

SWAP: Servidor de disco NFS

David Cabezas Berrido

dxabezas@correo.ugr.es

26 de mayo de 2021

Índice

1. Preparativos	2
2. Configurar servidor NFS	2
3. Configurar los clientes M1 y M2	2
3.1. Hacer la configuración permanente	3
4. Seguridad en el servidor NFS	3

1. Preparativos

Creamos una nueva máquina virtual llamada **NFS-dxabezas**. Al igual que las otras máquinas, configuramos el doble adaptador de red (NAT + Solo-Anfitrión), instalamos Ubuntu Server 18.04.1 y creamos un usuario *dxabezas* con contraseña *Swap1234*. Comprobamos mediante PING que la máquina NFS tiene conexión con el resto de máquinas en la granja.

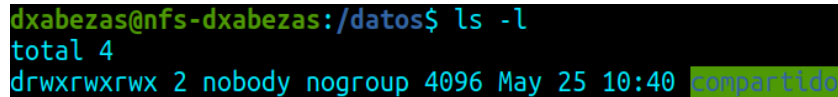
Su IP (en la red local) es 192.168.56.108.

2. Configurar servidor NFS

Comenzamos creando la carpeta a compartir, y cambiamos el propietario y los permisos.

```
sudo mkdir -p /datos/compartido
sudo chown nobody:nogroup /datos/compartido/
sudo chmod -R 777 /datos/compartido/
```

Comprobamos el propietario y los permisos que acabamos de asignar.

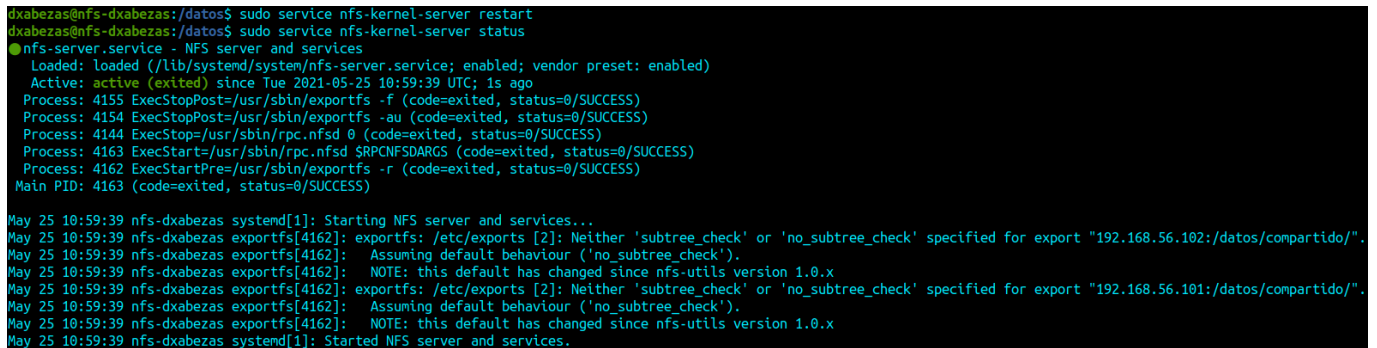


```
dxabezas@nfs-dxabezas:/datos$ ls -l
total 4
drwxrwxrwx 2 nobody nogroup 4096 May 25 10:40 compartido
```

Le damos permisos a M1 y M2 añadiendo la siguiente línea a */etc/exports*.

```
/datos/compartido/ 192.168.56.102(rw) 192.168.56.101(rw)
```

Finalmente, reiniciamos el servicio y comprobamos el estado.



```
dxabezas@nfs-dxabezas:/datos$ sudo service nfs-kernel-server restart
dxabezas@nfs-dxabezas:/datos$ 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 Tue 2021-05-25 10:59:39 UTC; 1s ago
     Process: 4155 ExecStopPost=/usr/sbin/exportfs -f (code=exited, status=0/SUCCESS)
     Process: 4154 ExecStopPost=/usr/sbin/exportfs -au (code=exited, status=0/SUCCESS)
     Process: 4144 ExecStop=/usr/sbin/rpc.nfsd 0 (code=exited, status=0/SUCCESS)
     Process: 4163 ExecStart=/usr/sbin/rpc.nfsd $RPCNFSDARGS (code=exited, status=0/SUCCESS)
     Process: 4162 ExecStartPre=/usr/sbin/exportfs -r (code=exited, status=0/SUCCESS)
    Main PID: 4163 (code=exited, status=0/SUCCESS)

May 25 10:59:39 nfs-dxabezas systemd[1]: Starting NFS server and services...
May 25 10:59:39 nfs-dxabezas exportfs[4162]: exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree_check' specified for export "192.168.56.102:/datos/compartido/".
May 25 10:59:39 nfs-dxabezas exportfs[4162]: Assuming default behaviour ('no_subtree_check').
May 25 10:59:39 nfs-dxabezas exportfs[4162]: NOTE: this default has changed since nfs-utils version 1.0.x
May 25 10:59:39 nfs-dxabezas exportfs[4162]: exportfs: /etc/exports [2]: Neither 'subtree_check' or 'no_subtree_check' specified for export "192.168.56.101:/datos/compartido/".
May 25 10:59:39 nfs-dxabezas exportfs[4162]: Assuming default behaviour ('no_subtree_check').
May 25 10:59:39 nfs-dxabezas exportfs[4162]: NOTE: this default has changed since nfs-utils version 1.0.x
May 25 10:59:39 nfs-dxabezas systemd[1]: Started NFS server and services.
```

Figura 1: Todo parece estar correcto, teniendo en cuenta que estamos asumiendo la opción `no_subtree_check`. El subtree check consiste en comprobar que cada petición NFS solicita sólo archivos que están siendo exportados, para ello necesita información sobre el directorio padre.

3. Configurar los clientes M1 y M2

La siguiente configuración se realiza tanto en M1 como en M2.

Primero instalamos los paquetes necesarios.

```
sudo apt install nfs-common rpcbind
```

A continuación, creamos el punto de montaje, el directorio **datos**. También le damos todos los permisos sobre él a todos los usuarios.

```
cd /home/dxabezas
mkdir datos
chmod -R 777 datos
```

Si ahora modificamos la carpeta compartida en alguna máquina (M1, M2 o servidor NFS), los cambios también se hacen efectivos en las otras dos.

```
dxabezas@m1-dxabezas:~$ ls -la datos/
total 8
drwxrwxrwx 2 nobody nogroup 4096 May 25 10:40 .
drwxr-xr-x 9 dxabezas dxabezas 4096 May 25 15:37 ..
dxabezas@m1-dxabezas:~$ touch datos/archivo.txt

dxabezas@m2-dxabezas:~$ ls -la datos/
total 8
drwxrwxrwx 2 nobody nogroup 4096 May 25 15:44 .
drwxr-xr-x 9 dxabezas dxabezas 4096 May 25 16:24 ..
-rw-rw-r-- 1 dxabezas dxabezas 0 May 25 15:44 archivo.txt

dxabezas@nfs-dxabezas:~$ ls -la /datos/compartido
total 8
drwxrwxrwx 2 nobody nogroup 4096 May 25 15:44 .
drwxr-xr-x 3 root root 4096 May 25 10:40 ..
-rw-rw-r-- 1 dxabezas dxabezas 0 May 25 15:44 archivo.txt
```

Figura 2: El archivo creado por M1 aparece en M2 y en el servidor NFS.

3.1. Hacer la configuración permanente

Para hacer la configuración permanente, sólo tenemos que añadir (tanto en M1 como en M2) la siguiente línea al fichero `/etc/fstab`.

```
192.168.56.108:/datos/compartido /home/dxabezas/datos/ nfs
auto,noatime,nolock,bg,nfsvers=3,intr,tcp,actimeo=1800 0 0
```

Al reiniciar la máquina, comprobamos que el directorio está montado y podemos acceder a los archivos.

4. Seguridad en el servidor NFS

Nuestro objetivo es configurar IPTABLES para que el servidor NFS siga funcionando mientras denegamos implícitamente el tráfico entrante. Partimos por tanto de la siguiente configuración:

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

Para los servicios **nfs** y **portmapper**, sólo tenemos que abrir los puertos 2049 y 111 respectivamente, tanto TCP como UDP. Sin embargo, los servicios **mountd** y **nlockmgr** utilizan puertos dinámicos. Podemos consultar qué puerto está utilizando cada servicio con la siguiente orden.

```

dxabezas@nfs-dxabezas:~$ rpcinfo -p
  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
    100003   3  udp    2049  nfs
    100227   3  udp    2049
    100021   1  udp    49400 nlockmgr
    100021   3  udp    49400 nlockmgr
    100021   4  udp    49400 nlockmgr
    100021   1  tcp    42601 nlockmgr
    100021   3  tcp    42601 nlockmgr
    100021   4  tcp    42601 nlockmgr

```

Como no podemos abrir puertos dinámicamente con IPTABLES, fijaremos puertos para estos servicios. Para **mountd**, modificamos el archivo `/etc/default/nfs-kernel-server` y añadimos la siguiente línea.

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

Para **nlockmgr**, creamos el archivo `/etc/sysctl.d/swap-nfs-ports.conf` con las siguientes opciones

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

Lanzamos el archivo de configuración y reiniciamos el servidor NFS con las siguientes órdenes.

```
sudo sysctl --system
/etc/init.d/nfs-kernel-server restart
```

La primera de ellas devuelve una salida muy extensa de configuraciones que se aplican. Entre ellas, está la siguiente secuencia.

```
* Applying /etc/sysctl.d/swap-nfs-ports.conf ...
fs.nfs.nlm_tcpport = 2001
fs.nfs.nlm_udpport = 2002
```

La segunda orden produce la siguiente salida.

```
[ ok ] Restarting nfs-kernel-server (via systemctl): nfs-kernel-server.service.
```

Como parece que todo va correctamente, volvemos a comprobar los puertos que utiliza cada servicio.

```

dxabezas@nfs-dxabezas:~$ sudo rpcinfo -p
  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
    100003   3  udp    2049  nfs
    100227   3  udp    2049
    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

```

Como podemos observar, son justo los puertos que hemos seleccionado. Ahora podemos abrirlos con IPTABLES (sólo para M1 y M2) como solemos hacer. Nuestro script de IPTABLES queda de esta forma.

```

#!/bin/sh

# 1: Eliminar todas las reglas, configuración limpia
iptables -F
iptables -X
iptables -Z

# 2: Denegación implícita del tráfico entrante
iptables -P INPUT DROP
iptables -P OUTPUT ACCEPT
iptables -P FORWARD DROP

# 3: Permitir conexiones
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# 4: Permitir acceso desde localhost (interface lo):
iptables -A INPUT -i lo -j ACCEPT
iptables -A OUTPUT -o lo -j ACCEPT

# 5: Abrir puerto 2049 (nfs), TCP y UDP, sólo M1 y M2
iptables -A INPUT -p tcp -s 192.168.56.102 --dport 2049 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.102 --dport 2049 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.56.101 --dport 2049 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.101 --dport 2049 -j ACCEPT

# 6: Abrir puerto 111 (portmapper), TCP y UDP, sólo M1 y M2
iptables -A INPUT -p tcp -s 192.168.56.102 --dport 111 -j ACCEPT

```

```
iptables -A INPUT -p udp -s 192.168.56.102 --dport 111 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.56.101 --dport 111 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.101 --dport 111 -j ACCEPT
```

7: Abrir puerto 2000 (mountd), TCP y UDP, sólo M1 y M2

```
iptables -A INPUT -p tcp -s 192.168.56.102 --dport 2000 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.102 --dport 2000 -j ACCEPT
iptables -A INPUT -p tcp -s 192.168.56.101 --dport 2000 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.101 --dport 2000 -j ACCEPT
```

8: Abrir puertos para nlockmgr: 2001 (TCP) y 2002 (UDP), sólo M1 y M2

```
iptables -A INPUT -p tcp -s 192.168.56.102 --dport 2001 -j ACCEPT
iptables -A INPUT -p udp -s 192.168.56.102 --dport 2002 -j ACCEPT
```

Activamos esta configuración. Para comprobar que sigue funcionando, realizamos satisfactoriamente una modificación similar a la de la Figura 2, modificando también desde M1.