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 120
PermitRootLogin no
StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

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

#AuthorizedPrincipalsFile none

#AuthorizedKeysCommand none
#AuthorizedKeysCommandUser nobody
```

Remember to save the file with ctrl+x.

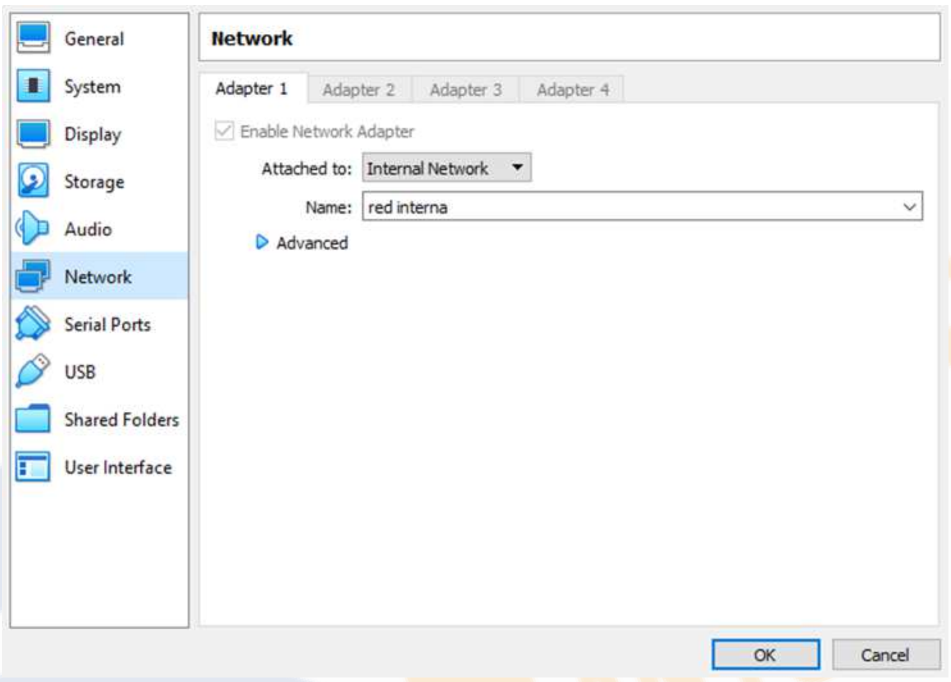
- Finally, it will be necessary to restart the service:

```
sudo service ssh restart
```

1.2 Managing the server from the other computer

The following point has been made with a virtual machine with Xubuntu.

1. First of all, the network adapter would be established with an internal network in order to connect with the server's internal network.



Network

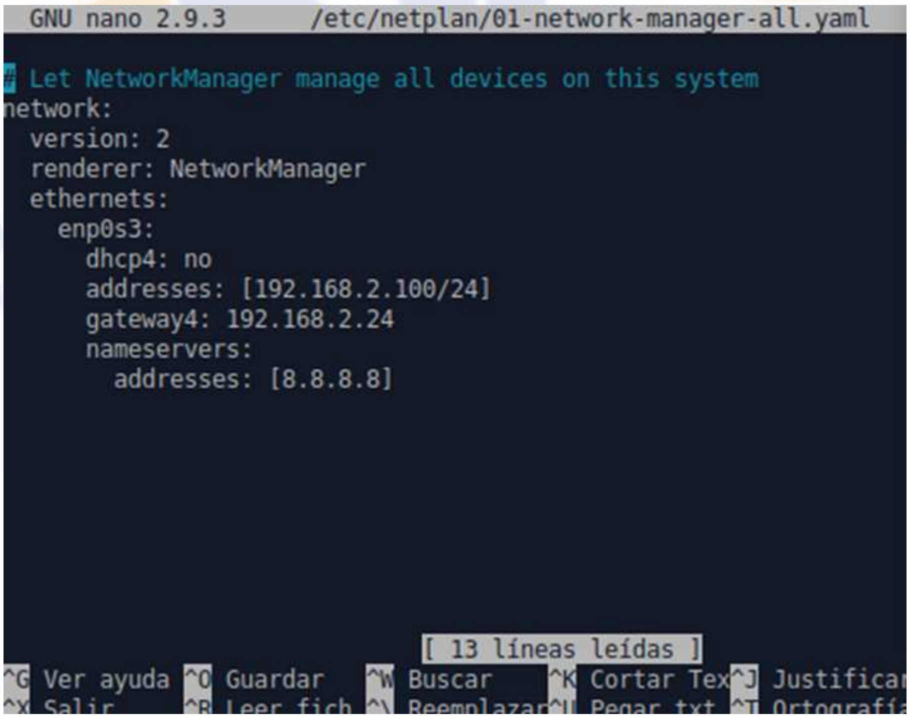
Enable Network Adapter

Attached to: Internal Network

Name: red interna

2. Once the adapter has been configured, the following step will also be in the network configuration with the netplan tool. Bear in mind that the IP of this computer should be in the same network of the Server Internal Network adapter, besides in the sshd_config file, the allowed connection is from the 192.168.2.100 address. So, it will be executed at terminal of this virtual machine:

```
sudo nano /etc/netplan/01-network-manager-all.yaml
```



```
  GNU nano 2.9.3        /etc/netplan/01-network-manager-all.yaml

# Let NetworkManager manage all devices on this system
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp0s3:
      dhcp4: no
      addresses: [192.168.2.100/24]
      gateway4: 192.168.2.24
      nameservers:
        addresses: [8.8.8.8]



                              [ 13 líneas leídas ]
^G Ver ayuda   ^O Guardar     ^W Buscar      ^K Cortar Tex^J Justificar
^X Salir       ^R Leer fich   ^\ Reemplazar^U Pegar txt  ^T Ortografía
```

Pay attention because in that Ubuntu flavour, the file is 01-network-manager-all.yaml instead of 50-cloud-init.yaml. There are some more little changes in the configuration file.

3. Remember updating the configuration by executing:

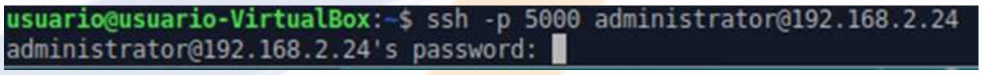
```
sudo netplan apply
        or
sudo systemctl status systemd-networkd
```

4. After that, in order to establish the connection, run:

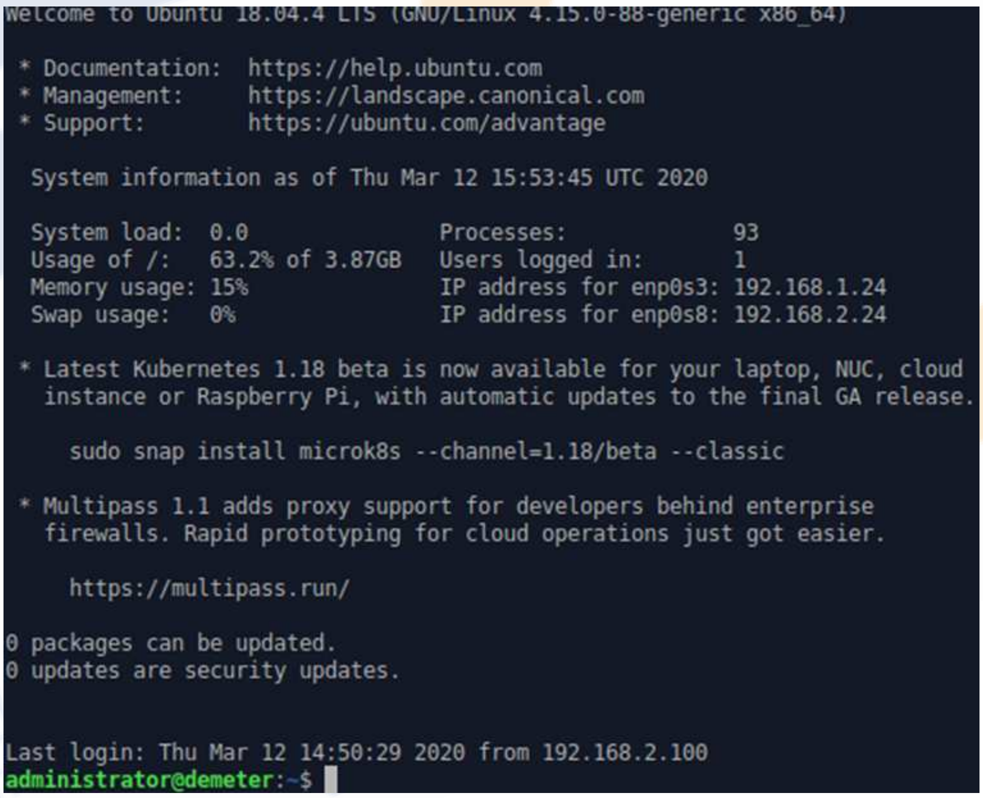
```
ssh -p port\_number  user\_name@IP address
```

In this case:

```
ssh -p 5000 administrator@192.168.2.24
```



```
usuario@usuario-VirtualBox:~$ ssh -p 5000 administrator@192.168.2.24
administrator@192.168.2.24's password:
```



```
Welcome to Ubuntu 18.04.4 LTS (GNU/Linux 4.15.0-88-generic x86_64)

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

  System information as of Thu Mar 12 15:53:45 UTC 2020

  System load:  0.0               Processes:             93
  Usage of /:   63.2% of 3.87GB   Users logged in:       1
  Memory usage: 15%               IP address for enp0s3: 192.168.1.24
  Swap usage:   0%                IP address for enp0s8: 192.168.2.24

 * Latest Kubernetes 1.18 beta is now available for your laptop, NUC, cloud
   instance or Raspberry Pi, with automatic updates to the final GA release.

     sudo snap install microk8s --channel=1.18/beta --classic

 * Multipass 1.1 adds proxy support for developers behind enterprise
   firewalls. Rapid prototyping for cloud operations just got easier.

     https://multipass.run/

0 packages can be updated.
0 updates are security updates.


Last login: Thu Mar 12 14:50:29 2020 from 192.168.2.100
administrator@demeter:~$
```

Once the password of the administrator user is introduced, it will appear in the welcome message from the server. It means that the access to the server has been corrected. Now it is possible to manage the server from the computer with xubuntu.

2 NFS Server

NFS (Network File System) is a protocol which is used in the local access network in order to create a distributed system of files. It was invented by Sun Microsystems. The main goal of that protocol is that several users which belong to the same network are able to access to shared files and directories as they do in their local computer. This way, the storage files and directories are centralized in the same point. So, the installation will have 2 side:

- The computer which is the server and it is where files and folders are stored and shared
- One or several computers which are clients.

2.1 Installing NFS in the Server

1. In order to use NFS, first of all, if it has not been done, the system should be updated:

   ```
   sudo apt-get update
   ```

   After updating the system, the package nfs-kernel-server should be installed:

   ```
   sudo apt-get nfs-kernel-server
   ```

2. Once the package is installed, the folder which is required to share with clients should be created. This folder is called export directory, and it will be created in /mnt directory. Remember that this directory is destinated to mount temporary drives:

   ```
   sudo mkdir -p /mnt/sharedfolder
   ```

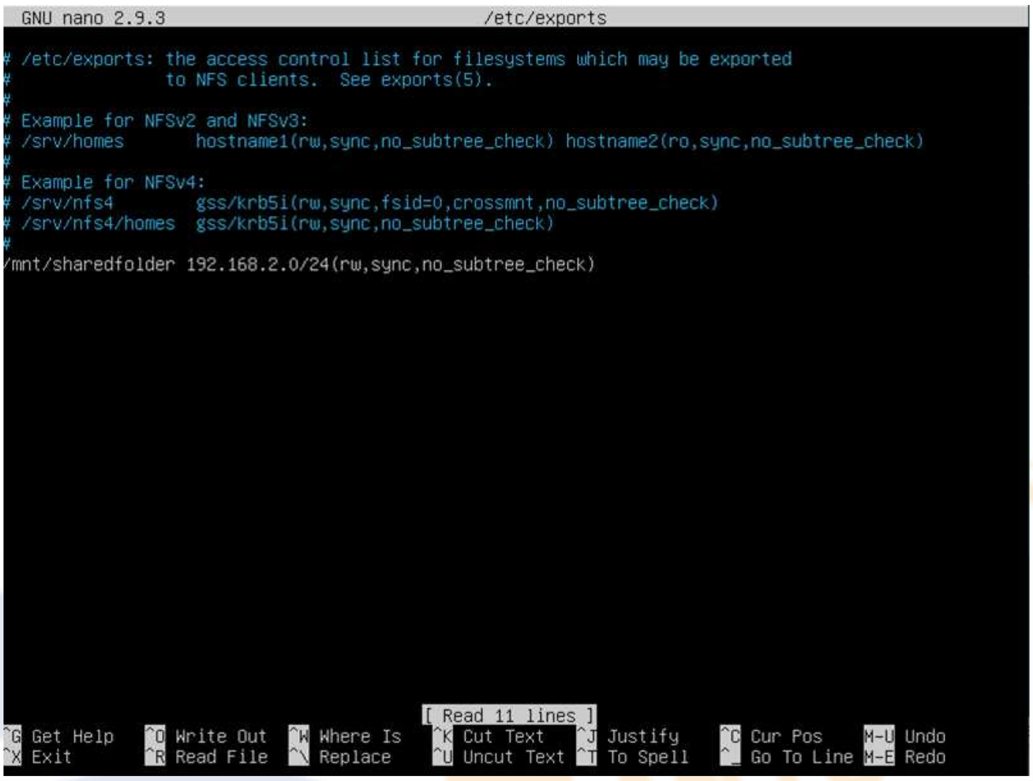
3. It is desired that all clients are to access the directory, restrictive permission should be removed of the export folder:

   ```
   sudo chown nobody:nogroup /mnt/sharedfolder
   sudo chmod 777 /mnt/sharedfolder
   ```

4. The follwoing step is to assign server access to clients through NFS export file. This file is located in /etc directory.

   ```
   sudo nano /etc/exports
   ```

5. There are differents ways to assign server access. It is possible to give individual access, multiple clients access or multiple clients access specifying the entire subnet. So, in the file, the next line should be added in order to specify access for all the network clients:



```
GNU nano 2.9.3                          /etc/exports

# /etc/exports: the access control list for filesystems which may be exported
#               to NFS clients.  See exports(5).
#
# Example for NFSv2 and NFSv3:
# /srv/homes       hostname1(rw,sync,no_subtree_check) hostname2(ro,sync,no_subtree_check)
#
# Example for NFSv4:
# /srv/nfs4        gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
# /srv/nfs4/homes  gss/krb5i(rw,sync,no_subtree_check)
#
/mnt/sharedfolder 192.168.2.0/24(rw,sync,no_subtree_check)









                              [ Read 11 lines ]
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text   ^J Justify   ^C Cur Pos    M-U Undo
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line M-E Redo
```

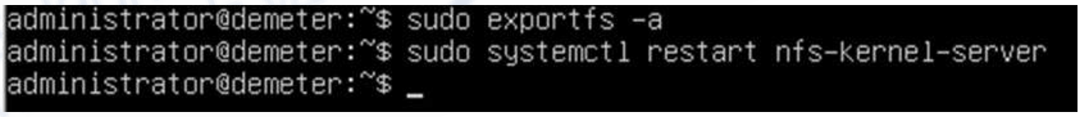
In that case, the internal ID network is 192.168.2.0/24. The parameters mean:

- rw: read and write operations
- sync: write any change to the disc before applying it
- no_subtree_check: prevent subtree checking. The permissions of the folder above are not checked. This means less security but faster access.

6. Now, the shared directory is exported through:

```
sudo exportfs -a
```

7. And the NFS Kerner server will be restarted:



```
administrator@demeter:~$ sudo exportfs -a
administrator@demeter:~$ sudo systemctl restart nfs-kernel-server
administrator@demeter:~$ _
```

2.1.1 Configuring Iptables at server

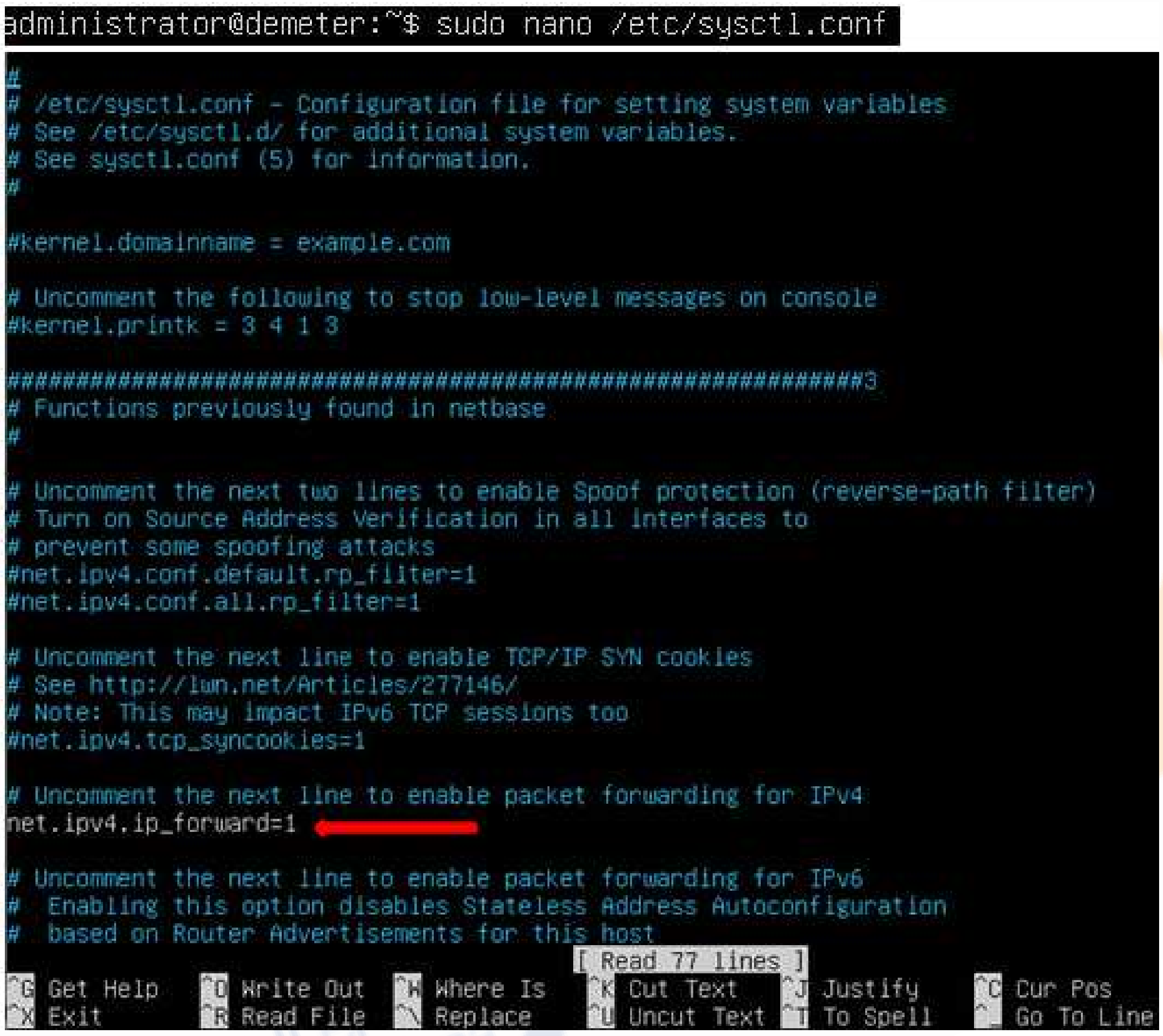
First of all, client does not have Internet access, because it is connected to the internal network to the server. In order to let Internet connection to client, some configurations should be carried out at the server.

To provide Internet access to the clients of the Server, the connection should be redirected to the adapter which has Internet connection.

1. To do this, it is necessary to check the value of the ip_forward variable. This variable is in charge of activating the data onward between two network cards. In order to check it:

```
cat /proc/sys/net/ipv4/ip_forward
```

1. If the returned value is “1”, it is not necessary to do anything, but if the returned value is 0, it will be necessary to change it. So, the variable will be searched at file /etc/sysctl.conf. The searched name is net.ipv4.ip_forward = 1:



```
administrator@demeter:~$ sudo nano /etc/sysctl.conf
#
# /etc/sysctl.conf - Configuration file for setting system variables
# See /etc/sysctl.d/ for additional system variables.
# See sysctl.conf (5) for information.
#

#kernel.domainname = example.com

# Uncomment the following to stop low-level messages on console
#kernel.printk = 3 4 1 3

##############################################################3
# Functions previously found in netbase
#

# Uncomment the next two lines to enable Spoof protection (reverse-path filter)
# Turn on Source Address Verification in all interfaces to
# prevent some spoofing attacks
#net.ipv4.conf.default.rp_filter=1
#net.ipv4.conf.all.rp_filter=1

# Uncomment the next line to enable TCP/IP SYN cookies
# See http://lwn.net/Articles/277146/
# Note: This may impact IPv6 TCP sessions too
#net.ipv4.tcp_syncookies=1

# Uncomment the next line to enable packet forwarding for IPv4
net.ipv4.ip_forward=1

# Uncomment the next line to enable packet forwarding for IPv6
#  Enabling this option disables Stateless Address Autoconfiguration
#  based on Router Advertisements for this host
                                   [ Read 77 lines ]
^G Get Help    ^O Write Out   ^W Where Is    ^K Cut Text    ^J Justify     ^C Cur Pos
^X Exit        ^R Read File   ^\ Replace     ^U Uncut Text  ^T To Spell    ^_ Go To Line
```

It could be that the variable already exists, but with the # character before. So, it is necessary to erase the # character in order to actívate it.

2. In order to allow the traffic between two cards, and thereby allowing traffic between Internet and client, it is necessary to establish the rule:

```
iptables -A POSTROUTING -t nat -s IP -o adaptername -j MASQUERADE
```

So, in this example:

```
iptables -A POSTROUTING -t nat -s 192.168.2.24/24 -o enp0s3 -j MASQUERADE
```

- 192.168.2.24 is the address of the internal network card of the server

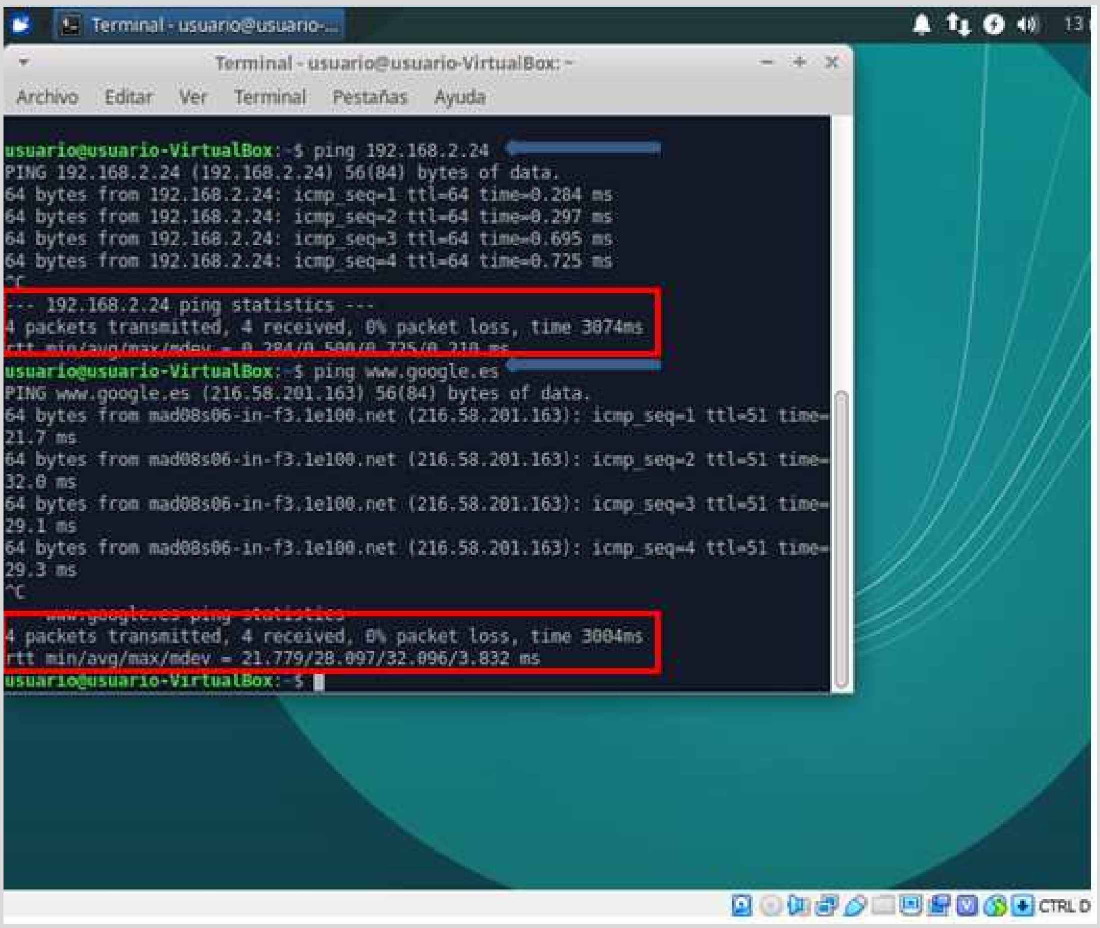
- enp0s3 is the name of the adapter which is connected on Internet throught a bridge connection.

This way, the traffic which arrives from the Internal network in order to surf on the Internet will be redirected to adapter enp0s3.

Check the conection

Now, to check the conection, it is possible to turn on the client machine and try with the command ping. Ping check the connection between the network card and the destionation which, it will be specified.

At the client computer, the server connection will be checked, and the connection with some web for instance www.google.com


Terminal - usuario@usuario-...
Terminal - usuario@usuario-VirtualBox: ~
Archivo
Editar
Ver
Terminal
Pestañas
Ayuda

```
usuario@usuario-VirtualBox:~$ ping 192.168.2.24
PING 192.168.2.24 (192.168.2.24) 56(84) bytes of data.
64 bytes from 192.168.2.24: icmp_seq=1 ttl=64 time=0.284 ms
64 bytes from 192.168.2.24: icmp_seq=2 ttl=64 time=0.297 ms
64 bytes from 192.168.2.24: icmp_seq=3 ttl=64 time=0.695 ms
64 bytes from 192.168.2.24: icmp_seq=4 ttl=64 time=0.725 ms
^C
--- 192.168.2.24 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3074ms
rtt min/avg/max/mdev = 0.284/0.500/0.725/0.210 ms
usuario@usuario-VirtualBox:~$ ping www.google.es
PING www.google.es (216.58.201.163) 56(84) bytes of data.
64 bytes from mad08s06-in-f3.1e100.net (216.58.201.163): icmp_seq=1 ttl=51 time=
21.7 ms
64 bytes from mad08s06-in-f3.1e100.net (216.58.201.163): icmp_seq=2 ttl=51 time=
32.0 ms
64 bytes from mad08s06-in-f3.1e100.net (216.58.201.163): icmp_seq=3 ttl=51 time=
29.1 ms
64 bytes from mad08s06-in-f3.1e100.net (216.58.201.163): icmp_seq=4 ttl=51 time=
29.3 ms
^C
--- www.google.es ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 21.779/28.097/32.096/3.832 ms
usuario@usuario-VirtualBox:~$
```

Data are sent in order to stop the sending, Ctrol+c will be pressed and at the end it is possible to check the results.

But there is a problem. When the server is rebooted, the establish rule is used in order to redirect the traffic from the Internal network to bridge adapter which is then removed. So, to avoid that, there are 2 different ways:

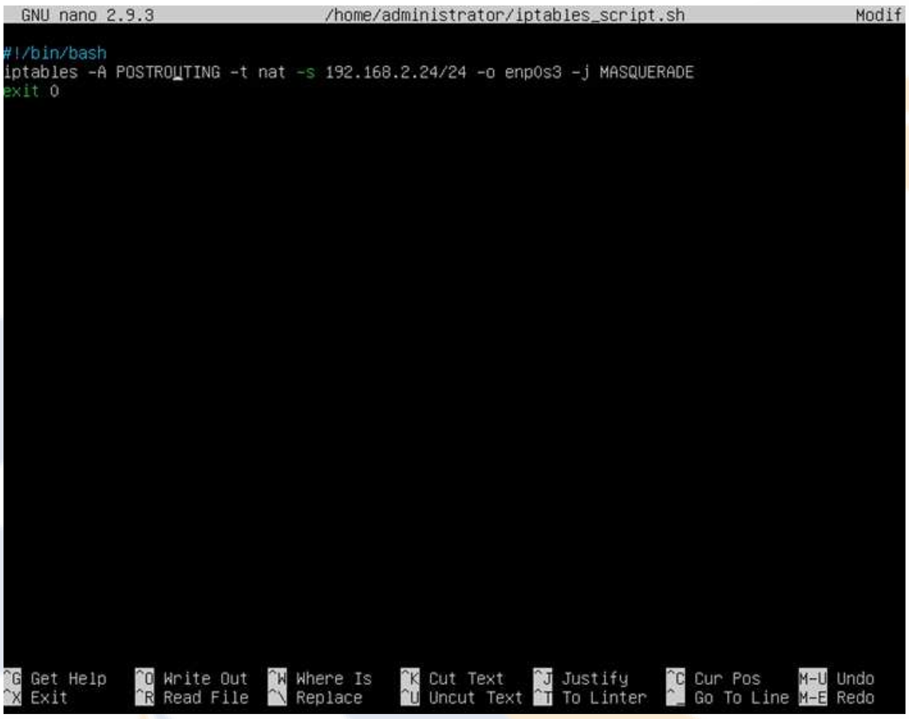
- Programming a task with crontab
- Enable rc.local with Systemd

2.1.1.1 Programming a task with crontab

First of all, a simple bash script should be written. This is considered as an advance of the bash script unit.

In order to write a bash script:

```
nano /home/administrator/iptables_script
```

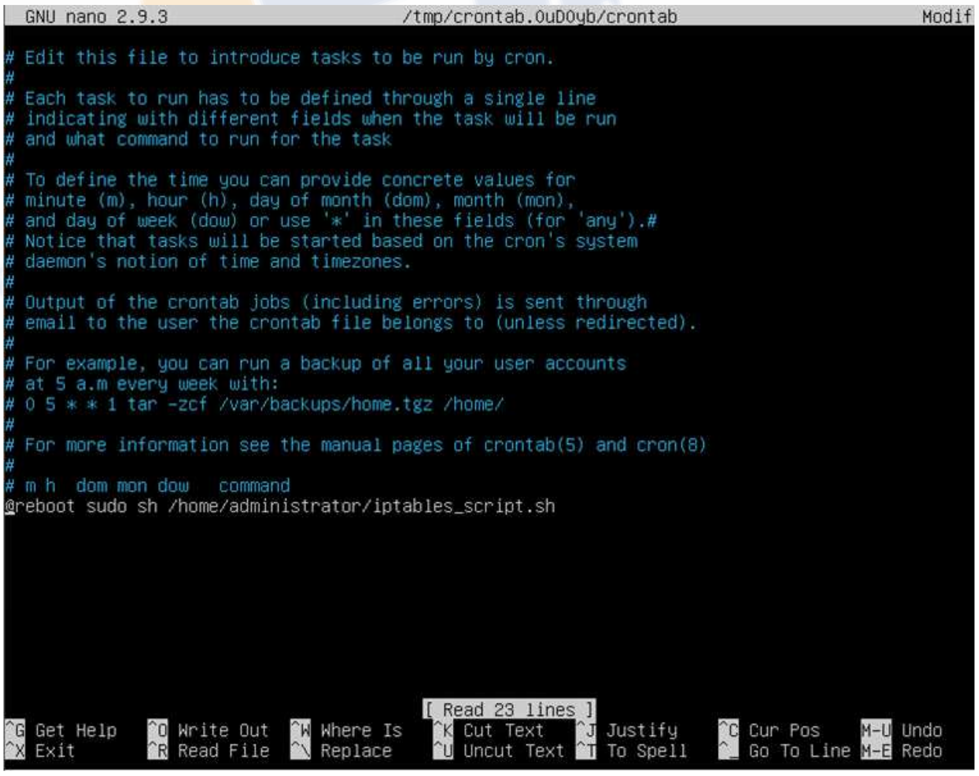


```
  GNU nano 2.9.3                /home/administrator/iptables_script.sh                Modif

#!/bin/bash
iptables -A POSTROUTING -t nat -s 192.168.2.24/24 -o enp0s3 -j MASQUERADE
exit 0

^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Linter  ^_ Go To Line M-E Redo
```

The task should be programmed when the computer is rebooted, so:

```
sudo crontab -e
```

And the file would be:



```
  GNU nano 2.9.3                  /tmp/crontab.0uD0yb/crontab                  Modif

# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
@reboot sudo sh /home/administrator/iptables_script.sh

                                  [ Read 23 lines ]
^G Get Help   ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos    M-U Undo
^X Exit       ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line M-E Redo
```

Check it

In order to check it:

```
sudo reboot
```

and check if the client can connect

```
ping 192.168.2.24
ping www.google.es
```

2.1.1.2 Enable rc.local with Systemd

rc.local file is a file that is run after the boot sequence. It contains commands or can contain even scripts which will be run after the boot sequence when operating system starts. But nowadays, Ubuntu 18.04, use Systemd service which replaces init which contains rc.local file with the main goal to join the basic configurations of Linux and services behaviours in all distributions.

So, /etc/rclocal script is needed to be enabled, and if it needs to run some command or script in the init process, perform the following:

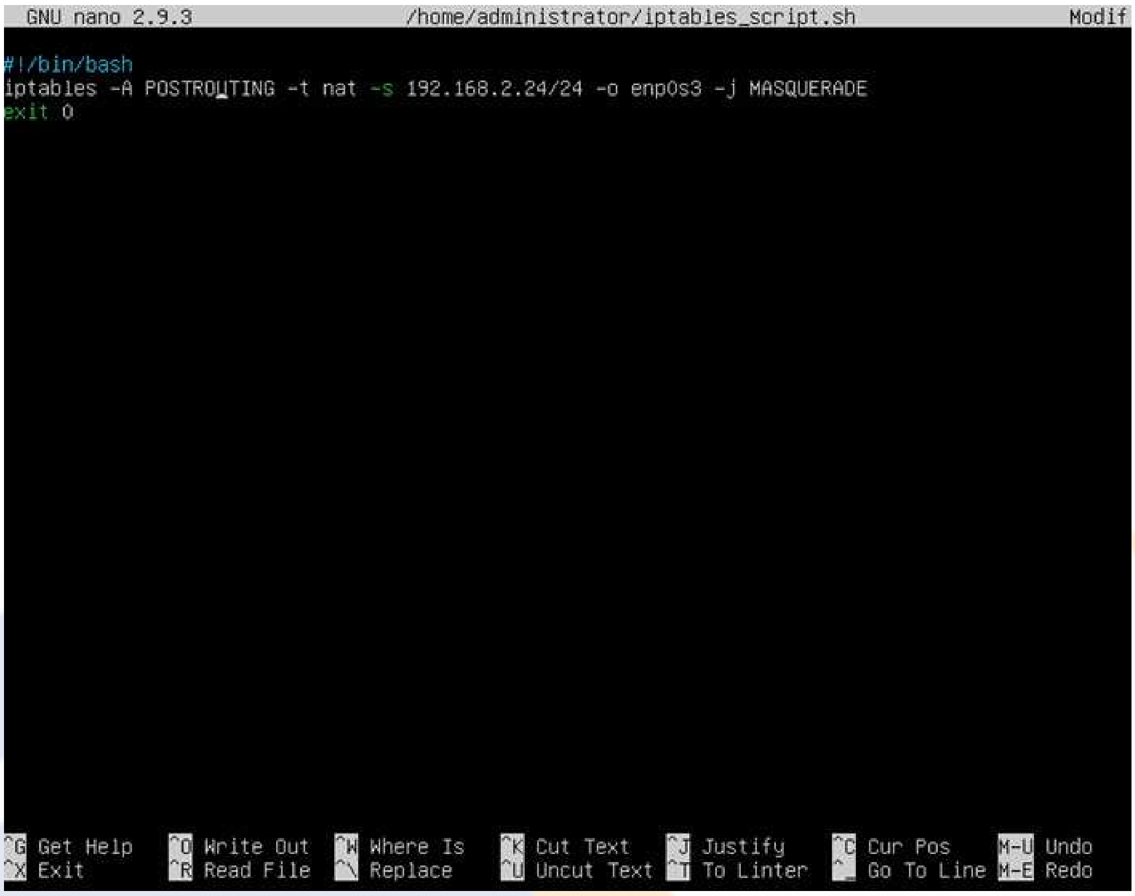
1. The first point is enable /etc/rc.local on Systemd so a file will be created with the following content:

   ```
   sudo nano /etc/systemd/system/rc-local.service
   ```

   ```
   [Unit]
    Description=/etc/rc.local Compatibility
    ConditionPathExists=/etc/rc.local

   [Service]
    Type=forking
    ExecStart=/etc/rc.local start
    TimeoutSec=0
    StandardOutput=tty
    RemainAfterExit=yes
    SysVStartPriority=99

   [Install]
    WantedBy=multi-user.target
   ```

   Remember save the file with ctrl+X

2. The file iptables_script.sh should be copied to /etc/scripts where the scripts folder should be created too. The file iptables_script should be the same file created in the point programming a task with crontab:



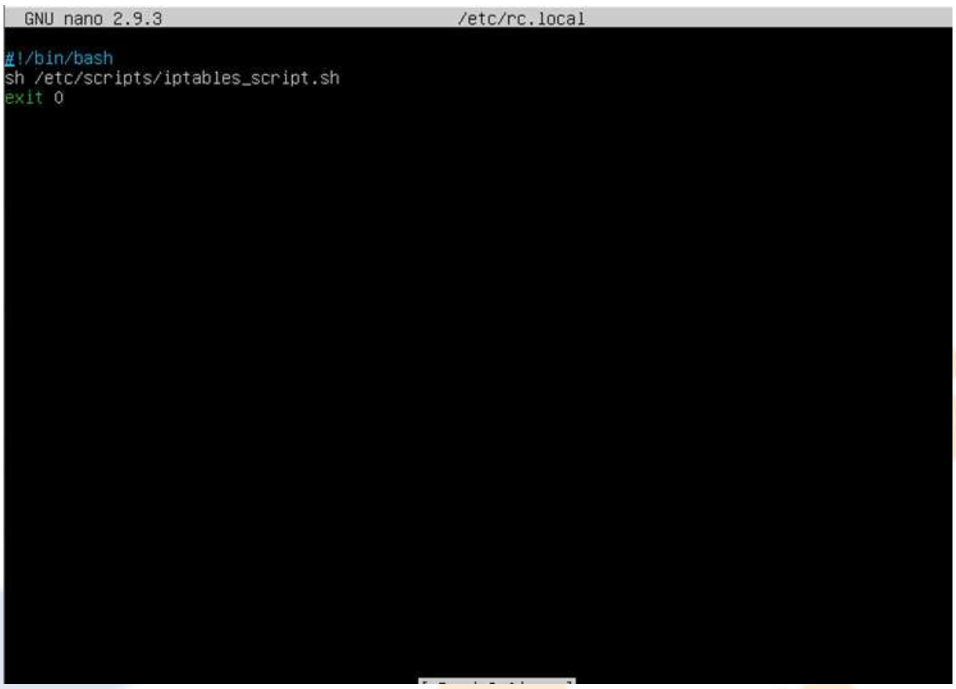
```
  GNU nano 2.9.3                /home/administrator/iptables_script.sh                Modif

#!/bin/bash
iptables -A POSTROUTING -t nat -s 192.168.2.24/24 -o enp0s3 -j MASQUERADE
exit 0
```

3. It is very important that the file has the suitable permissions in order to can be executed, so:

```
sudo chmod +x /etc/scripts/iptables_script.sh
```

4. Now, the rc.local file should be created. The content of this file will be what script or commands are needed to execute after the boot process.

```
sudo nano /etc/rc.local
```



```
GNU nano 2.9.3                       /etc/rc.local

#!/bin/bash
sh /etc/scripts/iptables_script.sh
exit 0
```

So, in that case, the iptables_script.sh created should be executed with the lines which are shown above.

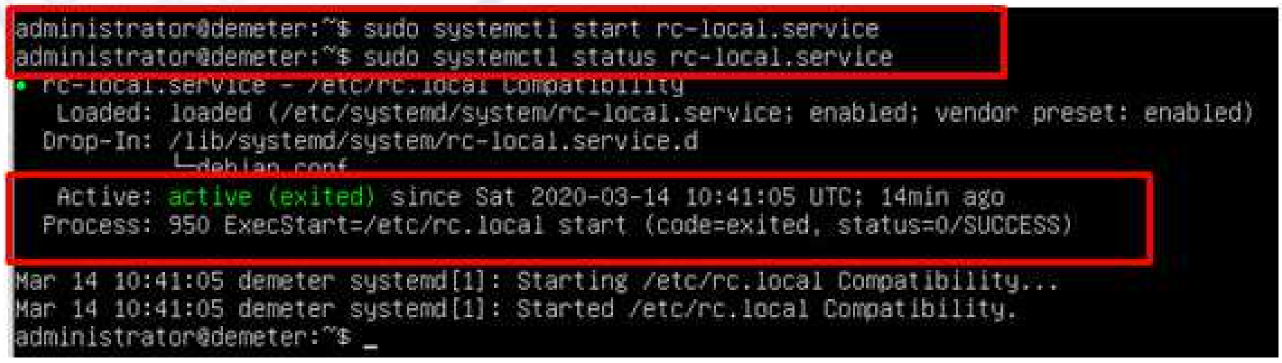
5. In the same way that the permission to the script was given, it is needed to do the same with rc.local file:

   `sudo chmod +x /etc/rc.local`

6. After that, the service on system boot should be enabled:

   `sudo systemctl enable rc-local`

7. Now, start the service and check its status:



```
administrator@demeter:~$ sudo systemctl start rc-local.service
administrator@demeter:~$ sudo systemctl status rc-local.service
● rc-local.service - /etc/rc.local Compatibility
   Loaded: loaded (/etc/systemd/system/rc-local.service; enabled; vendor preset: enabled)
  Drop-In: /lib/systemd/system/rc-local.service.d
           └─debian.conf
   Active: active (exited) since Sat 2020-03-14 10:41:05 UTC; 14min ago
  Process: 950 ExecStart=/etc/rc.local start (code=exited, status=0/SUCCESS)

Mar 14 10:41:05 demeter systemd[1]: Starting /etc/rc.local Compatibility...
Mar 14 10:41:05 demeter systemd[1]: Started /etc/rc.local Compatibility.
administrator@demeter:~$ _
```

The picture above shows the command in order to start and check the service, and the status which is active.

Check it

In order to check it:

```
sudo reboot
```

and check if the client can connect

```
ping 192.168.2.24
ping www.google.es
```

2.2 Configuring client machine

Once Internet connection has been configured, it is possible to update the client machine with:

```
sudo apt-get update
```

After updating the machine, the following steps describe how the client will be configured to have access to the shared folder and shared files.

1. It is necessary install the NFS Common client:

   ```
   sudo apt-get install nfs-common
   ```

2. The client computer needs a directory where all the shared content by the server in the export folder can be accessed. The same way that in the server, the folder will be created in the mnt directory:

   ```
   sudo mkdir -p /mnt/sharedfolder_client
   ```

3. In this step, the shared folder from the host will be mounted in the folder created on the client.

   ```
   sudo mount serverIP:/exportFolderServer /mnt/mountfolder_client
   ```

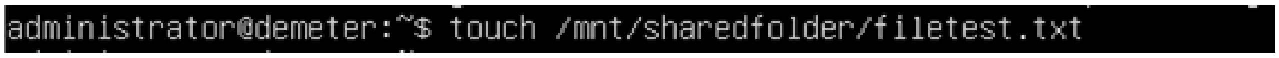
   In this case:

   ```
   sudo mount 192.168.2.24:/mnt/sharedfolder  /mnt/sharedfolder_client
   ```

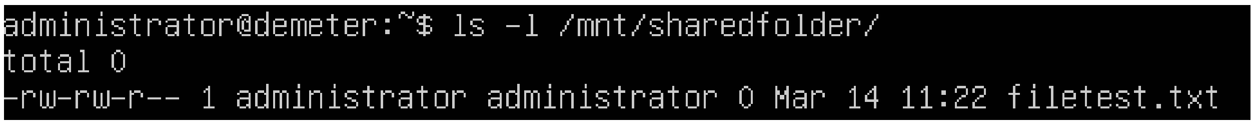
Mounting devices

Mount or umount commands are used for mounting or umounting devices, folders etc... That is used in Linux because the system needs to create, which is called a mount point in order to be able to use those devices or folders. Normally, that mount point is a folder.

4. Test. In order to test the correct performance, it is possible to create a file or folder in the export folder at server and open it at the client.

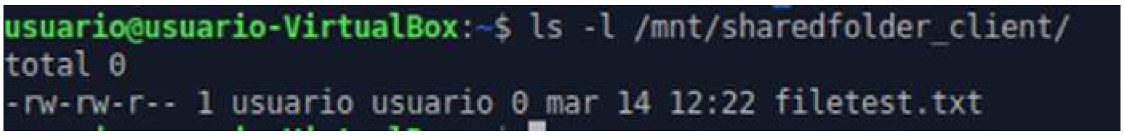


```
administrator@demeter:~$ touch /mnt/sharedfolder/filetest.txt
```



```
administrator@demeter:~$ ls -l /mnt/sharedfolder/
total 0
-rw-rw-r-- 1 administrator administrator 0 Mar 14 11:22 filetest.txt
```

5. Now, at the client machine:



```
usuario@usuario-VirtualBox:~$ ls -l /mnt/sharedfolder_client/
total 0
-rw-rw-r-- 1 usuario usuario 0 mar 14 12:22 filetest.txt
```

Mounting devices

Now, try to write some content at the file from the client and check if it is able to read the content at server and vice versa, write some content at the file from the server, and check if it is possible to read it at the client machine.