Creación de un entorno de trabajo homogéneo con Amd y NIS

Administración de Sistemas y Servicios Telemáticos

Ramón Fernández Marina <r.fernadez@ieee.org>
Antonio M. Fumero Reverón <antonio@ieeesb.etsit.upm.es>

Resumen

A continuación se describen dos herramientas presentes en los sistemas operativos tipo UNIX, Amd y NIS, y cómo pueden combinarse para crear un entorno de trabajo homogéneo en una red departamental. Esta memoria ha sido elaborada para la asignatura de *Administración de Sistemas y Servicios Telemáticos*.

Para conseguir un entorno de trabajo homogéneo en una red departamental, es preciso que el acceso a todos sus recursos se hagan de manera uniforme, idéntica, independientemente del terminal de red al que se conecte un usuario. Para ello, es preciso:

1. Que el acceso a los sistemas de ficheros donde los usuarios almacenan su información sea transparente
2. Que los usuarios puedan utilizar el mismo nombre de usuario y la misma palabra de paso

y todo ello con independencia de la máquina que deseen utilizar.

El primer requisito se consigue con Amd, una herramienta que hace que las rutas de acceso de los ficheros sean independientes del dispositivo físico donde estos están almacenados, pudiendo dichos dispositivos ser cambiados o replicados sin que el usuario tenga necesidad de saberlo.

El segundo requisito se consigue con NIS (y su mejoras y variantes). NIS es una herramienta que mantiene unas bases de datos de nombres de usuario y contraseñas que todas las máquinas de una red pueden consultar en cualquier momento.

La combinación de Amd y NIS hace posible que un usuario “se siente” en cualquier máquina de la red, teclee su nombre y su contraseña e inmediatamente tenga acceso a sus ficheros. Esto es una ventaja para el usuario, por la movilidad que le proporciona, y también para el administrador, el cual aísla la complejidad de la red a los usuarios, y puede configurarla a su antojo sin que aquellos lo perciban

Parte I
AMD

1 Objetivos y descripción del servicio

AMD son las siglas de “Auto Mounter Daemon”. Como su nombre indica, se trata de un demonio (proceso residente del sistema), cuya función consiste en montar sistemas de ficheros bajo demanda, y desmontarlos automáticamente cuando ya no sean necesarios. Es una herramienta tremendamente útil para la gestión y configuración de sistemas de ficheros.

El interés fundamental de Amd consiste en su habilidad para separar los nombres utilizados para acceder los ficheros de su localización físisca. Esto permite que el acceso a los ficheros en una red se haga de manera uniforme, independientemente del puesto de la red que utilicemos para dicho acceso.

Amd ve una red como un conjunto de servidores de ficheros, cada uno de los cuales contiene uno o más sistemas de ficheros, los cuales a su vez contienen uno o más volúmenes, entendiendo por volumen un conjunto de ficheros autocontenido (como la cuenta de un usuario, por ejemplo). Los nombres de los volúmenes han de ser únicos en toda la red, para evitar conflictos. Los volúmenes pueden replicarse (contienen la misma información y puede usarse cualquiera de ellos) o duplicarse (se dice que dos volúmenes están duplicados cuando tiene el mismo nombre y la misma información, pero distintos ficheros).

El volumen a montar se referencia mediante el conjunto `(máquina, sistema de ficheros, enlace)`, lo que hace posible referenciar cualquier volumen a través de la red. Cuando intentamos acceder a un fichero localizado en uno de esos volúmenes, Amd comprueba si dicho volumen está montado, y lo monta si no es así, creando un enlace a su situación física, para que el acceso se haga siempre de manera uniforme. Los volúmenes se montan utilizando NFS.

Amd ayuda a resolver el problema de la heterogeneidad en redes corporativas o departamentales. El montar los sistemas de ficheros con NFS simplemente no es una solución muy atractiva, porque el número de sistemas montados suele ser demasiado alto, causando problemas de tráfico, ralentizando las máquinas si un servidor se cae. Además, el mantenimiento de la configuración es prácticamente imposible, y es la pesadilla de cualquier administrador. Con Amd, todo el esfuerzo se realiza al principio, ya que el mantenimiento no es para nada costoso.

2 Requisitos

Amd está portado a multitud de arquitecturas (arm, ibm6000, mips, macII, sun, vax, i386, etc.) y funciona con numerosos sistemas operativos (irix, hpux, next, sunos, linux, etc.). Por tanto, sólo caben destacar los requisitos que han de cumplir estos sistemas operativos corriendo sobre aquellas arquitecturas.

Como se puede deducir de la sección anterior, el primer requisito de Amd es que las máquinas donde vaya a utilizarse implementen el protocolo NFS (los servidores) y soporten su sistema de ficheros (los clientes). Además, los servidores han de soportar los tipos de los sistemas de ficheros que vayan a montar localmente (ext2, msdos, vfs, hpfs, etc.). Por descontado, es necesario que las máquinas involucradas sean capaces de comunicarse a través de una red.

3 Funcionamiento del servicio

Amd introduce nuvos puntos de montaje en el espacio de nombres del *kernel*, el cual ve estos puntos de montaje como si fuesen servidores de NFS (operados por Amd). A partir de ese momento, Amd es capaz de controlar las operaciones del usuario sobre esos puntos de montaje, procesando las RPCs (*Remote Procedure Call*, llamada a procedimiento remoto) que van dirigidas a ellos. Amd no implementa todas las primitivas de NFS, sino sólo las necesarias para asociar nombres de volúmen a sistemas de ficheros (*lookup*, *readlink* y *readdir*).

Cuando se recibe una RPC para acceder a un fichero de un volúmen, Amd comprueba si el volúmen es accesible, montándolo si no lo és. Como resultado de esta operación (sea o no necesario montar el volúmen), se devuleve un enlace simbólico al volúmen recién montado, de tal manera que el *kernel* es capaz de redirigir las siguientes peticiones sobre ese volúmen al sistema de ficheros. Si el nombre del volúmen ya era conocido por Amd y el enlace simbólico ya existía, Amd no hace nada.

Si el volúmen requerido no está montado, Amd consulta sus mapas de configuración y determina dinámicamente qué sistema de ficheros tiene que montar y en qué punto de montaje, de tal forma que el volúmen sea accesible. El aspecto y la configuración de estos ficheros se explica en la sección 4.

Si el volúmen al que se intenta acceder está almacenado físicamente en un servidor remoto, el servidor Amd local se comunica con el servidor Amd de la máquina remota, donde tiene lugar el procedimiento descrito anteriormente. Una vez montado el sistema de ficheros en la máquina remota, la máquina local procede a montarlo por NFS. Se consigue así la independencia de la localización física de los datos en un entorno de red.

Como se menciona más arriba, utilizar simplemente sistemas de ficheros montados por NFS hace que el número de ellos montados sea demasiado grande, lo cual causa muchos problemas de rendimiento de la red. Para evitar esto, Amd elimina las referencias a los volúmenes que no se han usado durante un periodo de tiempo, manteniendo sólo las estrictamente necesarias. Cuando se elimina la última referencia a un volumen, el sistema de ficheros que lo contiene se desmonta. Esto se hace asociando un tiempo de vida a cada referencia a un volumen en el momento de crear dicha referencia (en realidad, esto se fija en los ficheros de configuración). Si no es posible desmontar un sistema de ficheros por alguna razón, el tiempo de vida simplemente se renueva.

Otro de los problemas de NFS es la caída de un servidor de ficheros remoto. Si esto sucede, los procesos locales que están accediendo a dicho servidor no son capaces de darse cuenta, por lo que antes o temprano acabarán "colgándose". Para paliar este comportamiento, Amd consulta periódicamente los servidores de ficheros, para asegurarse de que funcionan correctamente. Si un servidor deja de funcionar, Amd seguirá intentando contactar con él hasta que lo consiga, y si se produce alguna petición de un proceso de usuario mientras esto sucede, Amd devuelve mensajes de error a dicho proceso para evitar que se cuelgue. Si cuando un servidor deja de estar accesible Amd conoce la existencia de uno replicado, lo montará y se podrá seguir trabajando normalmente.

4 Configuración

En el sistema operativo *Debian GNU/Linux*, los ficheros de configuración están ubicados en el directorio `/etc/amd`, y el script encargado de iniciar o detener el servicio en `/etc/init.d/amd`. El fichero principal de configuración de Amd se llama `amd.master`, que contiene las opciones con las que se ejecutará `amd`. Su aspecto es el siguiente:

```
-a /amd
/import         /etc/amd/amd.import
/home           /etc/amd/amd.home
```

La primera línea especifica que los sistemas de ficheros remotos se montarán bajo el directorio `/amd`, que será necesario crear si no existe. Las dos siguientes líneas indican a `amd` que a través del directorio `/import` se accederá a los volúmenes especificados en el *mapa* `/etc/amd/amd.import`, y lo mismo para `/home`.

Los mapas de Amd son los ficheros de configuración que asocian los volúmenes con su localización física. Esta información tiene que existir en alguna parte, porque Amd no hace magia. Sin embargo, la sintaxis de estos mapas es tan potente que es posible crearlos de tal forma que *todas las máquinas de la red puedan compartir el mismo mapa*. Una vez que esto es posible, la administración de Amd es sencillísima, porque basta con mantener un único mapa que se distribuye a todas las máquinas de la red cuando es actualizado (es necesario modificar otros ficheros también, no sólo el mapa, pero la información importante está en los mapas, que son los que le dan versatilidad y potencia a Amd).

4.1 Un servidor de ficheros

Si queremos que una máquina (linux, por ejemplo) actúe como servidor de ficheros de Amd, necesitamos el software necesario (el paquete Amd) y un editor de texto.

La primera tarea es determinar qué sistemas de ficheros serán accesibles desde el exterior. Una vez hecho esto, es necesario que puedan ser montados, para lo cual habrá que modificar el fichero `/etc/fstab`. Si, por ejemplo, queremos que el sistema de ficheros que contiene la partición `/dev/hdb1` sea "visible" a través de Amd, hemos de añadir la siguiente línea al `fstab`:

```
/dev/hdb1        /export/dsk1    ext2    defaults     0   0
```

Estamos suponiendo que el sistema es de tipo *ext2*, el sistema de ficheros de linux, y que vamos a montar todos aquellos sistemas que exportemos por Amd bajo el directorio /tt /export, el cual debe existir previamente, por supuesto.

El siguiente fichero a modificar no pertenece a Amd, sino al servidor de ficheros NFS, y es el `/etc/exports`. En él, deberemos poner una línea como la siguiente:

```
/export *.acme(rw)
```

suponiendo que vayamos a permitir el acceso a los sistemas montados bajo `/export` a todas las máquinas del dominio `acme`, y que además vayamos a permitir que lean y escriban en dichos sistemas de ficheros. Para más información, consultar la página de manual (`man exports`).

Una vez hecho esto, hemos de configurar el servicio Amd. El fichero `amd.master` será de la forma:

```
-a /amd
/import         /etc/amd/amd.import
```

suponiendo que vayamos a acceder a todos los sistemas de ficheros visibles con Amd, ya sean locales o remotos, a través del directorio `/import`. El fichero `amd.import` será como sigue:

```
# Sistema de ficheros
dsk1            host==filemon;type:=link;fs:=/export/dsk1 || \
                host!=filemon;rhost:=filemon;rfs:=/export/dsk1

# Volúmenes
vol1            type:=link;fs:=/import/dsk1/vol1
vol2            type:=link;fs:=/import/dsk1/vol2
```

En este ejemplo, `filemon` es el nombre del servidor, que exporta el sistema de ficheros que se monta bajo `/export/dsk1` y que contiene los volúmenes `vol1` y `vol2`. Cuando accedamos al directorio `/import/vol1`, el servidor montará el sistema `dsk1` en su punto de montaje especificado en el fichero `fstab`, y creará los enlaces simbólicos:

- `/import/vol1` → `/import/dsk1/vol1`
- `/import/dsk1` → `/export/dsk1`

por lo que accederemos a `vol1` a través de su enlace en el directori `import`.

En la parte del mapa correspondiente al sistema de ficheros, se puede ver cómo se codifica la localización física de dichos sistemas, quedando entonces oculta al usuario (como quedará claro cuando se configuren los clientes).

4.2 Un cliente de Amd

Como ya se ha mencionado, el los mapas de Amd, si se configuran adecuadamente, pueden ser compartidos tanto por clientes como por servidores. Por tanto, después de instalar el software necesario en la máquina cliente, tan sólo es necesario copiar los ficheros de configuración del servidor para que podamos acceder a los volúmenes del sistema de ficheros `dsk1`, `vol1` y `vol2`, a través del directorio `/import` del cliente.

Cabe destacar, aunque se acaba de insinuar, que los directorios `/import`, `/amd` y `/export` han de existir en todas las máquinas, si bien este último no es necesario en aquellas máquinas que no actúen como servidores. También es obvio que una máquina puede ser cliente y servidor al mismo tiempo.

4.3 Cuentas de usuario y otras facilidades

Es muy común en un entorno departamental la necesidad de los usuarios de acceder a sus ficheros y a su correo personal desde cualquier máquina a la que se conecten. Para ello, no sólo han de poder usar el mismo nombre de usuario y la misma contraseña en todas las máquinas (tal y como se verá en la parte II), sino que la ruta de acceso a esos ficheros y a ese correo ha de ser siempre la misma. Esto es muy fácil de hacer con Amd.

Generalmente, a las cuentas de usuario se accede a través del directorio `/home`. Si queremos hacer esto con Amd, lo primero que debemos hacer es decirle a qué volúmenes vamos a acceder a través de dicho directorio, por lo que es precido añadir la línea

```
/home /etc/amd/amd.home
```

al fichero `amd.master`. Después, hemos de crear el fichero `amd.home`, el cual contendrá un listado de los volúmenes accesibles a través de `/home`. Suponiendo que el servidor de cuentas de usuario es `mortadelo`, el cual las exporta montándolas en el directorio `/export/dsk2`, es preciso añadir lo siguiente a `amd.master`:

```
dsk2            host==mortadelo;type:=link;fs:=/export/dsk2 || \
                host!=mortadelo;rhost:=mortadelo;rfs:=/export/dsk2

home            type:=link;fs:=/import/dsk2/home
```

y el fichero `amd.home` será de la forma:

```
/defaults type:=nfs;opts:=rw,grpid,nosuid

*               type:=link;fs:=/import/home/\${key}
```

En este ejemplo se considera cada cuenta de usuario como un volúmen *per se*. Cuando un usuario accede al sistema, Amd realiza los siguientes pasos:

1. El usuario intenta entrar en su cuenta, `/home/usuario`, por lo que Amd recibe una petición de acceso a un volúmen a través del directorio `/home`.

2. Como se especifica en la configuración, Amd ha de consultar el mapa `amd.home`, el cual le indica que para acceder a `/home/usuario` ha de hacerlo a través de `/import/home/usuario`, por medio de un enlace simbólico.

   (a) Al intentar acceder a un volúmen a través de `/import`, Amd consulta el mapa `amd.import`, donde se especifica que para acceder al volúmen `home` a través de `/import`, es necesario montar el sistema de ficheros `dsk2`.

   (b) Tal y como se indica en el mismo mapa, el sistema de ficheros requerido se encuentra en `mortadelo`. Si el usuario se está conectando al servidor directamente, el sistema de ficheros se montará en `/export/dsk2`.

   Si por el contrario el usuario no está conectándose directamente al servidor (sino a `ofelia`, por ejemplo), se generará una petición hacia este para que monte físicamente el sistema de ficheros, acción tras la cual `ofelia` montará por NFS el directorio de `mortadelo` `/export/dsk2` en `/amd/import/mortadelo/export/dsk2`.

   Una vez concluídos estos pasos, es posible acceder, por medio de enlaces simbólicos que crea Amd, al sistema de ficheros que contiene las cuentas de usuario a través de `/import/home`.

3. Ahora que Amd sabe cómo acceder a los volúmenes bajo `/import/home`, también sabe cómo acceder a los que se encuentran bajo `/home`, ya que son exactamente los mismos. Es entonces cuando es posible acceder a la cuenta del usuario, a través de los siguientes enlaces:

   (a) `/home/usuario` → `/import/home/usuario`
   (b) `/import/home` → `/import/dsk2/home`
   (c) `/import/dsk2` → `/export/dsk2`

   si el usuario se ha conectado a `mortadelo`, y estos otros si se ha conectado a otra máquina (`ofelia`):

   (a) `/home/usuario` → `/import/home/usuario`
   (b) `/import/home` → `/import/dsk2/home`
   (c) `/import/dsk2` → `/amd/mortadelo/export/dsk2`

5 Pruebas

Las pruebas a realizar para verificar el correcto funcionamiento de Amd son bien sencillas, porque basta con comprobar que podemos acceder a los volúmenes especificados.

El muy útil, como casi siempre, tener siempre visible los mensajes que genera el sistema a través de la herramienta `syslog`, tanto en el cliente como en el servidor, porque pueden ayudarnos a saber en caso de no poder acceder a algún volúmen si el problema reside en la configuración de Amd (sintaxis de los ficheros) o bien en la configuración de otros servicios, como el servidor de nombres (DNS) o el de NFS.

6 Gestión diaria

La gestión del servicio comprende dos acciones:

- Añadir un sistema de ficheros con sus correspondientes volúmenes a Amd, para que pueda ser visto desde toda la red.
- Eliminar uno volúmen o un sistema de ficheros.

En ambos casos es preciso distribuir el mapa modificado a todas las máquinas, bien para que tengan acceso a los nuevos volúmenes, bien para que no se produzcan errores al intentar acceder a un volúmen que ya no existe.

Utilizando la configuración presentada, en la que todos los accesos se hacen a través del directorio `/import` antes o después, el mapa más importante es el `amd.import`, porque en él están configurados todos los sistemas de ficheros y todos los volúmenes. El caso de acceso a las cuentas de usuario mostrado antes sirve para poder referenciar los volúmenes por medio de otras rutas, porque al final siempre se accede a ellos a través de `/import`, por medio de enlaces simbólicos. Además, en el caso de añadir volúmenes en `amd.home`, la línea

```
*                type:=link;fs:=/import/home/\${key}
```

siempre ha de ser la última del fichero.

La pequeña desventaja de esta configuración es que, a la hora de eliminar un volúmen, hay que hacerlo de dos ficheros de mapas, pero es insignificante comparada con la ventaja de poder utilizar los mismos mapas en todas las máquinas, ya que sólo es necesario configurar una de ellas (preferentemente el servidor) para que, una vez distribuidos los mapas, todas ellas puedan ver el mismo entorno de red homogéneo, al menos en cuando a sistemas de ficheros se refiele.

Parte II
NIS

7 Introducción

Introducción En este documento se pretende recoger una breve descripción de los servicios NIS y NIS+, sus características principales, arquitecturas y objetivos. Así mismo se incluyen unas secciones dedicadas a la instalación y configuración del NIS en Debian y pruebas del servicio. La información se ha extraído básicamente a partir de la Web personal de Thorsten Kukuk en `http://www.suse.de/~kukuk/`

8 Qué es y para qué sirve

El NIS (Network Information System) es un sistema que proporciona un servicio simple de búsqueda redes más o menos tochas. En principio se llamó YP (Yellow Pages); pero como era un nombre registrado como marca comercial por British Telecom, se cambió por el de NIS. La idea es que nos ayude a construir un sistema distribuido aparentemente consistente, de forma transparente para el usuario. Se trata de construir una base de datos de ficheros de configuración, gestionada por un servidor que atenderá las peticiones de los clientes; de forma que éstos ya no tengan que usar los ficheros locales.

Un caso común, que suele servir de ejemplo es el fichero de contraseñas en /etc/passwd. Lo que se hace es construir un mapa (se llama así) de contraseñas a partir del fichero original en el servidor NIS maestro, que se difunde a los servidores esclavos. En un determinado dominio, tendremos un servidor maestro para cada uno de los mapas, y los servidores esclavos que consideremos oportunos. Todos los clientes NIS les preguntarán a esos servidores por la contraseña de usuario. Una vez que NIS esté operativo, ya no necesitarás editar el fichero de contraseñas local; bastará con que actualices el fichero en el servidor maestro para que este lo difunda a los esclavos.

¿Qué ventajas ofrece esto a tus usuarios? Pues que ya no tendrán que preocuparse de tener cuentas distintas en máquinas diferentes, de las contraseñas de cada una de ellas, o de si podrá hacer login en tal o cual máquina de la red. Además no necesitará cambiar su contraseña en cada cliente de forma individual.

La idea es que si lo combinamos con un sistema de ficheros distribuido (como puede ser NFS) nos facilite la gestión de usuarios en redes con muchas máquinas.

9 Servidores NIS

Los mapas YP [guaypí] están formados por dos campos: la clave y el valor. El servidor NIS no sabe nada de los contenidos del mapa; en principio, para él es lo mismo un mapa de contraseñas que uno de máquinas o de grupos: son sólo conjuntos de parejas clave-valor, es decir datos y las claves para acceder a ellos; de forma que puedes meter en ellos lo que te parezca mejor, puedes incluso usarlos como directorio telefónico. Sólo el cliente debe ser capaz de interpretar los datos. El servidor YP sólo busca ciertas claves que contienen información acerca de el servidor maestro y la marca de tiempo que se le puso en su creación.

En casos como el de los mapas de contraseñas o de grupos en que se podría, en principio, realizar la búsqueda usando como clave tanto el UserID como el GroupID, lo que se hace es crear un mapa para cada clave, puesto que el NIS sólo permite usar una clave: tendremos unos mapas tal que así, password.byname, password.byuid, group.byname, group.bygid. Esta forma de hacer las cosas significa que tendremos que transferir los datos dos veces cada vez que cambiemos algo.

Hay mapas en que ese apaño chapucero no funcionaría, como por ejemplo el caso del fichero de servicios, donde tenemos el nombre del servicio, el número de puerto y el protocolo. Te podrías crear un mapa con el nombre del servicio como clave, pero y qué pasa si estás buscando un servicio del que sólo sabes el número de puerto y el tipo de protocolo? Pues que te tendrá que patear todo el mapa a pelo: hay funciones para eso, como yp_first() e yp_next(), o yp_all()

Lo que hay que tener claro es que necesitas tres cosas para identificar una entrada de un mapa YP: el nombre de dominio, el nombre del mapa y la clave: el servidor irá a /var/yp/nombre_dominio, abrirá nombre_mapa, y buscará por "clave". De esta forma el servidor podrá dar servicio a tantos dominios como directorios puedas crear en /var/yp/ ; y para inicializar un dominio nuevo sólo necesitarás crear el directorio correspondiente, meterte en él, y ejecutar make del Makefile del directorio padre.

10 El protocolo YP

Es el protocolo de comunicaciones que se establece entre el servidor YP (ypserv) y un cliente. Hay dos versiones oficiales. La primera versión es muy vieja y ya no se usa. Se conocen las llamadas de funciones RPC, pero no está claro qué es lo que hace cada función; aun así hay versiones del ypbind de Sun que sólo usan ese protocolo para encontrar al servidor YP adecuado. Para solventar esto, hay implementaciones de ypserv que registran las funciones de la versión 2 como de versión 1 (como los parámetros de las funciones que se usan son los mismos, la cosa va bien; a no ser que te pongas a jugar con software muy viejo que use la versión 1). La otra solución tonta es implementar las funciones de la versión 1. En las nuevas versiones de Solaris, Sun ha extendido la versión 2 con algunas funciones, que no están documentadas. Para el ypbind hay un montón de versiones de protocolo, pero sólo la 2 está bien documentada.

En la versión 2 del protocolo NIS hay 11 funciones RPC para el servidor:

- YPPROC_DOMAIN, YPPROC_DOMAIN_NONACK son para encontrar un servidor YP en un determinado dominio. Para la primera, el sevidor devolverá siempre TRUE o FALSE. Para la segunda, se devolverá TRUE sólo si servimos a ese dominio; de lo contrario, no retornaremos nunca.
- YPPROC_MATCH, YPPROC_FIRST, YPPROC_NEXT, YPPROC_ALL se usan para obtener entradas de un mapa. Todas usan UDP, salvo YPPROC_ALL que te devuelve el mapa entero a través de una conexión TCP
- YPPROC_XFR, YPPROC_CLEAR, YPPROC_MASTER, YPPRO_ORDER, YPPROC_MAPLIST. La primera sirve para hacer un yppush: le dice al servidor esclavo que un mapa ha cambiado en el maestro y que necesita actualizarlo. La segunda la usa el servidor para borrar todas la caches y los manejadores de ficheros. Es una función que se llama después de actualizar un mapa, por ejemplo ejecutando /usr/lib/yp/makedbm. Las otras tres se usan para obtener información especial del mapa.

11 El protocolo NIS

Para que esto de las páginas amarillas funcione, necesitaremos dos demonios: el ypserv en un servidor YP, y el ypbind en el cliente.

Si un cliente necesita una entrada de un mapa YP, lo primero que hace es buscar un servidor YP que esté activo: para eso abrirá una conexión TCP con un proceso ypbind local y le mandará el nombre de dominio al que pertenece el proceso cliente. El ypbind hace un broadcast a la función YPPROC_DOMAIN_NONACK, con lo cual todos los servidores YP que den servicio al dominio cuyo nombre se ha difundido, contestarán con un ACK, mientras que el resto no responderá. Ypbind devuelve el status de la búsqueda y la dirección del primero de los servidores que contestó al proceso cliente. Con esa información ya podremos hacer una petición enviando el nombre de dominio, nombre de mapa y clave; a continuación, ypserv devolverá el resultado de la petición y, si tiene éxito, también los datos.

Es un protocolo lento, y necesita muchos sockets. Para solventarlo, lo que se hace es que ypbind mantiene una lista de servidores activos, que guardará en

```
/var/yp/binding/nombre\_dominio.versión\_protocolo\_YP
```

para ahorrarse la conexión TCP entre libc y el ypbind, a los cuales sondea en cortos intervalos de tiempo para comprobar que siguen con vida.

12 Problemas de seguridad

YP se diseñó para alrededor de 100 clientes y unos pocos servidores, en una época en que las máquinas no estaban muy separadas unas de otras y no había muchos requisitos de seguridad. Hay muchas cosas que han cambiado desde entonces. Uno de los problemas de YP es la necesidad de tener un mapa para cada clave: en una base de datos medio grande tendremos un montón de información redundante. Si se hace algún cambio, por pequeño que sea, hay que transferir el mapa completo a los esclavos; y si tenemos más de un mapa del mismo tipo para diferentes claves, deberemos transferirlos todos. Esto implica tiempo y carga de tráfico en la red: no hay forma de transmitir sólo la entrada que ha cambiado en el mapa.

Puesto que no había requisitos de seguridad, cualquiera podría ver el contenido de las tablas, e iniciar un ataque de fuerza bruta contra el mapa de contraseñas, por ejemplo. Se supone que para evitar ese tipo de ataques chorras se usan las contraseñas en la sombra, pero como YP no sabe nada de autenticación de usuarios, ese mecanismo no se pudo implementar. Una idea sería usar el hecho de que sólo las peticiones de cliente que se hicieran desde un puerto cuyo número fuera menor de 1024 podrían ver la contraseña. Pero los puertos privilegiados no están disponibles en todos los sistemas (en DOS por ejemplo).

Otro problema es que YP no cifra los datos que manda por la red: con lo cual, cualquiera que instale un sniffer podría ver las contraseñas.

Para atacar un servidor YP, necesitas encontrar un servidor activo y conocer el nombre de dominio. Para encontrar un servidor activo, podrías hacer un broadcast del RPC proc 0 (zero). Las implementaciones actuales de portmap no reenvían ese tipo de peticiones difundidas hacia los servidores YP por razones de seguridad. El problema es que mucha gente las usa para comprobar el funcionamiento de su sistema.

También existe la posibilidad de que cualquiera te instale un servidor falso: una máquina que se dedique a responder a las peticiones que difunde el ypbind, identificándose como servidor de dominio. Este es el motivo por el que ypbind no debería usar broadcast en ningún caso: de hecho la versión Linux de ypbind lee una lista de servidores YP del fichero de configuración /etc/yp.config, y sólo usa esos servidores.

13 NIS+

Es el sucesor de YP, y se supone que viene a resolver gran parte de sus problemas. La primera diferencia es que tiene un espacio de nombre jerárquico, como el DNS, al contrario que YP que usaba un dominio plano.

En NIS+ tendremos un servidor maestro en cada dominio o subdominio. Además podríamos tener servidor de réplicas: cada servidor maestro de un subdominio deberá ser un servidor de réplicas del dominio del que cuelga (tendrá su importancia cuando busquemos tablas en subdominios).

Cada usuario deberá seguir un proceso de autenticación en el servidor NIS+ usando RPC seguro: la hora actual se cifra con la clave secreta del usuario; el servidor la descifra con la clave pública de ese usuario y comprueba que la hora esté en una pequeña ventana de su propia hora. Para que esto funcione es importante que todos los servidores y clientes se sincronicen regularmente, usando xntpd por ejemplo.

Los datos se almacenan en tablas. Cada columna tiene una serie de flags, que indican si se puede usar como clave de búsqueda, si está cifrada, si distingue mayúsculas y minúsculas o si está almacenada en formato binario o en XDR. Esto quiere decir que podemos buscar por más de una clave en cualquier tabla o que podemos combinar varias claves en una consulta.

Podríamos poner derechos de acceso a los directorios o a las tablas, o incluso a cada columna y cada entrada de una columna. Es decir que podríamos hacer que un usuario viera sólo su contraseña en la tabla de contraseñas, aunque pueda ver al resto de los usuarios.

Hay un campo de TTL que le dice al cliente cuánto tiempo puede tener en su cache un objeto, tabla o entrada de una tabla.

Otra ventaja es que en NIS+ sólo se transfieren los cambios en las tablas al servidor de réplicas. Esto también tiene su cruz, y es que el servidor de réplicas puede perder el sincronismo con el servidor maestro.

14 Seguridad en NIS+

La autenticación se hace con SecureRPC. Los derechos de acceso a un objeto (tablas o directorios), una columna o una entrada se dividen en cuatro partes: Nobody, Owner, Group y World. Funcionan igual que los derechos de acceso de los sistemas de ficheros Unix de toda la vida, sólo que tienen una clase más.

La cuenta de root es un usuario normal. Sin los permisos adecuados no podrá leer las contraseñas cifradas de la tabla de contraseñas. Puede hacer un su a otro usuario, pero no puede autentificarse como ese usuario.

En NIS+ no se mandan contraseñas en claro por la red (nispasswd protocol). Aunque es la excepción, puesto que el resto de datos no van cifrados.

Un servidor NIS+ puede funcionar con tres niveles de seguridad diferentes:

0. Sin autenticación: se usa para pruebas y depuración, y jamás debería usarse en un entorno en producción.
1. Autenticación Unix: El cliente enviará el uid y el gid; no es seguro.
2. Autenticación DES: se usa SecureRPC

Si fuera necesario se podría poner a funcionar a un servidor NIS+ como servidor YP; pero habría que ponerlo en un modo inseguro, y cualquiera podría ver tus tablas NIS+. Es decir, que no sería una idea en absoluto afortunada.

El servidor NIS+ tiene 24 funciones RPC que, aparte de dos de ellas que no están documentadas, se dividen en tres grupos:

Funciones del servicio de nombresè NIS_LOOKUP, NIS_ADD, NIS_MODIFY, NIS_REMOVE sirven para manejar tablas.

Funciones para el manejo de la base de informaciónè NIS_IBLIST, NIS_IBADD, NIS_IBMODIFY, NIS_IBREMOVE, NIS_IBFIRST, NIS_IBNEXT sirven para acceder a las entradas de las tablas.

Funciones Administrativasè NIS_FINDDIRECTORY, NIS_STATUS, NIS_DUMPLOG, NIS_DUMP, NIS_CALLBACK, NIS_CPTIME, NIS_CHECKPOINT, NIS_PING, NIS_SERVSTATE, NIS_MKDIR, NIS_RMDIR, NIS_UPDKEYS se usan para la comunicación entre los servidores de réplicas, y para mantener la jerarquía de directorios.

15 Protocolo NIS+

Lo primero es extraer el nombre de dominio de la consulta. Si conocemos un servidor para ese dominio, miramos en el COLD_START_FILE de NIS+. Si no es así, tenemos dos posibilidades:

Si estamos en un dominio padre del dominio que estamos buscando, contactaremos con todos los servidores de nuestro dominio; uno de ellos será el servidor maestro del subdominio, puesto que el servidor maestro de un subdominio debe ser un servidor de réplicas para el dominio padre.

Si estamos en un subdominio del dominio que estamos buscando, contactaremos con nuestro servidor maestro, que será una réplica del dominio padre.

Repetiremos ese proceso hasta dar con el dominio correcto.

Luego cogemos la lista de todos los servidores para ese objeto, y mandamos lo que se llama un ping difusivo: usaremos el servidor que responda primero.

16 NYS

NYS viene de NIS+, YP y Switch; se trata de un intento de añadir NIS+ y NSS (Name Service Switch) a Linux. Con NSS puedes especificar las fuentes de las bases de datos: los ficheros de hosts, grupos o contraseñas y el orden de búsqueda. Podrías, por ejemplo, buscar primero en el fichero de local hosts y luego en el mapa NIS por nombre del host; o podrías decirle que buscara sólo en el DNS, y que si no lo encuentra que busque en el local hosts, pero nunca en el NIS...

Hay dos módulos de NSS para NIS: el normal, que sólo busca en la tabla NIS; y el compat, que se usa para compatibilidad hacia atrás entre el glibc 2.x y el libc 5.

NYS utiliza el /etc/yp.conf en vez del ypbind para construir la lista de servidores.

17 Requisitos

Lo primero será ver si vamos a necesitar instalar un servidor maestro, un esclavo o un cliente. Para eso consideraremos dos casos:

1. Si tu máquina va a formar parte de una red donde ya hay servidores NIS, sólo necesitarás programas cliente (ypbind, ypwhich, ypcat, yppoll, ypmatch) . El más importante de todos es ypbind: se trata de un demonio, que estará siempre corriendo y deberá arrancarse en el fichero de inicio (e.g. /etc/init.d/nis, /sbin/init.d/ypclient, /etc/rc.d/init.d/ypbind, /etc/rc.local) . En cuanto ypbind esté corriendo, ya tienes un cliente NIS en tu sistema.

2. Si no tienes servidores NIS en tu red, necesitarás un programa servidor, como ypserv de Peter Eriksson y Thorsten Kukuk, o el yps de Tobias Reber.

Vamos a necesitar ejecutar el programa /usr/sbin/portmap. Algunas distribuciones de Linux ya traen el código necesario para arrancar este demonio en el fichero /sbin/init.d/ o en el /etc/rc.d/ , con lo cual sólo necesitas activarlo (quitar la línea de comentario) y rearrancar la máquina.

El portmap es un servidor que convierte números de programa de RPC en números de puerto TCP/IP (o UDP/IP). Necesitamos que el portmap esté corriendo antes de arrancar el inetd, que es el demonio que arranca los servidores RPC estándar.

Si queremos usar Secure RPC, necesitaremos que esté funcionando el servicio de sincronización de relojes (time): debe activarse en /etc/inetd.conf en todas las máquinas, quedando algo como:

```
#
# Time service is used for clock syncronization.
# time stream tcp nowait root internal

time dgram udp wait root internal
```

18 Instalación y configuración

Instalación y Configuración Lo normal será instalar NIS; a menos que tengas unos requisitos de seguridad muy severos no vale la pena, hoy por hoy, instalar NIS+: la administración, sobre todo la del servidor, es bastante más compleja.

18.1 Cómo se instala un cliente local de NIS

Lo primero es instalar los paquetes netbase, netstd y nis: en el proceso de instalación se te pedirá un nombre de dominio NIS. Sólo es un nombre que describe el grupo de sistemas que usan NIS; no se trata del nombre de una máquina. Se almacena en el fichero de configuración /etc/defaultdomain, y es bueno que lo sepas por si alguna vez lo tienes que cambiar.

Si el servidor de NIS no es local a tu red, tendrás que realizar un “ajuste fino”. El fichero de configuración del ypbind está en /etc/yp.conf.

Para arrancar el NIS no hay más que teclear: /etc/init.d/nis stop /etc/init.d/nis start

18.2 Cómo especificamos los recursos que va a usar NIS

Para libc 6: Comprueba que en el fichero de configuración /etc/nsswitch.conf aparecen las entradas passwd, group, shadow y netgroup de la forma siguiente

```
passwd: compat
group: compat
shadow: compat
netgroup: nis db files
```

Usuarios: Añade la siguiente línea en los ficheros /etc/passwd en tus clientes NIS:

```
+::::::
```

Además puedes usar los caracteres + y - para añadir, eliminar o cambiar usuarios: si quieres excluir a pepito, pones -pepito en /etc/passwd. Si lo que quieres es usar una shell distinta para el usuario juanito, vas al /etc/passwd y pones ”+juanito::::::/bin/ksh”(sin las comillas) . Debes dejar vacíos los campos que no quieras cambiar. Por ejemplo, para permitir hacer login a manolo, jorge y martin, pero manteniendo el acceso a los datos del resto de usuarios disponibles, pondrás:

```
+manolo::::::
+jorge::::::
+martin::::::
+:*:::::/etc/NoShell.
```

Esta es una de esas cosas que puedes hacer en Linux: machacar sin más el campo de contraseña.

Grupos: Añade la línea siguiente al fichero /etc/group

```
+:::
```

Hosts: Normalmente no deberías buscar máquinas con NIS; deberías usar DNS para eso. Pero si alguna vez realmente tienes que hacerlo, debes hacerlo así:

Para las aplicaciones libc5: si quieres usar los mapas de hosts de tu servidor NIS, debes cambiar tu fichero /etc/host.conf. Sólo tienes que añadir "nisª la línea ºrder". El fichero podría quedar así:

```
order host,bind,nis
multi on
```

Para las aplicaciones libc6: edita el fichero /etc/nsswitch.conf y cambia la entrada hosts:

```
hosts: nis files
```

18.3 Instalación y configuración de un servidor NIS maestro

Lo primero es instalar el paquete nis. Además necesitas los paquetes netbase y netstd instalados para los demonios RPC (rpc.portmap).

Asegúrate de que todos los sistemas que vayas a usar como servidores NIS, maestros o esclavos tienen al resto de sistemas en el fichero /etc/hosts. El FQDN (Fully Qualified Domain Name) debería ser el primer nombre de la máquina después cualquier dirección IP, seguido sólo por el nombre de la máquina (sin el nombre de dominio). Por ejemplo:

```
192.168.88.10 troi.cistron.nl troi
```

Es el aspecto que tiene en el servidor NIS el fichero de máquinas de NIS (que suele ser sólo /etc/hosts) debería tener esta configuración. Esto es así porque el NIS no usa DNS.

Hay que cambiar el NIS domain en el fichero /etc/defaultdomain. El dominio NIS es sólo un nombre para identificar a un grupo de sistemas que usan NIS, no es el nombre de una máquina. Es algo muy común encontrarte gente que usa su nombre de dominio DNS. No supone correr riesgos de seguridad, como mucha gente cree.

Pon al ypserv como maestro (ypserv=master) en /etc/init.d/nis

Añade tu número de red local al fichero /etc/ypserv.securenets. No tienes por qué hacerlo, puesto que por defecto se permite a cualquiera acceder a tu servidor NIS. Para añadir seguridad deberás también editar el fichero /etc/ypserv.conf para meter mano al fichero de contraseñas de algunos clientes (no lo hagas si no tienes servidores esclavos debian en tu red)

Puedes inicializar el servidor con "/usr/lib/yp/ypinit -m"

Arranca el servidor escribiendo

```
/etc/init.d/nis stop
/etc/init.d/nis start
```

Esto arrancará al servidor (ypserv) y el demonio de contraseñas (yppasswdd) .

Si quieres restringir el acceso a tu servidor NIS, tendrás que configurarlo como cliente, con el ypbind. Las funciones de biblioteca ignorarán todas las entradas normales después de la primera entrada NIS, y cogerá el resto de la información a través de NIS. De esta forma se mantienen las reglas de acceso NIS. Ejemplo:

```
root:x:0:0:root:/root:/bin/bash daemon:*:1:1:daemon:/usr/sbin:
bin:*:2:2:bin:/bin: sys:*:3:3:sys:/dev:
sync:*:4:100:sync:/bin:/bin/sync games:*:5:100:games:/usr/games:
man:*:6:100:man:/var/catman: lp:*:7:7:lp:/var/spool/lpd:
```

```
mail:*:8:8:mail:/var/spool/mail: news:*:9:9:news:/var/spool/news:
uucp:*:10:50:uucp:/var/spool/uucp: nobody:*:65534:65534:nada
denada,,,,:/dev/null:
+antoniof:::::: +:*:::::/etc/NoShell
[Todos losusuarios normales van después de esta línea]
tester:*:299:10:Es sólo una cuenta de prueba:/tmp:
antoniof:1234567890123:101:10:Antonio Fumero:/home/antoniof:/bin/zsh
```

Existirá un usuario tester, pero tiene una shell de /etc/NoShell. antoniof tendrá acceso normal.

Alternativamente, podrías editar el archivo /var/yp/Makefile y configurar al NIS para que use otro fichero de contraseñas fuente. En sistemas grandes, los ficheros de grupo y contraseñas están normalmente almacenados en /var/yp/ypfiles/. Si haces esto las herramientas normales de administración del fichero de contraseñas, tales como "passwd", "chfn", ªdduser"no funcionarán más y tendrás que hacerte tus propias herramientas caseras al efecto.

Sin embargo yppasswd, ypchsh e ypchfn sí que funcionarán. La versión Debian de estas utilidades (y el demonio yppasswdd) tienen una extensión no estándar. "Root"puede, usando la contraseña de root, cambiar las contraseñas, la información de búsqueda y la shell de otra gente. Si editas el fichero de contraseñas NIS manualmente o si usas el fichero estándar /etc/passwd, recuerda que después de cada cambio que hagas en los ficheros fuente NIS, tendrás que ejecutar un "make"en el directorio /var/yp para actualizar los mapas NIS. Sería razonable hacerlo periódicamente desde cron para asegurarte de que los mapas NIS están actualizados.

18.4 Contraseñas en la sombra

La libc5 para Linux no soporta mapas NIS con contraseñas en la sombra. Puedes utilizar una alternativa que se conoce como seguridad "como siütilizaras contraseñas en la sombra (SHADOW-LIKE SECURITY) Puedes proporcionar ese nivel de seguridad ªpañando"la contraseña para las búsquedas NIS de entradas de archivo de contraseñas. Puedes leerte las páginas de manual para "ypserv.conf"y los comentarios en la muestra de /etc/ypserv.conf.

Libc6 incorpora soporte shadow real para NIS. Funciona como cabría esperar; exporta el mapa shadow desde el servidor NIS, y es ese el que utiliza. El mapa shadow debería construirse con la opción s"(secure) del makedbm. Es un proceso automático en todos los ficheros /var/yp/Makefile modernos.

18.5 Instalación y configuración de un servidor NIS esclavo

Lo primero será configurar el sistema para que trabaje como cliente (tal como se vio más arriba)

Seguiremos los mismos pasos que con el maestro, sólo que pondremos ypserv=slave en el fichero /etc/init.d/nis.

Para arrancarlo e inicializarlo pondremos:

```
/etc/init.d/nis stop
/etc/init.d/nis start
/usr/lib/yp/ypinit -s
```

Ahora debes decirle a tu servidor maestro que tiene un esclavo. Primero tienes que ajustar el fichero Makefile en el maestro de forma que todas las futuras actualizaciones se distribuyan automáticamente a los esclavos. Te vas al fichero /var/yp/Makefile y pones la variable NOPUSH así:

```
NOPUSH=""
```

Ahora debes informar al maestro acerca de sus esclavos ejecutando "/usr/lib/yp/ypinit -m". Mete los nombres de los servidores esclavos. Los mapas se reconstruirán y se mandarán a los esclavos.

Podrías querer editar el crontab de root en el esclavo y añadir las siguientes líneas:

```
20 * * * * /usr/lib/yp/ypxfr\_1perhour
40 6 * * * /usr/lib/yp/ypxfr\_1perday
55 6,18 * * * /usr/lib/yp/ypxfr\_2perday
```

Esto asegurará que muchos mapas NIS se mantienen actualizados, incluso en el caso de que alguna actualización se pierda a causa de que el esclavo estuviera caído cuando se hizo la actualización en el maestro.

19 Pruebas

Para comprobar si la instalación de NIS ha ido bien, podemos usar un par de comandos muy simples (suponiendo que tenemos el fichero de contraseñas en el NIS). El comando

```
% ypcat passwd
```

debería darte los contenidos de tu fichero de contraseñas. El comando

```
% ypmatch userid passwd
```

(donde userid es el login de un usuario cualquiera) debería darte la entrada de ese usuario en el fichero de contraseñas del NIS. Los programas "ypcat"e "ypmatch"deberían venir con la distribución de NIS o NYS.

Si un usuario no puede hacer login, ejecuta el siguiente programa en el cliente:

```
#include <stdio.h> include <pwd.h> include <sys/types.h>

int main(int argc, char *argv[]) { struct passwd *pwd;

  if(argc != 2) { fprintf(stderr,"Usage: getwpnam username\n");
    exit(1); }

  pwd=getpwnam(argv[1]);

  if(pwd != NULL) { printf("name.....: [%s]\n",pwd->pw\_name);
    printf("password.: [%s]\n",pwd->pw\_passwd); printf("user id..:
    [%d]\n", pwd->pw\_uid); printf("group id.: [%d]\n",pwd->pw\_gid);
    printf("gecos....: [%s]\n",pwd->pw\_gecos); printf("directory:
    [%s]\n",pwd->pw\_dir); printf("shell....: [%s]\n",pwd->pw\_shell); }
    else fprintf(stderr,"User \"%s\" not found!\n",argv[1]);

  exit(0);
}
```

Ejecutando este programa con el nombre de usuario como parámetro podremos ver toda la información que la función getpwnam puede darnos sobre el usuario. Con esto deberías poder ver qué es lo que está pasando. El problema más común suele ser que en el campo de contraseña hay un asterisco ("*").

La biblioteca GNU C Library 2.1 (glibc 2.1) viene con una herramienta que se llama getent. Puedes usar ese programa en lugar del cacho de código de antes:

```
getent passwd o getent passwd login
```

20 Cómo se instala NIS+

Para acceder a las últimas fuentes y documentación, puedes ir `http://www.suse.de/~kukuk/linux/nisplus.html`

Después de instalar la nueva libc y las nis-tools, hay que crear las "credenciales"para los nuevos cliente y servidor de NIS+. Debes asegurarte de que portmap está corriendo. Luego hay que comprobar que tu máquina tiene la misma hora que el servidor NIS+. Tendrás una pequeña ventana de tres minutos en que las credenciales son válidas para Secure RPC. Es buena idea tener corriendo en cada máquina el xntpd. Luego ejecutamos esto:

```
% domainname nisplus.domain.
% nisinit -c -H <NIS+ server>
```

para inicializar lo que se conoce como el COLD_START_FILE. Hay más opciones que se recogen en las páginas del manual de nisinit. Luego debes cambiar el fichero /etc/nsswitch.conf. Asegúrate de que el único servicio que hay después de publickey es nisplus ("publickey:nisplus"), y nada más. Después de esto hay que arrancar keyserv y asegurarse de que siempre se arranca como el primer demonio después de portmap en la secuencia de arranque. Ejecuta keylogin -r para almacenar la clave secreta de root en el sistema.

Ahora al hacer "niscat passwd.org_dir"debería mostrarte todas las entradas de la base de datos de contraseñas.

20.1 NIS+, keylogin, login y PAM

Cuando un usuario entra en el sistema, necesitará que el servidor de claves le dé su clave secreta. Esto se hace llamando a "keylogin", que lo hará de forma transparente para el usuario, si se compiló contra la glibc 2.1. Para un login PAM tendrás que instalar pam_keylogin-1.2.tar.gz y cambiar el fichero /etc/pam.d/login para usar pam_unix_auth, en lugar de pwdb, que no soporta NIS+. Como ejemplo tenemos el siguiente:

```
#%PAM-1.0
auth     required /lib/security/pam\_securetty.so
auth     required /lib/security/pam\_keylogin.so
auth     required /lib/security/pam\_unix\_auth.so
auth     required /lib/security/pam\_nologin.so
account  required /lib/security/pam\_unix\_acct.so
password required /lib/security/pam\_unix\_passwd.so
session  required /lib/security/pam\_unix\_session.so
```

20.2 El fichero nsswitch.conf

El fichero nsswitch.conf de configuración del NSS determina el orden de búsqueda que se sigue cuando se pide algún tipo de información; lo mismo que se hacía con el /etc/host.conf para determinar cómo se buscaba a una máquina. Por ejemplo, la linea

```
hosts: files nisplus dns
```

especifica que los funciones de búsqueda de host mirarán primero en el fichero /etc/hosts, luego buscarán en NIS+, y finalmente en el DNS (/etc/resolv.conf y named), y si aun así no se ha encontrado nada, sedevolverá un error.

Un buen ejemplo de fichero de configuración para NIS+ sería:

```
# /etc/nsswitch.conf Un ejemplo del fichero de configuración del
#NSS. Este fichero debería ponerse con los servicios más usados al
#principio.  La entrada '[NOTFOUND=return]' significa que la búsqueda
#de una determinada entrada debería detenerse si la búsqueda de la
#entrada inmediatamente anterior no devolvió nada. Hay que darse
#cuenta de que si la búsqueda falló por cualquier otro motivo(como que
#el servidor NIS no contesta) , la búsqueda continuará con la
#siguiente entrada.  Las entradas que se permiten son: nisplus Utiliza
#NIS+ (NIS version 3) nis Utiliza NIS (NIS version 2), also called YP
#dns Utiliza DNS (Domain Name Service) files Utiliza los ficheros
#locales db Utiliza las bases de datos /var/db [NOTFOUND=return]
#Detiene la búsqueda si no ha encontrado nada

passwd: compat # for libc5: passwd: files nisplus group: compat # for
libc5: group: files nisplus shadow: compat # for libc5: shadow: files
nisplus

passwd\_compat: nisplus group\_compat: nisplus shadow\_compat: nisplus

hosts: nisplus files dns

services: nisplus [NOTFOUND=return] files networks: nisplus
[NOTFOUND=return] files protocols: nisplus [NOTFOUND=return] files
rpc: nisplus [NOTFOUND=return] files ethers: nisplus [NOTFOUND=return]
files netmasks: nisplus [NOTFOUND=return] files netgroup: nisplus
bootparams: nisplus [NOTFOUND=return] files publickey: nisplus
automount: files aliases: nisplus [NOTFOUND=return] files
```

Índice General