NFS (Network File System)

Iván Belmonte < ttyp0@inet2u.com>
24-4-2k2

1. Qué es NFS?

NFS significa **Network File System**
NFS se desarrolló con el fin de permitir montar particiones remotamente entre máquinas situadas en una misma red. Puede entenderse como algo parecido a lo que es un servidor de ficheros en Window$. Para compartir particiones con otras máquinas dentro de una red, el servidor simplemente *comparte* un directorio, y el cliente lo *monta* de un modo similar a como sucede al montar un CDrom o una partición de un disco duro.

· Versión actual: **NFS versión 3 (kernels 2.4 de Linux o superiores)**
· Site oficial: **http://nfs.sourceforge.net**
· Software necesario para configurar un servidor o cliente NFS: **kernel 2.4.0 o superior, nfs-utils (última versión es la 0.3.3 a fecha de 24 de Abril de 2k2)**

2. Configuración de un servidor NFS

Es necesario editar solamente 1 fichero para que nuestro servidor de NFS funcione, en el que especificaremos los directorios que queremos compartir y con quién los compartiremos, además de las condiciones en que vamos a hacerlo. Haciendo esto nuestro servidor NFS funcionará perfectamente, aunque estaremos expuestos a un tremendo fallo de seguridad, de modo que se recomienda editar otros dos ficheros que controlen los accesos.
El primero de ellos, el fichero que especifica los directorios, máquinas y permisos, se encuentra en **/etc/exports**. Este fichero debe tener una estructura similar a la siguiente:

```
----------------------[ cut here ]---------------------------
Directorio-local  IP-cliente(opcion1,opcion2,opcion3...)  IP-cliente2(opcion1,opcion2,opcion3...)
Directorio-local2  IP-cliente(opciones...)  IP-cliente2(opciones...)
(...)
----------------------[ cut here ]---------------------------
```

· Directorio local: puede ser cualquier directorio, normalmente suele utilizarse NFS para compartir *home's* de usuarios en diferentes máquinas, o directorios a los que tengan que tener acceso varias máquinas. Al compartir un directorio TAMBIEN se comparten sus subdirectorios.

· IP-cliente: dirección IP del cliente. Puede ser especificada como IP o como NOMBRE, es decir, es lo mismo especificar *ttyp0.mine.nu* que *217.126.51.143*
Del mismo modo cabe decir que las IP's de los clientes pueden especificarse como **un solo host**, como **un rango de IP's** , se pueden **usar wildcards (comodines, '*' por ejemplo)** y se puede hacer referencia a un **grupo de red** concreto. Aquí teneis algunos ejemplos:

ttyp0.mine.nu
markitos@assl.ath.cx
**.mine.nu*
@assl
192.168.1.0/255.255.255.0
192.168.1.0/24
ttyp0@192.168.1.10

· Opciones: especificación del modo en que queremos compartir ese directorio. Podemos espeificar si queremos dar permiso de lectura solamente, de lectura y escritura, de escritura como root, de escritura pero no como root... las posibles opciones son listadas a continuación:

ro -> el directorio se comparte en modo 'read-only', de manera que el cliente no podrá escribir en él una vez montado.
rw -> el cliente tendrá acceso al directorio con permiso de lectura y escritura
no_root_squash -> Normalmente cuando el usuario *DE LA MAQUINA CLIENTE* que hace una petición para montar una particion es el *root*, sus permisos sobre la partición montada serán de usuario *nobody*. Si añadimos esta opción, permitiremos que el usuario *root* de la máquina cliente también tenga acceso *root* a la máquina servidor, cosa que no nos interesa en lo más mínimo si no es con un buen motivo.
root_squash -> especifica justamente lo contrario, es decir, que se mapeen los uid de *root* a *nobody*.
all_squash -> especifica que se mapeen los uid de *cualquier usuario* a *nobody*.
anonuid y **anongid** -> especifican los uid y gid que queremos que figuren en las peticiones anonimas... es decir, por ejemplo si tenemos compartido el directorio **/home/pedro** quiza nos interese que las peticiones para montar ese directorio aparezcan todas con el uid y el gid del usuario **pedro**. Se hace especificando estas dos opciones ;-)
no_subtree_check -> si solamente está *exportada* parte de un volumen, una rutina llamada 'subtree checking' comprovará que el fichero pedido por el cliente se encuentra en la parte del volumen que está compartida. Si sabemos sobradamente que el directorio que estamos compartiendo se encuentra enteramente en el mismo volumen, podemos deshabilitar esta opcion para evitar tráfico y tiempo.
sync -> el servidor avisa al cliente de la escritura de un fichero cada vez que se finaliza dicha escritura. Es importante por seguridad para los datos ante la posibilidad de la caida del servidor.
nohide -> Si la máquina cliente tiene un sistema montado en un directorio, y exporta este directorio, el cliente que lo monte deberá montar también el directorio 'hijo' que el servidor tiene montado, para poderlo ver. Esto sucede porque el directorio montado previamente por el sevridor esta 'oculto'. Podemos evitar todo este engorro activando la opcion *nohide*.

Con la especificacion de estas opciones en el fichero **/etc/exports** junto a los directorios y las máquinas que deben tener acceso ya tenemos montado un servidor de NFS que puede funcionar perfectamente, aunque como decíamos antes, es un caso muy inseguro dejar esto así, de modo que debemos protegernos de los ataques malintencionados. Lo que se trata es de evitar que NADIE monte particiones de nuestro servidor salvo los usuarios y máquinas que NOSOTROS hemos especificado expresamente.

3. Seguridad

Los ficheros que debemos editar ahora son **/etc/hosts.allow** y **/etc/hosts.deny** con tal de controlar los accesos a nuestro servidor.

Lo que el servidor hará una vez recibida una petición será comprobar el fichero **/etc/hosts.allos**. Si encuentra una descripción que coincida con la petición del cliente, permitirá su acceso. Si ninguna entrada coincide con la descripción del cliente, buscará en **/etc/hosts.deny** si figura allí, y en tal caso la petición será rechazada. Finalmente, si en ninguno de los dos ficheros aparece descripción alguna que coincida con el cliente, por defecto *SE PERMITIRA EL ACCESO*.

NOTA: Los ficheros **/etc/hosts.allow** y **/etc/hosts.deny** controlan normalmente el acceso a la ejecución de los demonios lanzados desde **inetd** (internet superserver daemon) tales como FTP, TELNET, etc... aunque también pueden controlar el acceso a los demonios ofrecidos por el servidor de NFS.

El primer demonio al que debemos restringir el acceso es el *portmapper*, que especifica a los clientes la manera de encontrar en el sistema el servicio que buscan. Restringir el acceso al *portmapper* es posiblemente la mejor defensa que podemos tener contra los ataques de NFS, ya que de este modo impedimos que sepan encontrar los recursos que nuestro servidor de NFS ofrece.

La estructura que debería tener nuestro **/etc/hosts.deny** es la siguiente:
----------[cut here]----------
portmap:ALL
lockd:ALL
mountd:ALL
rquotad:ALL
statd:ALL
----------[cut here]----------

Algunos sysadmins suelen especificar directamente *ALL:ALL* denegando el acceso a cuelquier recurso o servicio. Realmente es la opción más segura, aunque posiblemente nos volvamos locos cuando dentro de algún tiempo estemos configurando algún servicio nuevo, y habiéndonos olvidado ya del **/etc/hosts.deny** nuestro servicio no funcione, y no tengamos ni idea de por qué.

La sintaxis de **/etc/hosts.allow** debe ser del estilo:
servicio: host [o red/mascara], host [o red/mascara]

La estructura de **/etc/hosts.allow** debe ser similar a la siguiente:
----------[cut here]----------
portmap: 192.168.1.10, 192.168.1.100, 172.26.0.0/24
lockd: 192.168.1.10, 192.168.1.100, 172.26.0.0/24
mountd: 192.168.1.10, 192.168.1.100, 172.26.0.0/24
rquotad: 192.168.1.10, 192.168.1.100, 172.26.0.0/24
statd: 192.168.1.10, 192.168.1.100, 172.26.0.0/24
----------[cut here]----------

4. Arrancando los servicios

Para arrancar los servicios de nuestro servidor NFS debemos ejecutar simplemente los 'rpc' (Remote Procedure Call) que activan los demonios del portmapper, asi como el mount y el demonio de nfs. Simplemente debemos ejecutar:

/usr/sbin/exportfs -r
/usr/sbin/rpc.quotad
/usr/sbin/rpc.lockd
/usr/sbin/rpc.statd
/sbin/rpc.portmap
/usr/sbin/rpc.mountd
/usr/sbin/rpc.nfsd

NOTA: Aunque podemos hacerlo a mano, cada distribución suele tener una aplicación o script que lo lanza y para correctamente, por ejemplo en Slackware tenemos **/etc/rc.d/rc.nfsd start** y **/etc/rc.d/rc.nfsd stop**.

Para comprobar que nuestro servidor de NFS esta funcionando correctamente, basta con hacer un listado de los procesos (ps -aux, por ejemplo) y ver si realmente se han lanzado bien los demonios :-)

Si posteriormente queremos realizar cambios en nuestro **/etc/exports**, deberemos ejecutar **/usr/sbin/exportfs -ra** para obligar al demonio de NFS a re-leer la tabla de comparticiones al re-arrancarse. Una vez hecho esto, re-arrancamos el demonio de NFS, podemos hacerlo con las opciones **-HUP** del comando **kill** o **killall**.

5. El cliente

Para ejecutar un cliente de NFS simplemente debemos comprovar que nuestra version de **mount** es superior a la **2.10m** si queremos usar NFS version 3 (sí, lo queremos usar). aparte de tener soporte en la configuración de nuestro kernel para montar particiones NFS.
Para comenzar a usar una máquina como cliente de NFS solamente deberemos tener funcionando el portmapper, el statd y el lockd, por lo que podemos arrancarlos igual que habiamos hecho antes en la parte del servidor:

/sbin/rpc.portmap
/usr/sbin/rpc.lockd
/usr/sbin/rpc.statd

Una vez funcionando los demonios necesarios, podemos montar las particiones usando el siguiente comando:
bash# mount ttyp0.mine.nu:/music /home/music

y para desmontarlas, simplemente:
bash# umount /home/music

Para especificar al cliente que monte las particiones al arrancar, podemos editar el fichero **/etc/fstab** y añadirle una linea como la siguiente:

```
--------------------------[ cut here ]--------------------------
# device                        mount point     fs-type         options     dump        fsckord
(...)
server.haxor.net:/home/hacker  /home/hacker     nfs                         rw          0           0
(...)
--------------------------[ cut here ]--------------------------
```