

## Práctica 6 SWAP

El objetivo de esta práctica es crear un servidor NFS en el que se puedan almacenar y compartir datos. El fin último de esto es que podamos tener las bases de datos de la granja web en nuestro servidor NFS, y que los servidores web se comuniquen con el mismo para recibir los datos.

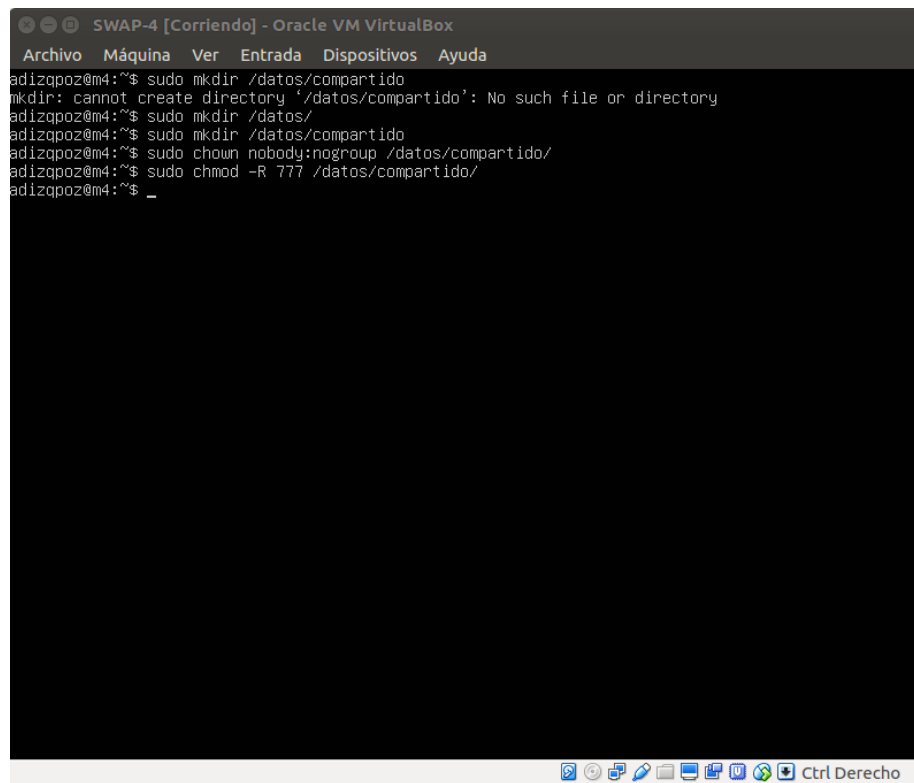
Al acabar esta práctica sabemos tener servidores dedicados al balanceo de carga, servidores dedicados a servir los sitios web y servidores que almacenan las bases de datos que utilizan nuestros sitios web, y todo ello correctamente configurado. Con estos conocimientos somos capaces de crear una granja web de forma escalable, ya que si deseamos mayor capacidad, simplemente debemos adquirir y crear más máquinas, además de añadirlas a las distintas configuraciones de seguridad o de replicado de datos.

### 1. Creación del servidor NFS

Para crear nuestro servidor NFS debemos crear una nueva máquina virtual al igual que creamos las otras máquinas, con un adaptador NAT y un adaptador sólo-anfitrión, el cual tiene una dirección IP estática, que en este caso será la 192.168.56.104.

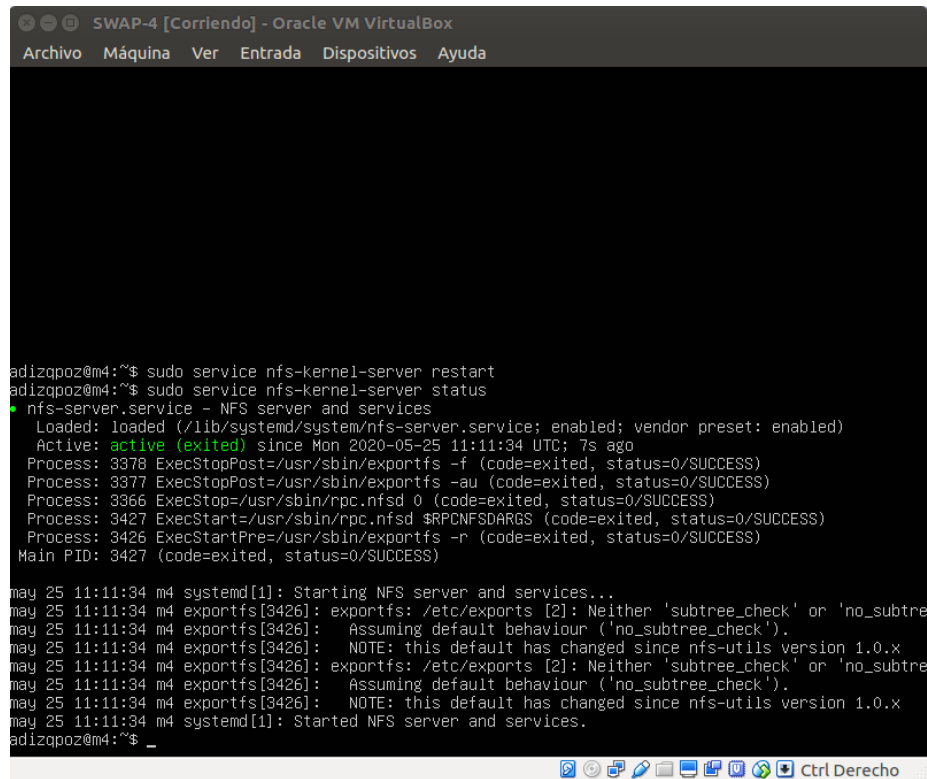
Posteriormente instalaremos el servicio NFS, tanto para servidor como para cliente, y rpcbind, dado que es necesario para que los servidores web puedan acceder de forma remota.

Posteriormente lo que hacemos es crear el directorio que tenemos pensado compartir a nuestros servidores web. En este caso serán simples archivos de texto, pero se puede aplicar a estructuras de datos más complejas de ser necesario.



```
SWAP-4 [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
adizqpoz@m4:~$ sudo mkdir /datos/compartido
mkdir: cannot create directory '/datos/compartido': No such file or directory
adizqpoz@m4:~$ sudo mkdir /datos/
adizqpoz@m4:~$ sudo mkdir /datos/compartido
adizqpoz@m4:~$ sudo chown nobody:nogroup /datos/compartido/
adizqpoz@m4:~$ sudo chmod -R 777 /datos/compartido/
adizqpoz@m4:~$ _
```

Ahora damos permiso para que los dos servidores web puedan acceder a nuestra carpeta compartida, y al reiniciar el servicio todo debe estar correcto.



```
adizqpoz@m4:~$ sudo service nfs-kernel-server restart
adizqpoz@m4:~$ sudo service nfs-kernel-server status
• nfs-server.service - NFS server and services
   Loaded: loaded (/lib/systemd/system/nfs-server.service; enabled; vendor preset: enabled)
   Active: active (exited) since Mon 2020-05-25 11:11:34 UTC; 7s ago
     Process: 3378 ExecStopPost=/usr/sbin/exportfs -f (code=exited, status=0/SUCCESS)
     Process: 3377 ExecStopPost=/usr/sbin/exportfs -au (code=exited, status=0/SUCCESS)
     Process: 3366 ExecStop=/usr/sbin/rpc.nfsd 0 (code=exited, status=0/SUCCESS)
     Process: 3427 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited, status=0/SUCCESS)
     Process: 3426 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
    Main PID: 3427 (code=exited, status=0/SUCCESS)

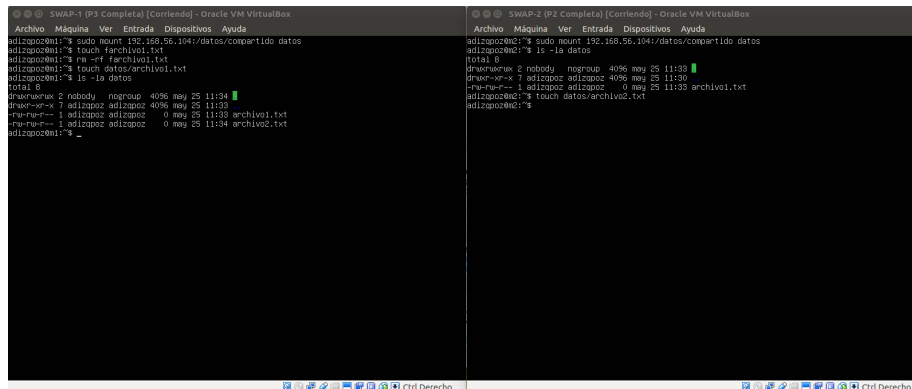
may 25 11:11:34 m4 systemd[1]: Starting NFS server and services...
may 25 11:11:34 m4 exportfs[3426]: exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree
may 25 11:11:34 m4 exportfs[3426]: Assuming default behaviour ('no_subtree_check').
may 25 11:11:34 m4 exportfs[3426]: NOTE: this default has changed since nfs-utils version 1.0.x
may 25 11:11:34 m4 exportfs[3426]: exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree
may 25 11:11:34 m4 exportfs[3426]: Assuming default behaviour ('no_subtree_check').
may 25 11:11:34 m4 exportfs[3426]: NOTE: this default has changed since nfs-utils version 1.0.x
may 25 11:11:34 m4 systemd[1]: Started NFS server and services.
adizqpoz@m4:~$ _
```

## 2. Configuración de los clientes

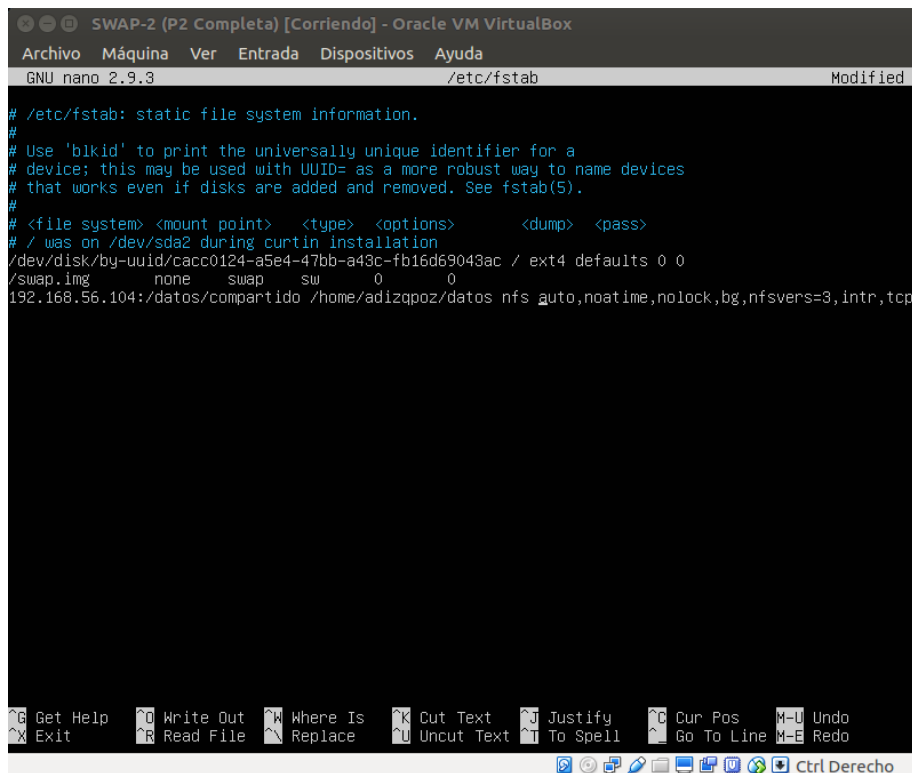
En este apartado lo que debemos hacer es configurar a los servidores web para que puedan acceder a nuestro servidor nfs mediante una carpeta compartida.

Para ello, instalamos el cliente NFS y rpcbind. Posteriormente, en una ruta cualquiera, en nuestro caso, el home de nuestro usuario, creamos una carpeta *datos*, la cual será la carpeta compartida. Además, damos permiso a todos los usuarios.

Posteriormente, montamos la carpeta exportada por el servidor en el cliente, y comprobamos que todo funciona correctamente.



Sin embargo, cuando montamos la carpeta no se trata de un cambio permanente en el sistema. Por tanto, debemos añadir una línea en el archivo `/etc/fstab`, el cual controla el sistema de montaje de dispositivos, para montar la carpeta compartida por el servidor NFS.



Tras ello ya no hemos de preocuparnos por montar manualmente la carpeta compartida cada vez que arranquemos el sistema.

### 3. Seguridad en el servidor NFS

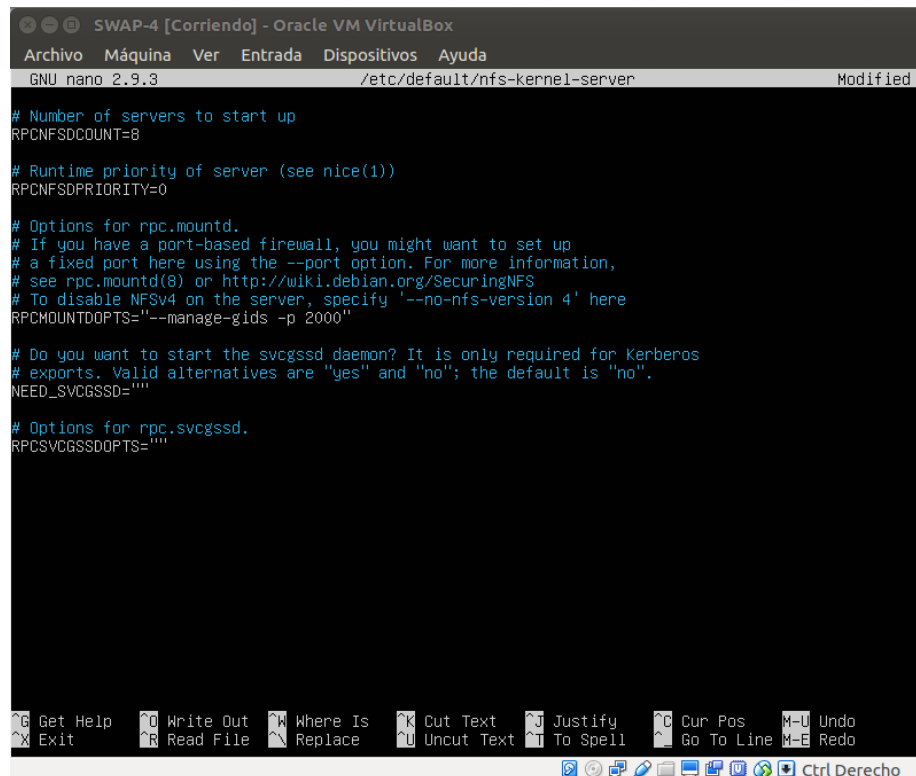
En este apartado asumimos que los servidores web no tienen el cortafuegos habilitado. En un sistema real, cada uno de los cortafuegos de los servidores web deben poder enviar y recibir peticiones por los puertos que se utilizan para hacer posible el sistema NFS, los cuales enumeraremos posteriormente.

En primer lugar, utilizamos las reglas necesarias para denegar todo tráfico entrante, y aceptar las conexiones establecidas y relacionadas.

Posteriormente debemos tener en cuenta los servicios que se utilizan:

- nfs: utiliza el puerto 2049 para tcp y udp.
- portmapper: utiliza el puerto 111 para tcp y udp
- mountd: por defecto utiliza puertos dinámicos. Para fijar un puerto para este servicio debemos modificar una línea en el archivo `/etc/default/nfs-kernel-server`, para que escuche específicamente el puerto 2000, por ejemplo. Es decir, debemos tener una línea tal que así:

```
RPCMOUNTDOPTS="--manage-gids -p 2000"
```



```
SWAP-4 [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 2.9.3 /etc/default/nfs-kernel-server Modified

# Number of servers to start up
RPCNFSDCOUNT=8

# Runtime priority of server (see nice(1))
RPCNFSDPRIORITY=0

# Options for rpc.mountd.
# If you have a port-based firewall, you might want to set up
# a fixed port here using the --port option. For more information,
# see rpc.mountd(8) or http://wiki.debian.org/SecuringNFS
# To disable NFSv4 on the server, specify '--no-nfs-version 4' here
RPCMOUNTDOPTS="--manage-gids -p 2000"

# Do you want to start the svcgssd daemon? It is only required for Kerberos
# exports. Valid alternatives are "yes" and "no"; the default is "no".
NEED_SVCSSD=""

# Options for rpc.svcgssd.
RPCSVCGSSDOPTS=""

^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  ^N Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line M-E Redo
Ctrl Derecho
```

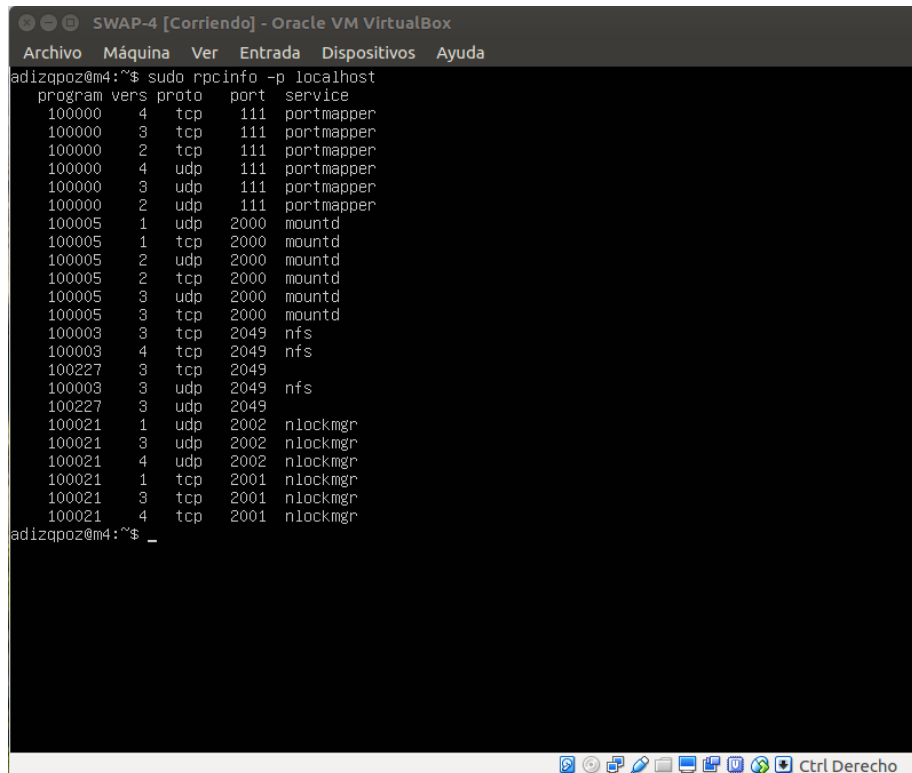
- nlockmgr: es parte de un módulo del kernel. Debemos crear un archivo de configuración en la carpeta `/etc/sysctl.d/` donde habilitaremos para este

servicio el puerto 2001 para tcp y 2002 para udp:

```
fs.nfs.nlm_tcpport = 2001
fs.nfs.nlm_udpport = 2002
```

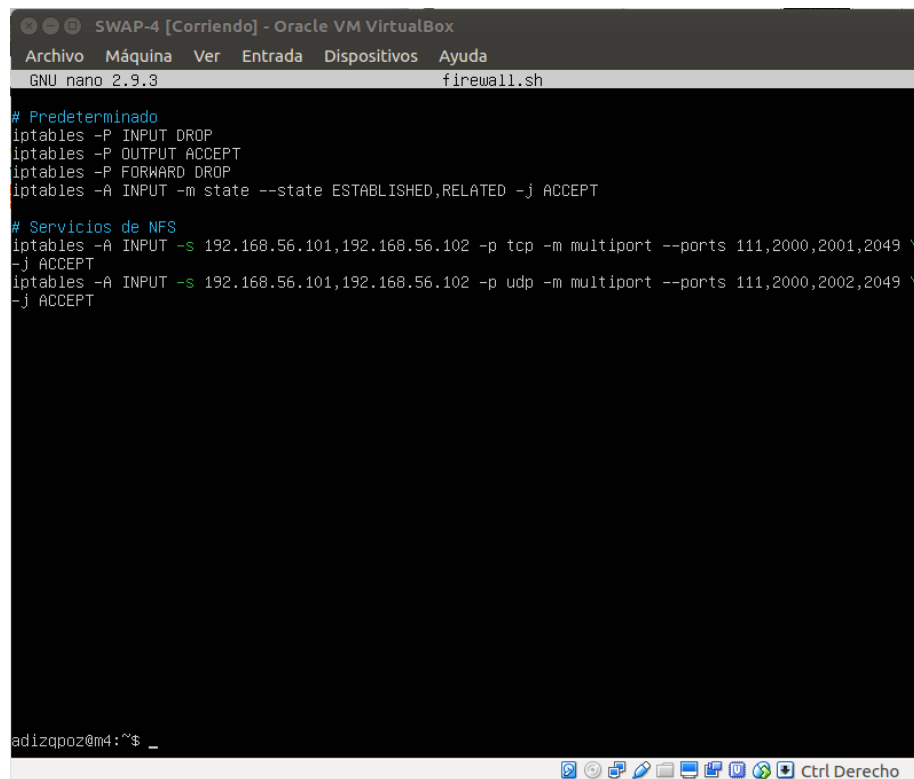
Posteriormente lanzamos el nuevo archivo de configuración, y reiniciamos el servidor NFS.

Ahora será útil comprobar que utilizamos los puertos que hemos fijado con el comando `rpcinfo -p server`:



```
SWAP-4 [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
adizqpoz@m4:~$ sudo rpcinfo -p localhost
program vers proto port service
100000 4 tcp 111 portmapper
100000 3 tcp 111 portmapper
100000 2 tcp 111 portmapper
100000 4 udp 111 portmapper
100000 3 udp 111 portmapper
100000 2 udp 111 portmapper
100005 1 udp 2000 mountd
100005 1 tcp 2000 mountd
100005 2 udp 2000 mountd
100005 2 tcp 2000 mountd
100005 3 udp 2000 mountd
100005 3 tcp 2000 mountd
100003 3 tcp 2049 nfs
100003 4 tcp 2049 nfs
100227 3 tcp 2049 nfs
100003 3 udp 2049 nfs
100227 3 udp 2049 nfs
100021 1 udp 2002 nlockmgr
100021 3 udp 2002 nlockmgr
100021 4 udp 2002 nlockmgr
100021 1 tcp 2001 nlockmgr
100021 3 tcp 2001 nlockmgr
100021 4 tcp 2001 nlockmgr
adizqpoz@m4:~$
```

Sabiendo esto, ya podemos configurar correctamente el tráfico que puede entrar, no salir. Determinamos de dónde puede venir el tráfico TCP y UDP y sus puertos con las dos últimas reglas de este script:



```
SWAP-4 [Corriendo] - Oracle VM VirtualBox
Archivo  Máquina  Ver  Entrada  Dispositivos  Ayuda
GNU nano 2.9.3  firewall.sh

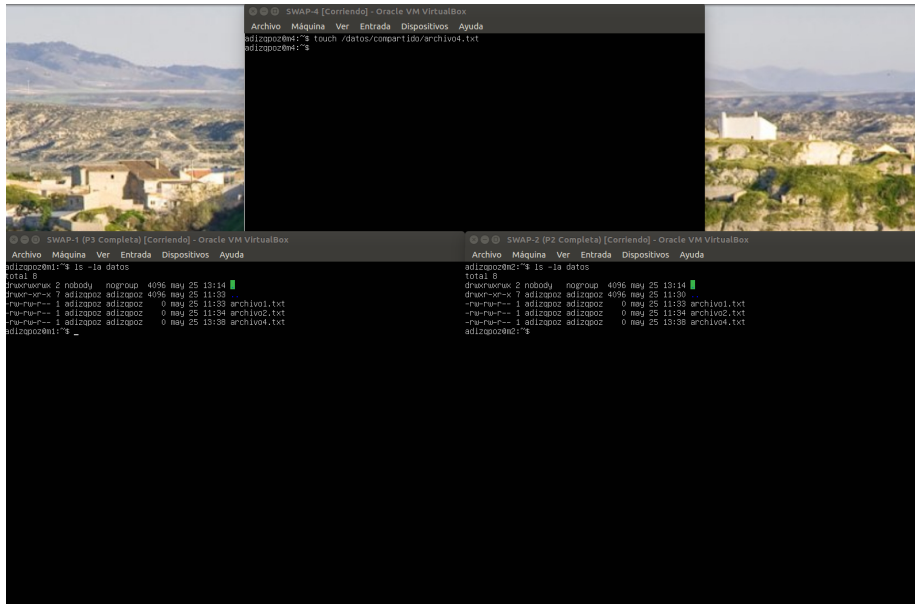
# Predeterminado
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Servicios de NFS
iptables -A INPUT -s 192.168.56.101,192.168.56.102 -p tcp -m multiport --ports 111,2000,2001,2049 \
-j ACCEPT
iptables -A INPUT -s 192.168.56.101,192.168.56.102 -p udp -m multiport --ports 111,2000,2002,2049 \
-j ACCEPT

adizqpoz@m4:~$
```

Este script, tal como hicimos en la práctica 4, lo ejecutaremos en cada arranque con el demonio cron.

Tras reiniciar, realizamos la prueba de que todo funciona correctamente creando un archivo en la carpeta compartida desde M4:



---

Autor: Adrián Izquierdo Pozo

Si desea ver el archivo Markdown puede verlo en mi repositorio de Github