

---

# Unidad 9-a: Ficheros en red: NFS

**VI Curso de Introducción a la Administración de  
Servidores GNU/Linux  
Extensión Universitaria. Universidad de Sevilla  
Marzo 2009**

**por Enrique Ostúa**

## Contenidos

---

1. Introducción: Sistemas de ficheros en red
2. Conceptos: RPC, portmap, NFS
3. Instalación de NFS
4. Configuración
  - /etc/exports y exportfs
  - Opciones de exportación
  - Control de Acceso, tcp\_wrapper

## 1. Sistemas de ficheros en red

---

- Los sistemas de ficheros en red permiten trabajar con directorios y ficheros de forma remota, como si fueran locales al cliente.
- El acceso a la información se encuentra centralizado en el servidor, lo que facilita el mantenimiento y la gestión de la seguridad.
- Resulta muy importante el control de accesos a la información y los permisos de los ficheros y directorios, sobre todo para trabajar “en grupos”.
- Algunos son: NFS y Samba (Red de Windows)

## 2. Como se trabaja con NFS

---

- El servidor NFS exporta unos directorios y los clientes los pueden montar por nfs y trabajar con ellos como si fueran locales
- No hay autenticación, solo control por IP
- Los cambios se realizarán en el servidor de forma automática y se reflejan en ambos sitios
- Pero el uid/gid del usuario local y remoto pueden no coincidir, por lo que los permisos son muy importantes para poder trabajar bien
- Además el root del cliente, por defecto, no tiene permisos de root en el directorio montado

## RPC y portmap

---

- *RPC: Remote Procedure Call*
- RPC es un protocolo por el que se pueden ejecutar llamadas a determinadas funciones de forma remota (a otros equipos), sin preocuparse de la forma de comunicación
- RPC fue diseñado por Sun y muy utilizado en sistemas UNIX 'clásicos'
- Los servicios RPC utilizan unos demonios propios para establecer la comunicación
- Estos demonios, al lanzarse, se “registran” en portmap

## El servidor NFS

---

- NFS es un servicio que trabaja a través de RPC
- Los demonios que se usan en el servidor NFS:
  - rpc.mountd: controla el montaje de directorios
  - rpc.nfsd: controla la exportación de archivos
  - rpc.lockd: controla el bloqueo de ficheros
  - rpc.statd: recuperación de ficheros en caso de fallo
- Ellos se registran en el demonio 'portmap' local
- El cliente NFS se conectará al servicio portmap para pedirle el acceso a los demonios de NFS

## 3. Instalación de NFS

---

- Hay dos versiones del servidor:
  - nfs-kernel-server: necesita que el kernel tenga instalado el soporte para nfs (p.e. por módulos)
  - nfs-user-server: no usa ningún recurso del kernel, tiene un rendimiento algo menor.
- Ambos funcionan de forma similar
- En ubuntu podemos usar cualquiera, así que nos decidimos por instalar nfs-kernel-server
- Script de arranque /etc/init.d/nfs-kernel-server (y /etc/init.d/nfs-common)

## 4. Configuración

---

- La lista de directorios exportados se almacena en /etc/exports, con un formato sencillo:
  - directorio1 cliente1(opcs) cliente2(opcs) ...
  - directorio2 cliente3(opcs) cliente4(opcs) ...
- Los “clienteX” pueden ser una IP o nombre de máquina, o IP/mask o \*.midominio.com
- Las “opcs” son opciones, separadas por comas y aplicadas a cada “clienteX”.
- Al cambiar algo se ejecutará:  

```
# exportfs -a
```

## Opciones de exports

---

- rw o ro: exporta lectura-escritura o sólo lectura
- async o sync: escrituras asíncronas o síncronas
- secure o insecure: sólo root puede montar en el cliente (o cualquiera, el insecure)
- mountpoint=path: sólo exporta si path es un punto de montaje y está montado [en el server]
- root\_squash y no\_root\_squash: mapea (o no) el usuario root como usuario nobody en el servidor
- all\_squash y no\_all\_squash: mapea (o no) todos los usuarios como usuario nobody en el servidor
- anonuid=65534 y anongid=65534: el uid/gid del usuario anónimo en el servidor

## Ejemplos de /etc/exports

---

```
/                master(rw) trusty(rw,no_root_squash)
/projects        proj*.micro.aulas(rw)
/usr            *.micro.aulas(ro) 10.1.15.120(rw)
# mapea el acceso al HOME de JOE de "cualquier usuario" a JOE
/home/joe       pc001(rw,all_squash,anonuid=150,anongid=100)
# carpeta de acceso publico, solo-lectura
/pub           (ro)
```

## Puesta en marcha

---

- Portmap viene en ubuntu configurado para escuchar sólo peticiones en el loopback, así que comentamos en /etc/default/portmap la línea “OPTIONS='-i 127.0.0.1'” (o mejor cambiamos la IP por una “más segura”)

```
$ ps ax|grep portmap
15134 ?    Ss   0:00 /sbin/portmap -i 127.0.0.1
$ sudo vi /etc/default/portmap
[....]
$ sudo /etc/init.d/portmap restart
$ ps ax|grep portmap
16095 ?    Ss   0:00 /sbin/portmap -i 10.1.15.8
```

## Comprobando el servidor

---

- Al instalar el paquete, no había configuración y por tanto el demonio NFS no se lanzó. Hay que arrancarlo tras la configuración.
- Para consultar los servicios RPC registrados, usamos el comando rpcinfo:
  - **rpcinfo -p**: lista los servicios RPC disponibles
    - portmapper, nfs, nlockmgr, mountd y status
- Para ver los directorios exportados o montados por los clientes, tenemos showmount:
  - **# showmount -e** (disponibles en el servidor)
  - **# showmount** (montados por clientes)

## Desde el cliente

---

- El cliente debe tener instalado el portmap
- El cliente montará la carpeta como cualquier otro punto de montaje, con la opción “-t nfs”
  - # mount -t nfs  
servidor.micro aulas:/shared /opt/shared
  - Para dejar permanente, añadir al /etc/fstab (con tipo nfs)
- Cuidado con el mapeo de usuarios!

## Control de acceso en NFS

---

- Para el control de acceso se usa tcp\_wrapper
- TCP\_wrapper es una librería de control de acceso genérica para servidores. Está disponible:
  - como librería, para compilar los servidores con soporte para tcp\_wrapper
  - con un comando adicional, tcpd, que controla el acceso y posteriormente invoca al servidor (generalmente se usa/usaba desde inetd)
- Los demonios de NFS en debian/ubuntu vienen compilados con soporte tcp\_wrapper

## Configuración tcp\_wrapper

---

- Se usan los ficheros /etc/hosts.allow y .deny
- El proceso de control de acceso es:
  - si está en hosts.allow se permite el acceso
  - si no, se busca en hosts.deny y se rechaza el acceso
  - si no está en ninguno, se permite el acceso!
- Las reglas en estos ficheros son de la forma:
  - demonio/s: cliente/s
- Hay comodines: como ALL (demonio o cliente) y \* (para clientes)

## Configuración tcp\_wrapper

---

- Algunas reglas válidas en hosts.allow|.deny:
  - ALL: ALL
  - ALL: 10.1.15.120 .miempresa.com
  - portmap: 10.1.15.\*
  - statd: localhost
- La opción mas segura es poner “ALL: ALL” en hosts.deny y ahora abrir servicio a servicio en hosts.allow, de esta forma:
  - portmap: 10.1.15.\*
  - nfsd: 10.1.15.\*
  - mountd: 10.1.15.\*