

Seguridad en: NFS Y Samba

Jesús Alberto Ramírez Viera.

Índice

- Seguridad en NFS
- Seguridad en Samba

NFS

Network File system

Seguridad en NFS

- El NFS (Network File system) es un servicio RPC (Remote Procedure Call) que se usa en conjunción con el portmap y otros servicios relacionados para proveer un sistema de ficheros en red para clientes.
- NFS controla quien puede montar y exportar sistemas de ficheros basados en la máquina que lo pide, no en el usuario que utilizará el sistema de ficheros. Las máquinas tienen que tener los derechos para montar los sistemas de ficheros exportados explícitamente

Seguridad en NFS:

Consideraciones previas.

- Antes de pasar a mirar los métodos que podemos usar para asegurar el NFS sería recomendable tener en cuenta los siguientes aspectos:
 - Debemos asegurar el servicio portmap (demonio de asignación dinámica de puertos) mediante:
 - TCP wrappers (limitar que redes o hosts tienen acceso al portmap)
 - Iptables (El servidor restringe las redes)

Seguridad en NFS:

Consideraciones previas.

- Cuidado con los errores sintácticos. El servidor NFS determina que sistemas de archivos se exportan en el fichero `/etc/exports`, un error en éste fichero puede provocar un grave agujero de seguridad.

Ejemplo (en `/etc/exports`) :

La línea siguiente exporta el sistema de ficheros del directorio `/tmp/nfs/` al host `aso.example.com` con permiso de lectura y escritura.

```
/tmp/nfs/          aso.ejemplo.com(rw)
```

Seguridad en NFS:

Consideraciones previas.

En cambio la línea siguiente:

```
/tmp/nfs/          aso.ejemplo.com (rw)
```

Exporta el directorio a aso.example.com como lectura (por defecto) y al resto (todos) lo exporta como lectura y escritura. (Tan solo por el espacio en blanco antes de "(rw)").

Podemos comprobar los sistemas compartidos con:

```
# showmount -e <hostname>
```

- Planificar la red con cuidado → NFS envía la información sin encriptar, por lo tanto sería recomendable ejecutar NFS tras un firewall o en una red segmentada y segura.

Seguridad en NFS:

Consideraciones previas.

- No usar la opción “`no_root_squash`” → Por defecto el NFS cambia los archivos compartidos pertenecientes al root al usuario `nfsnobody`, de esta forma prevenimos que puedan subir archivos con el bit `setuid` activo (podrían ejecutar programas como si fueran el root del sistema).
- Los comodines o metacaracteres deben ser usados lo menos posible cuando garantizamos el acceso a una compartición NFS. El uso de los comodines puede permitir el acceso a sistemas que puede no saber que existen y que no deberían montar el sistema de ficheros.

Seguridad en NFS:

Seguridad en el Montaje

- Una vez que el sistema de ficheros es montado como lectura-escritura por una máquina remota, la protección de cada fichero compartida depende de sus permisos, y del ID de su usuario y grupo propietario. Si dos usuarios que comparten el mismo valor ID montan el mismo sistema de ficheros NFS, serán capaces de modificarse los ficheros entre sí. Además, cualquiera que esté conectado como root en el sistema cliente, puede usar el comando **su** para convertirse en un usuario que tenga acceso a determinados archivos a través de la compartición NFS.

Seguridad en NFS:

Seguridad en el Montaje

- El procedimiento predeterminado cuando exportamos un sistema de ficheros a través de NFS es usar *root squashing* (sobreponerse a root). Esto cambia el ID de usuario de cualquiera que utilice la compartición NFS, aunque sea el root de su máquina local, al valor de la cuenta nobody del servidor. Nunca debe desactivarlo a menos que no le importe que haya múltiples usuarios con acceso de root en su servidor.
- Si sólo está permitiendo a los usuarios que lean archivos de su compartición NFS, debería considerar usar la opción *all_squash*, la cual hace que todos los usuarios que accedan a su sistema de ficheros exportado tomen la ID del usuario nobody.

Seguridad en NFS

- Podemos utilizar dos métodos para hacer más seguro un servidor NFS:
- Limitar el acceso a nuestra máquina mediante:
 - /etc/hosts.allow
 - /etc/hosts.deny
 - /etc/exports
 - TCP wrappers (tcpd usado en /etc/inetd.conf)
- Usar iptables (firewall).

Seguridad en NFS

/etc/hosts.allow

- En este archivo especificaremos que hosts van a poder usar los servicios:
 - Portmap (demonio de asignación dinámica de puertos).
 - Mountd (demonio para montar sistemas de ficheros remotos o locales).
 - Rquotad (Demonio para gestión de cuotas remotas).

Seguridad en NFS

/etc/hosts.allow

- Ejemplo → En /etc/hosts.allow

portmap:192.168.0.0/255.255.255.0

mountd:192.168.0.0/255.255.255.0

rquotad:192.168.0.0/255.255.255.0

Seguridad en NFS

/etc/hosts.deny

- En este archivo especificaremos que hosts no van a poder usar los servicios:
 - Portmap (demonio de asignación dinámica de puertos).
 - Mountd (demonio para montar sistemas de ficheros remotos o locales).
 - Rquotad (Demonio para gestión de cuotas remotas).
- Lo más recomendable es denegar el acceso por completo.

Seguridad en NFS

/etc/hosts.deny

- Ejemplo → En /etc/hosts.deny

portmap:ALL //Nadie puede acceder al portmap

mountd:ALL //No deja que monten particiones

rquotad:ALL // No da información de cuotas.

Seguridad en NFS

/etc/exports

- La estructura del fichero exports es bastante simple:
 - Directorio que se quiere exportar
 - Cliente (utilizar siempre IP's, los nombres de hosts se pueden falsear)
 - Opciones.

NOTA: El cliente puede tener una única IP (10.0.0.1), nombre de host (aso.com), una subred (10.0.0.0/255.255.255.0), o un metacaracter (*.aso.com).

- Algunas de las directivas más interesantes (y útiles) del fichero de configuración son:
 - secure → la sesión nfs se debe originar desde un puerto privilegiado, es decir, el root **TIENE** que ser el que esté intentando montar el directorio. Esto es útil si el servidor que se está exportando también está asegurado.
 - ro → Sólo Lectura, solo permite que el cliente lea.
 - noaccess → Utilizado para cortar el acceso, p. ej. exportar¹⁶ /home/ pero poner como noaccess el /home/root

Seguridad en NFS

/etc/exports

- `root_squash` → limita el UID del root a la UID/GID del usuario anónimo (normalmente "nobody"), muy útil si se están exportando los directorios a servidores de administradores en los que no se confía al 100% (el root casi siempre puede leer cualquier fichero "OJO → incluido el `/etc/passwd`")
- `no_root_squash` → Útil si se quiere perder el tiempo en directorios exportados como root para arreglar cosas (como los permisos del site `www`). *Se recomienda no usarla.*
- `squash_uids` y `squash_gids` → limitar ciertos UID(s) o GID(s) a los del usuario anónimo, en Red Hat un buen ejemplo sería 500-10000 (por defecto, Red Hat comienza a añadir usuarios y grupos en el 500), permitiendo a cualquier usuario con UID's más bajas (p. ej. cuentas especiales) acceder a cosas en especial.
- `all_squash` → todos los privilegios se eliminan y todo el mundo es guest (invitado). *Es recomendable usarlo.*
- `anonuid` y `anongid` – configuran específicamente el UID / GID del usuario anónimo (quizás se quiera algo especial como "anonnfs").

Seguridad en NFS

Introducción a IPTABLES.

- Iptables permite configurar un firewall de forma que tengamos controlado quien entra, sale y/o enruta a través de nuestro servidor.
- Es una aplicación en línea de comandos que gestiona el filtrado de paquetes en sistemas linux (kernels 2.4.x), en base a las reglas que hayamos definido. Iptables es mucho más potente que su antecesor Ipchains (kernels 2.2.x).

Seguridad en NFS

Introducción a IPTABLES.

- La estructura de Iptables es básicamente una cola: cuando un paquete llega, éste es validado contra cada una de las reglas del firewall, en el momento que alguna regla coincide (match), se ejecuta la acción que haya sido definida en la regla (descartar el paquete, aceptarlo, enrutarlo, etc.).

Seguridad en NFS

Introducción a IPTABLES.

■ ESTRUCTURA DE IPTABLES

- #iptables -t [tabla] -[AIRD L F Z N X P]
[regla] [criterio] -j [accion]
- -t → Tabla en la que se quiere añadir la regla. Hay tres:
 - NAT: enmascarar conexiones, realizar redirecciones de puertos, etc.
 - Filter: Tabla donde se añaden las reglas relacionadas con el filtrado.
 - Mangler: Podemos modificar cualquier aspecto del paquete (flags, TTL, etc).

Seguridad en NFS

Introducción a IPTABLES.

- - [AIRDLFZNX] [regla]:
 - A → Añade una regla al final.
 - I → Inserta una regla en el orden especificado.
 - R → Reemplaza una regla.
 - D → Borra una regla.
 - L → Lista una regla.
 - F → Borra todas las reglas o las reglas asociadas a una clase.
 - P → Política por defecto (aceptar todas las conexiones)

Seguridad en NFS

Introducción a IPTABLES.

- - [criterio] : Especifica las características del tipo de paquete que casará con una regla. Opciones:
 - s → ip/red fuente.
 - d → ip/red destino.
 - sport → puerto fuente. (también --source-port)
 - dport → puerto destino. (también --destination-port)
 - p → protocolo.
 - i [!] [nombre] → nombre de la interfaz por la que se recibe el paquete. El "!" indica la negación.
 - o [!] [nombre] → nombre de la interfaz por la que se envía el paquete. El "!" indica la negación.

Seguridad en NFS

Introducción a IPTABLES.

- -j [accion] → Establece lo que hay que hacer en/con el paquete:
 - ACCEPT → Aceptará el paquete.
 - REJECT o DROP → Rechazará el paquete.
 - REDIRECT → Redirecciona el paquete.
 - LOG → Logueará el paquete para analizarlo después.
- Ejemplo de la estructura de Iptables:

```
# iptables -A FORWARD -p [protocolo] -s [ip/red fuente] --  
sport [puerto fuente] -d [ip/red destino] --dport [puerto  
destino] -j DROP
```

Seguridad en NFS

Seguridad con IPTABLES.

- Hay que establecer reglas, para controlar el flujo de los paquetes, del tipo:
 - iptables -t filter -A input -p tcp -s 192.168.200.1 -d 192.168.200.5 --destination-port sunrpc -j ACCEPT
 - iptables -t filter -A input -p udp -s 192.168.200.1 -d 192.168.200.5 --destination-port sunrpc -j ACCEPT
 - iptables -t filter -A input -p udp -s 192.168.200.1 -d 192.168.200.5 --destination-port 2049:2050 -j ACCEPT
 - iptables -t filter -A input -i! lo -d 192.168.200.5 --destination-port sunrpc -j DROP
 - iptables -t filter -A input -i! lo -d 192.168.200.5 --destination-port 2049:2050 -j DROP

Samba



Introducción

- ¿Qué es Samba?
 - Es un sistema de compartición de archivos e impresoras en red.
 - Permite la interconexión de sistemas heterogéneos entre sí (Linux y Windows).
 - Los clientes windows tendrán la sensación de estar ante un servidor Windows NT.
 - Controlar el acceso de clientes Windows a servicios de red Windows o Unix.

Introducción

- ¿Qué protocolo usa?
 - SMB (Server Message Block) → Compartir los recursos.
 - CIFS (Common Internet File System) → Implementación mejorada de internet.
 - NetBIOS (Network Basic Input/Output System) → Servicio de nombres:
 - Nombres lógicos en la red.
 - Sesiones entre los nombres.

Introducción

- ¿Cuándo es útil?
 - No quieres pagar un servidor Windows NT para obtener las funcionalidades que este proporciona.
 - Homogeneizar la red local ante clientes Windows y Unix.
 - Compartir impresora entre clientes Windows y Unix.

Configuración

- Utiliza dos demonios:
 - Smbd → Permite la compartición de archivos e impresoras sobre una red SMB y proporciona **autenticación y autorización de acceso** para clientes SMB.
 - Nmbd → Se ocupa de anunciar servicios, es decir, informa a las máquinas en la red de cuales son los servicios disponibles.
- Podemos configurar mediante:
 - El fichero smb.conf
 - El front-end → SWAT

Configuración `smb.conf`

■ Smb.conf

- Se puede encontrar en `/etc/` o en `/usr/local/samba/lib/`
- Este archivo determina que recursos del sistemas vas a compartir y que restricciones deseas poner en ellos.
- Consta de varias secciones distintas que empiezan por `[nombre-recurso]`. Tiene una sección general y común → `[global]`
- Iniciar y parar con: `/etc/init.d/samba {start/stop}`

Configuración `smb.conf`

■ Secciones:

- `[global]` → Define variables de carácter general y aplicables a todos los recursos.
- `[homes]` → Permite a los usuarios remotos acceder a su directorio personal desde su máquina local (Ya sean clientes Windows o Linux), pero han de tener cuenta en la máquina servidora.
- `[printers]` → Para compartir impresoras.

Configuración smb.conf

Ejemplo:

[global]

```
workgroup = grupo8
server string = Samba Server
; hosts allow = 192.168.1. 192.168.2. 127.
printcap name = /etc/printcap
load printers = yes
log file = /var/log/samba/log.%m
max log size = 50
encrypt passwords = yes
smb passwd file = /etc/smbpasswd
```

[DATGRUP]

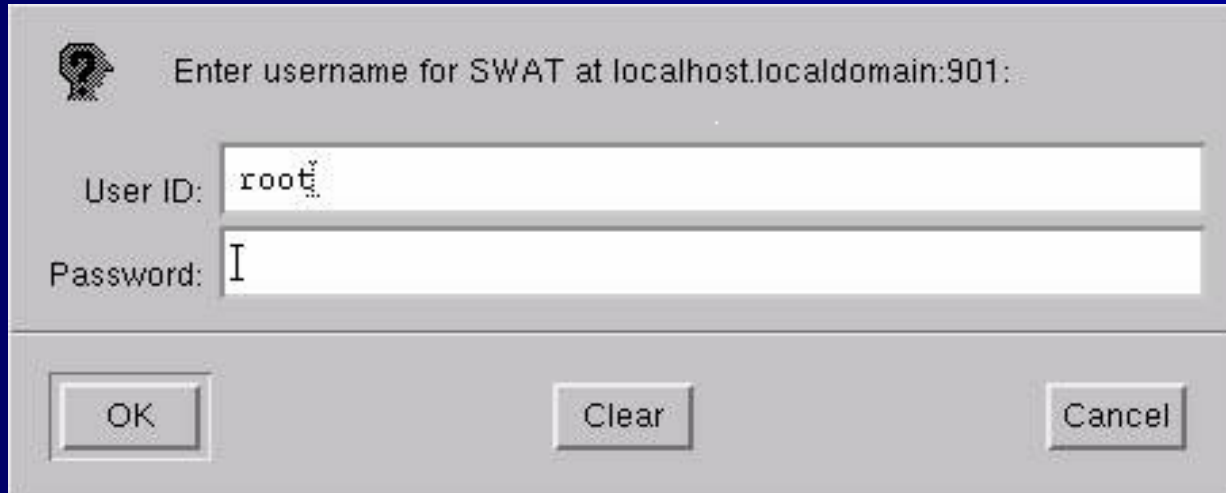
```
comment = espacio para grupo red8
path = /usr/local/datgrup
writable = yes
printable = no
public = no
valid users = @red8
```


Configuración SWAT

- SWAT (Samba Web Administration Tool).
- Es una aplicación que puede acceder desde cualquier explorador Web desde cualquier máquina de su red local.
- Utiliza el puerto 901.

Configuración SWAT

- acceda con el navegador web, a la dirección IP del servidor SAMBA en el puerto 901.
 - por ejemplo, a <http://localhost.localdomain:901>
- Acto seguido se le pedirá un login y una contraseña. Ponga los de *root para configurar*.



Enter username for SWAT at localhost.localdomain:901:

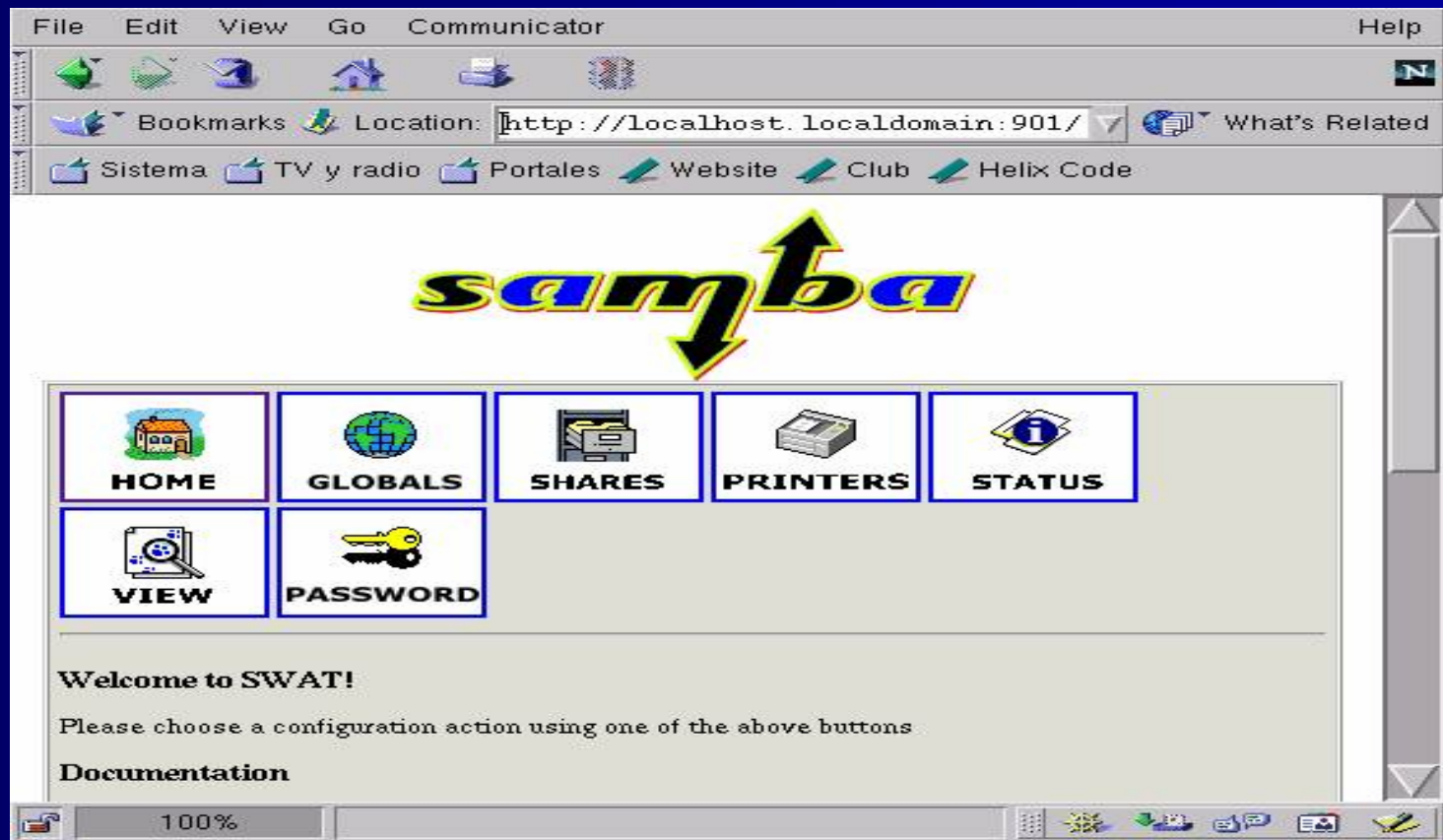
User ID:

Password:

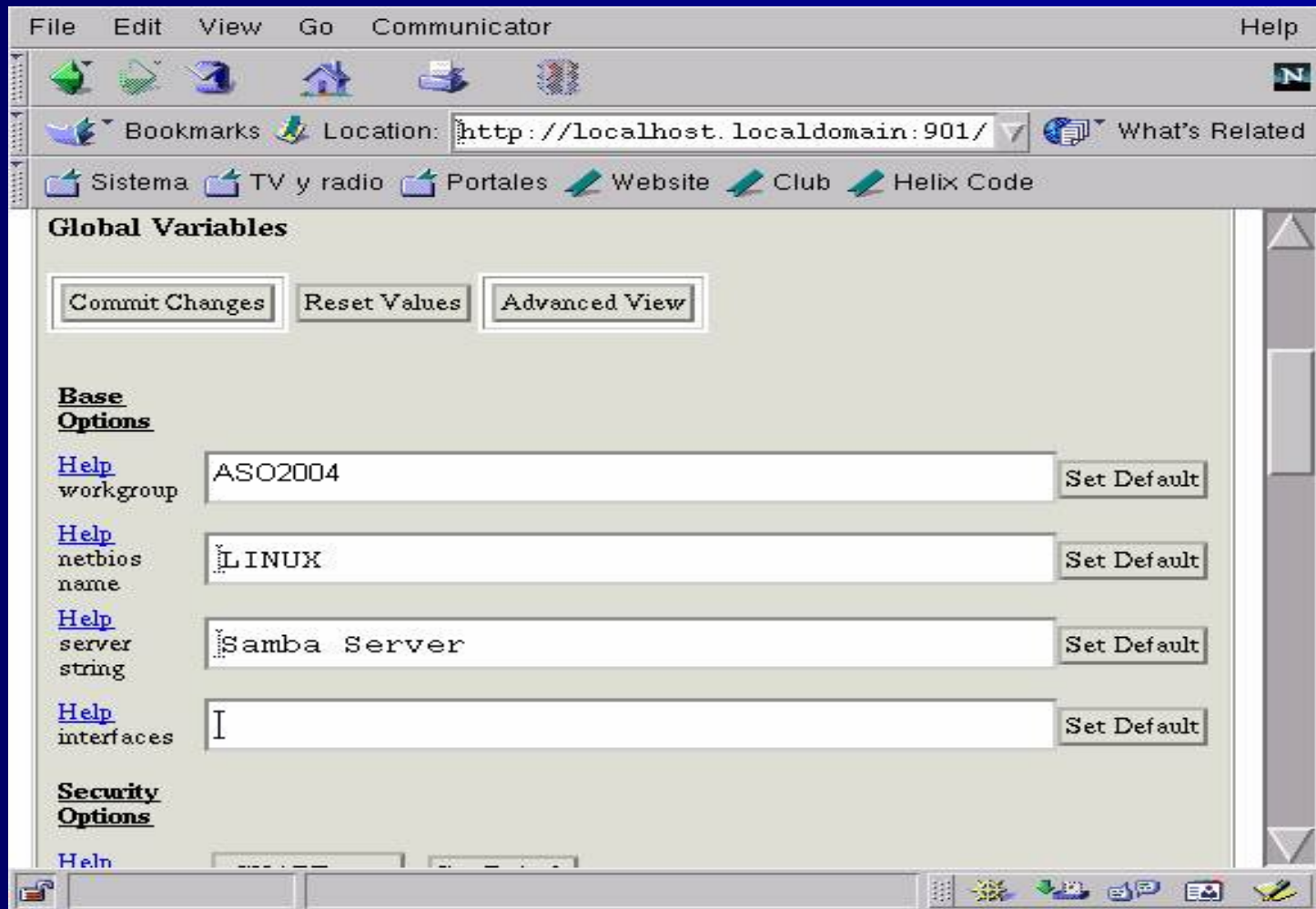
OK Clear Cancel

Configuración SWAT

- Tan solo quedará rellenar cada uno de los campos (muy intuitivo).



Configuración SWAT



Ciente de Samba

- En Windows → Es transparente para el usuario.
- En Linux → usaremos el `smbclient`:
 - Podemos buscar recursos en un host:
`#smbclient -L host`
 - Conexión a un recurso:
`#smbclient servicio <password>`
 - Servicio → Máquina y servicio.
 - Password → Es un literal con la clave
 - El resultado es un prompt (estilo ftp).

Seguridad en Samba

- Actualmente Samba utiliza cuatro niveles de seguridad:
 - Seguridad a nivel de recursos (share) → Cada recurso compartido tiene asociada una o varias contraseñas y cualquiera que las conozca puede acceder al recurso.
 - Seguridad a nivel de usuario (user) → Cada recurso se configura para permitir el acceso a determinados usuarios.

Seguridad en Samba

- Seguridad a nivel de servidor (server) → Igual que a nivel de usuario, pero en este caso el servidor Samba utiliza otro servidor SMB para validar los usuarios y contraseñas antes de conceder el acceso.
- Seguridad a nivel de dominio (domain) → En este caso Samba se convierte en un miembro de un dominio Windows y se valida primero al usuario en el dominio para luego darle un atributo especial que le da el acceso a todos los recursos a los que tenga derecho.

Seguridad en Samba

- Contraseñas → Pueden ser encriptadas o no. Por defecto no se encriptan, para compatibilidad con sistemas windows tales como Win95 o NT 3.x.

- Más seguridad si están encriptadas. Para ello añadimos a la sección global:

```
[global]
```

```
security = user
```

```
encrypt passwords = yes
```

```
smb passwd file =
```

```
/usr/local/samba/private/smbpasswd
```


Seguridad en Samba

- El fichero `smbpasswd` :
 - Estructura parecida al `passwd`
 - Hay que guardarlo muy bien y no permitir a los usuarios ni siquiera su lectura.
 - Para añadir usuarios:
 - `#smbpasswd -a usuario`
 - Para añadir o cambiar la clave:
 - `#smbpasswd usuario`
 - NOTA: El usuario ya debe existir en `/etc/passwd`.

Seguridad en Samba

- Podemos usar Iptables para filtrar los paquetes con Samba, usando los puertos de Samba (137:139).
- Ejemplos:
 - #Iptables -t filter -A INPUT -p tcp
-s 192.168.200.1 -d 192.168.200.5
--destination-port 137:139 -j ACCEPT
 - #Iptables -t filter -A INPUT -i ! lo
-s 192.168.200.1 -d 192.168.200.5
--destination-port 137:139 -j DROP

Seguridad en Samba

Vulnerabilidades

- Las versiones de Samba anteriores a la 3.0.7 son susceptibles a dos ataques remotos de denegación de servicio (DoS).
 - Un bug en el código del servidor "smbd" permite que un usuario remoto, sin necesidad de autenticarse, lance un número de instancias ilimitado, provocando un consumo de recursos que puede ocasionar la propia caída de la máquina servidor.
 - Un bug en el código del servidor "nmbd" permite que un usuario remoto mate el servicio, impidiendo el correcto funcionamiento del sistema Samba.

Seguridad en Samba

Vulnerabilidades

- (17-11-2004) e-Matters comunicó que las versiones 3.0.x de Samba, incluyendo la 3.0.7, tienen un desbordamiento de buffer que podría permitir a un atacante remoto ejecutar código en el sistema. El desbordamiento se da cuando reciben un paquete de solicitud especialmente diseñado. El equipo de desarrolladores de Samba indicaron que esta falla no está presente en la versión 3.0.8, disponible desde la semana pasada.

Samba Vs NFS

- NFS está limitado prácticamente a máquinas Unix, ya que no existe una buena implementación en Windows.
- NFS presenta problemas de seguridad
→ Solo para LAN's bien definidas y bien seguras.
- NFS no permite compartir impresoras.
- Samba se presenta como una buena solución a todos estos problemas.

Bibliografía

- Manual de referencia de Red Hat Linux.
- Red Hat Linux Security Guide.
- SAMS Linux Máxima seguridad. Anónimo. Editorial Prentice Hall.
- Man
- <http://www.europe.redhat.com>
- www.samba.org
- Diversos foros sobre vulnerabilidades.
- **Guía de Seguridad del Administrador de Linux**
Por Kurt Seifried -Traducción al español de José Antonio Revilla.
<http://www.tau.org.ar/base/computacion/gsal-19991128-htm/default.htm>