Ensalada de Openafs

o como configurar Openafs con Kerberos y OpenLDAP

Javier Monrové Morán
Proyecto integrado

Índice de contenido

Introducción a AFS:

¿Qué es AFS?

Andrew File System (AFS) es un sistema de ficheros distribuido que permite a los usuarios acceder y compartir todos los archivos almacenados en una red de servidores como si estuvieran en su propia máquina.

AFS almacena los ficheros en un grupo de máquinas servidores en la red. Éstos servidores proveen almacenamiento y entrega de los ficheros así como otros servicios para las máquinas clientes .
Los servidores AFS corren una serie de procesos y cada uno de ellos provee de un servicio especializado, uno gestiona las peticiones de archivos, otro localiza los archivos, otro maneja la seguridad...

Celda

Cada celda es independiente de las demás y tiene un administrador del sistema que es el que toma las decisiones sobre como configurar y mantener la celda de la mejor manera para sus usuarios sin tener que consultar a los administradores de otras celdas.
Se componen de una colección de máquinas servidores y clientes pertenecientes a la celda. Una maquina solo puede pertenecer a una celda en cada momento. Los usuarios tienen una cuenta en la celda pero al contrario de las máquinas pueden tener cuentas en distintas celdas.
Una celda puede ser por ejemplo en una universidad cada departamento que la conforma(aunque no importa la distancia entre las máquinas sino la velocidad de la red).

Volúmenes:

AFS agrupa los archivos en volúmenes, haciendo posible distribuir los archivos en varias máquinas y mantener un espacio uniforme. Un volumen es una unidad de espacio de disco que funciona como un contenedor para un grupo de ficheros relacionados, manteniéndolos juntos en una partición. Estos volúmenes pueden variar en tamaño pero por defecto son mas pequeños que una partición.

Usuarios:

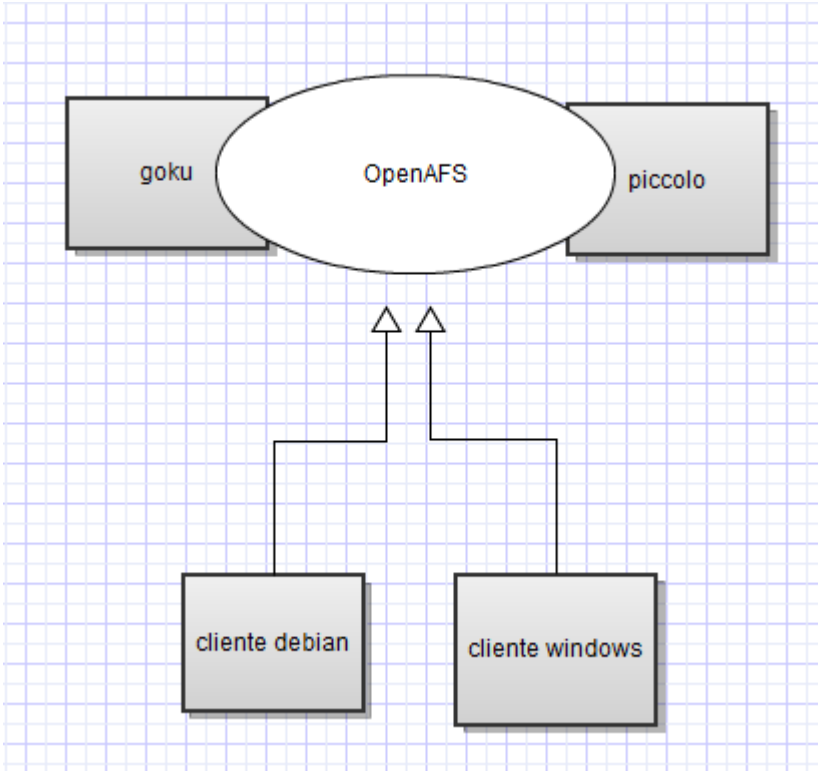
Aunque cada celda es administrativamente independiente, probablemente quieras organizar la colección local de ficheros (tu espacio de archivos o ARBOL) de manera que los usuarios de otras celdas puedan acceder a la información en ella. AFS permite que las celdas se combinen en un espacio de ficheros global y lo hace de una manera que es transparente al usuario y este no necesita saber donde están los archivos para acceder a ellos, lo único que necesitan es una ruta al archivo, que tiene la misma pinta en cada celda. Además cada usuario en cada máquina ve la colección de archivos de la misma manera proveyendo a los usuarios de una nomenclatura de espacio uniforme.

Control de acceso:

Para que todos los usuarios no tengan los mismos permisos sobre los ficheros una de las maneras de implementar esto es requiriendo que los servidores y los clientes intercambien sus identidades.
Los mismos usuarios controlan otro aspecto de la seguridad de AFS como es el accesos por otros a los ficheros y directorios de su propiedad construyendo una ACL que permite o deniega el acceso a los contenidos del directorio.

Idea Principal:

Configurar OpenAFS(con kerberos), 2 servidores 1 cliente (al menos) .
Me gustaría integrar kerberos con OpenLDAP para centralizar los directorios de los usuarios en /afs/usuarios/ de manera que los documentos de los usuarios no estén en la máquina local sino en los servidores AFS



Ya veremos que hacemos con el cliente de windows en cuanto al “home” del usuario

¿Como se ha montado esto?

El montaje se ha realizado en máquinas virtuales hechas con Virtualbox bajo Windows 7, cada una de ellas con 512 megas de ram asignados y unos 4 GB de espacio máximo de disco,
He utilizado como servidores dos máquinas con debian lenny y otra más como cliente
Otras dos máquinas windows (una windows 7 que murió durante las pruebas) y de las que queda solo la de windows xp debido a la exageración de recursos que consume un windows mas moderno.

Los paquetes usados para esta configuración han sido extraidos de los repositorios o en su defecto descargados de la página web de openafs.org,

Pasos previos:

Instalación de un servidor DNS(en goku):

En mi caso voy a configurar un servidor DNS para la zona monrove.is-a-geek.com:

```
aptitude install bind9
```

/etc/bind/named.conf.local:

```
zone "monrove.is-a-geek.com" {
        type master;
        file "/var/cache/bind/monrove.is-a-geek.com.bd";
};
zone "192.168.2.in-addr-arpa" {
        type master;
        file "/var/cache/bind/monrove.is-a-geek.com.bi";
};
```

activamos tambien el forwarder en /etc/bind/named.conf.options

busqueda directa monrove.is-a-geek.com.bd:

```
@          IN        SOA       monrove.is-a-geek.com.   admin.monrove.is-a-
geek.com. (
                          201003251       ; fecha de hoy + serie
                          8H              ; tiemp refresco
                          2H              ; tiempo retry
                          4W              ; tiempo expiraciÃ³n
                          1D )            ; mÃnimo
;
@       IN      NS goku.monrove.is-a-geek.com.

localhost       IN      A       127.0.0.1
goku    IN      A       192.168.2.150
piccolo IN      A        192.168.2.151
```

busqueda inversa goku.monrove-is-a-geek.com.bi:

```
@       IN      SOA     goku.monrove.is-a-geek.com.     admin.monrove.is-
a-geek.com. (
                          201003250
                          8H
                          2H
                          4W
                          1D )
;
        IN      NS      goku.monrove.is-a-geek.com.
150      IN     PTR     goku.monrove.is-a-geek.com.
151      IN     PTR     piccolo.monrove.is-a-geek.com.
```

Instalación de un servidor kerberos 5 (en goku)

Para instalar OpenAFS necesitaremos configurar primero Kerberos y crear nuestro reino:

```
aptitude install krb5-admin-server
```

que nos instalará por dependencias todos los paquetes necesarios.

IMPORTANTE: necesitamos que nuestro kerberos sea compatible con kerberos v4 por lo que cuando lo instalemos haremos un **dpkg-reconfigure krb5-kdc** y elegiremos compatibilidad "**full"**

Después de instalarlo hacemos **`dpkg-reconfigure krb5-config`**
y rellenamos con los datos necesarios

Editamos **/etc/krb5.conf** y configuramos nuestro reino poniendo en cada sitio los datos necesarios

En este caso mi reino sera MONROVE.IS-A-GEEK.COM

```
[libdefaults]
        default_realm = MONROVE.IS-A-GEEK.COM

[realms]
        MONROVE.IS-A-GEEK.COM = {
                admin_server = goku.monrove.is-a-geek.com
                default_domain = monrove.is-a-geek.com
                kdc = goku.monrove.is-a-geek.com
        }

[domain_realm]
        .monrove.is-a-geek.com  = MONROVE.IS-A-GEEK.COM
        monrove.is-a-geek.com = MONROVE.IS-A-GEEK.COM
```

Creamos nuestro reino con:
#krb5_newrealm

e iniciamos los servicios: **`/etc/init.d/krb5-admin-server start`**
`/etc/init.d/krb5-kdc start`

accedemos a la configuración del reino con **#kadmin.local**
y añadimos nuestros principales
en mi caso solo he añadido por ahora los necesarios para una configuración mínima:

tenemos que añadir el principal afsadmin:

addprinc afsadmin/admin@MONROVE.IS-A-GEEK.COM

Destacar en la creación de principales la creación del host afs/dominio
o también podemos utilizar como yo he hecho el principal mas simple sin incluir el dominio, cuando pidamos ticket con aklog simplemente probará primero con afs/dominio@ y después con afs@ tampoco es un problema.
Por lo tanto podemos crear cualquiera de los siguientes principales:

addprinc -randkey -e des-cbc-crc:v4 afs/monrove.is-a-geek.com@MONROVE.IS-A-GEEK.COM
addprinc -randkey -e des-cbc-crc:v4 afs@MONROVE.IS-A-GEEK.COM

Este principal es el nombre de la celda (parecido al reino kerberos),
por convención se llama a la celda en minúsculas y al reino kerberos en mayúsculas.

Cada servidor ha de tener la clave que hemos generado arriba así que podemos exportarla a un archivo:

ktadd -k /etc/krb5.keytab.afs -e des-cbc-crc:v4 afs

Actualmente afs cuenta solo con soporte para claves des-cbc-crc es por eso por lo que hay que generar el keytab de la manera anteriormente explicada

IMPORTANTE APUNTAR LA VERSION kvno QUE NOS HA DADO AL EXPORTAR LA CLAVE
nos hará falta en durante la configuración de afs.

LDAP

instalamos los paquetes que nos van a hacer falta
servidor openldap:

```
aptitude install slapd
```

autentificación con kerberos mecanismos GSSAPI:

```
aptitude install libsasl2-modules-gssapi-mit sasl2-bin
```

configuramos el servidor openldap con **dkpg-reconfigure slapd**

para la autentificación GSSAPI creamos el fichero /etc/ldap/sasl2/slapd.conf y añadimos la linea

```
mech_list: GSSAPI
```

entramos en la administración de kerberos para crear un principal de ldap y exportarlo en /etc/krb5.keytab

```
#kadmin.local

kadmin.local: addprinc -randkey ldap/goku.monrove.is-a-
geek.com@MONROVE.IS-A-GEEK.COM

kadmin.local: ktadd -k /etc/krb5.keytab ldap/goku.monrove.is-a-
geek.com
```

Ahora hacemos ese keytab legible para openldap:

```
#chgrp openldap /etc/krb5.keytab
#chmod 640 /etc/krb5.keytab
```

Para añadir entradas al LDAP voy a usar Apache Directory Studio pero podéis usar cualquier herramienta (incluso las ldap-utils del repositorio)

Realmente no es necesaria la autentificación ldap puesto que las autentificaciones se van a realizar por medio de tickets kerberos y ldap solo se utilizará para almacenar la información de los usuarios

Instalación servidor OpenAFS

Instalación de la "base"

Ahora instalamos/configuramos OpenAFS, para ello tenemos que compilar un módulo para nuestro núcleo, nos descargamos openafs-modules-source y lo compilamos con module-assistant

Después tendremos que instalar los paquetes que necesitamos de afs

```
aptitude install openafs-fileserver openafs-krb5 openafs-dbserver
```

y lo configuramos con los datos pertinentes y añadimos SOLO el servidor que estamos configurando a la lista de servidores

Por dependencias se habrá instalado **openafs-client** y tendremos que configurarlo así que ejecutamos **dpkg-reconfigure openafs-client** y elegimos que no se inicie el cliente al iniciar el ordenador, que no busque la celda en el dns, aquí para mas seguridad podemos encriptar el trafico entre los servidores o dejarlo sin encriptar, le decimos que genere dinámicamente los contenidos de /afs, y que use fakestat para que no se quede colgado cuando listamos /afs

Editamos /etc/openafs/server/ThisCell y ponemos el nombre de nuestra celda(si no está ya añadido que normalmente lo hará por si solo):

```
goku:/afs# cat /etc/openafs/server/ThisCell
monrove.is-a-geek.com
```

Editamos /etc/openafs/server/CellServDB y añadimos nuestros servidores
(mirar primero por si ya está añadido, lo mismo que el anterior)

```
goku:/afs# cat /etc/openafs/server/CellServDB
>monrove.is-a-geek.com
192.168.2.150        # goku.monrove.is-a-geek.com
```

He observado que esta dirección la toma de /etc/hosts, por lo que
si ponemos en etc hosts las ip estáticas de los servidores nos ahorramos esta modificación

ATENCIÓN # servidor.celda **ES NECESARIO y NO ES UN COMENTARIO**

// el siguiente paso puede no ser necesario:

Esta información será necesaria para los clientes así que la copiamos en nuestros clientes, para probarlo en el servidor lo copiamos en la siguiente ruta en el caso de que actúe como cliente.
ASEGURARSE ANTES DE QUE NO ESTÁ YA AÑADIDO EN LOS FICHEROS

```
goku:/afs# cp /etc/openafs/server/ThisCell /etc/openafs/ThisCell
```

```
goku:/afs# cat /etc/openafs/server/CellServDB >> /etc/openafs/CellServDB
```

En el CellServDB se encuentran las direcciones de todos los servidores AFS de cada dominio

Viendo esto podemos suponer que para añadir servidores al dominio basta con configurarlos y añadirlos en estos archivos. Lo veremos mas adelante

```
//esto SI es completamente necesario!
```

Una vez hecho esto ya podemos decirle a afs que principal de kerberos tiene que usar y que clave usar para autentificarse
Para esto tendremos que haber instalado el paquete **openafs-krb5**

```
asetkey add kvno /etc/krb5.keytab.afs afs
```

el kvno tendréis que sustituirlo por el numero de kvno que nos generó al exportar la clave de la celda afs al archivo afs.keytab.afs

Una vez llegado a este punto tendremos que empezara a gestionar nuestra celda de AFS para poder crear volúmenes y moldearla a nuestras necesidades. Para esto es necesario conocer los procesos de AFS.

Procesos Fundamentales de AFS

File server:

Es el proceso mas importante ya que entrega los ficheros de datos desde los servidores a las estaciones de trabajo cuando lo piden, y los actualiza cuando se producen cambios en los archivos.

Basic OverSeer Server(BOS):

Se asegura que el resto de procesos se ejecutan correctamente.

Kerberos:

Ayuda a que las comunicaciones en la red sean seguras.

Protection Server:

Ayuda a los usuarios a controlar quien accede a sus ficheros y directorios. Los usuarios pueden conceder acceso a otros usuarios poniéndolos en una entrada de grupo en la "Protection Database" mantenida por este Protection Server.

Volume Server:

Se encarga de la manipulación de los volúmenes y ayuda al administrador a mover volúmenes de una máquina a otra para equilibrar la carga entre máquinas.

Volume Location Server:

Mantiene la base de datos de volúmenes (Volume location database, VLDB) en la que se guarda la localización de los volúmenes.

Update Server:

Distribuye nuevas versiones e información de configuración a todos las maquinas servidor.

Backup Server:

Mantiene la base de datos de copias de seguridad. Permite al administrador hacer copias de seguridad de los volúmenes y restaurarlas.
Salvager(que no es salvaje sino "Rescatador"):
Cuando el File Server o el Volume Server fallan, este proceso repara las inconsistencias que se hayan producido.

Network Time Protocol Daemon (NTPD):

Es MUY IMPORTANTE que los servidores de hora estén sincronizados para el correcto funcionamiento de AFS

Cache Manager:

Es un set de instrucciones en las maquinas clientes que permiten la comunicación con los servicios de los servidores.

Podemos ver mas detalladamente cada una de las funciones de los servicios en: http://docs.openafs.org/AdminGuide/ch01s03.html

Creación de la Celda:

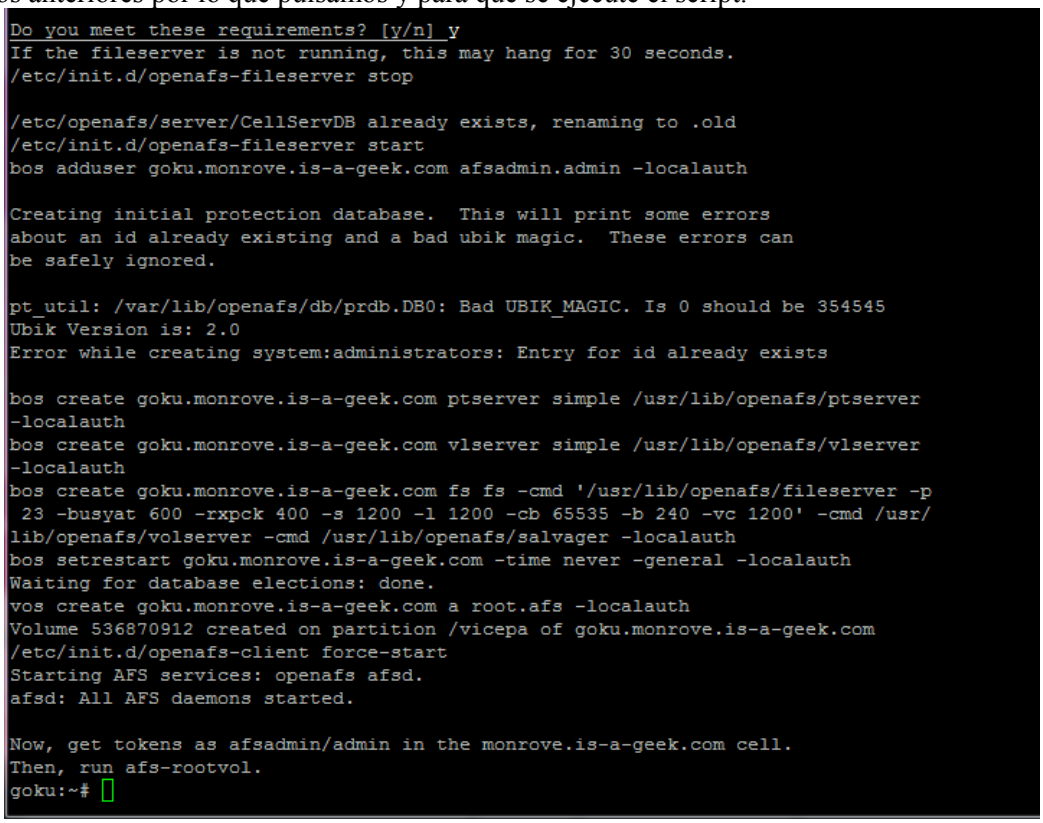
AFS-newcell

Primero crearemos una partición y la montaremos en el lugar por defecto "/vicepa" esta será la partición principal y la necesitaremos.
En mi caso he tenido que agregar un disco duro nuevo a la maquina virtual crear una partición en este , formatearlo en ex3 y después montarlo en /vicepa.
(cada servidor puede tener 256 particiones nombradas /viceXX donde XX va de 'a' a 'z' y de 'aa' a 'iv')
Ahora ejecutamos el script de creación:

```
# afs-newcell --admin afsadmin/admin
```

(le estamos pasando por parámetro el usuario administrador de la celda)

Nos soltará información sobre los PREREQUISITOS que tenemos que cumplir, que ya los hemos cumplido en los pasos anteriores por lo que pulsamos y para que se ejecute el script.



Si nos diera algún error bastará con corregirlo y volver a ejecutarlo(podemos ignorar el error de Ubik).
El error mas probable será un fallo de comunicación, probablemente sea por algún error de configuración en el dns, podemos añadir y es bastante recomendable en /etc/hosts las direcciones de los servidores para ahorrarnos el error.

AFS-rootvol

Cuando se termina de ejecutar nos pedirá que consigamos un token de la celda que hayamos configurado como afsadmin/admin y después ejecutemos **afs-rootvol**.

Para ello haremos lo siguiente:
`kinit afsadmin/admin` para el ticket kerberos
`aklog` para el ticket afs

y luego ejecutamos el script de generación del volumen raiz:
`afs-rootvol`

También nos mostrará lo prerequisitos que tenemos que cumplir, supuestamente el script anterior ya los ha completado así que seguimos adelante.

```
Do you meet these conditions? (y/n) y

You will need to select a server (hostname) and AFS partition on which to
create the root volumes.

What AFS Server should volumes be placed on? goku.monrove.is-a-geek.com
What partition? [a] a

vos create goku.monrove.is-a-geek.com a root.cell -localauth
Volume 536870915 created on partition /vicepa of goku.monrove.is-a-geek.com
fs sa /afs system:anyuser rl
fs mkm /afs/monrove.is-a-geek.com root.cell -cell monrove.is-a-geek.com -fast || true
fs mkm /afs/grand.central.org root.cell -cell grand.central.org -fast || true
fs mkm /afs/wu-wien.ac.at root.cell -cell wu-wien.ac.at -fast || true
fs mkm /afs/hephy.at root.cell -cell hephy.at -fast || true
fs mkm /afs/cgv.tugraz.at root.cell -cell cgv.tugraz.at -fast || true
fs mkm /afs/itp.tugraz.at root.cell -cell itp.tugraz.at -fast || true
fs mkm /afs/sums.math.mcgill.ca root.cell -cell sums.math.mcgill.ca -fast || true
fs mkm /afs/cern.ch root.cell -cell cern.ch -fast || true
```

Con esto habremos creado el volumen root.afs ubicado en /vicepa

si hacemos un listado en /afs podemos ver el resto de celdas y entre ellas la nuestra (en la imagen monrove no es mas que una redirección hacia la celda monrove.is-a-geek.com)

```
goku:/# ls afs/
1ts.org                     enea.it                  jpl.nasa.gov             qatar.cmu.edu
acm-csuf.org                engr.wisc.edu            kfki.hu                  rhic.bnl.gov
acm.uiuc.edu                eng.utah.edu             kloe.infn.it             riscpkg.org
ams.cern.ch                 epfl.ch                  kth.se                   rl.ac.uk
andrew.cmu.edu              es.net                   laroia.net               rose-hulman.edu
anl.gov                     ethz.ch                  lcp.nrl.navy.mil         rpi.edu
asu.edu                     extundo.com              le.infn.it               rrz.uni-koeln.de
athena.mit.edu              f9.ijs.si                lnf.infn.it              ruk.cuni.cz
atlass01.physik.uni-bonn.de fnal.gov                 lngs.infn.it             rz.uni-jena.de
atlas.umich.edu             fusione.it               lrz-muenchen.de          sanchin.se
ba.infn.it                  glue.umd.edu             lsa.umich.edu            sbp.ri.cmu.edu
bazquux.org                 gppc.de                  math.unifi.it            scoobydoo.psc.edu
biocenter.helsinki.fi       grand.central.org        mcc.ac.gb                scotch.ece.cmu.edu
bme.hu                      grif.fr                  md.kth.se                s-et.aau.dk
caspur.it                   hackish.org              mech.kth.se              setfilepointer.com
cats.ucsc.edu               hallf.kth.se             membrain.com             sinenomine.net
cede.psu.edu                hep.caltech.edu          meteo.uni-koeln.de       sipb.mit.edu
cern.ch                     hep-ex.physics.metu.edu.tr monrove                slackers.net
cgv.tugraz.at               hephy.at                 monrove.is-a-geek.com    slac.stanford.edu
chem.cmu.edu                hep.man.ac.uk            mpe.mpg.de               soap.mit.edu
ciemat.es                   hep.sc.edu               mrph.org                 sodre.cx
```

Y nuestra celda tiene las siguientes carpetas por ahora (vacias)

```
goku:/# ls /afs/monrove.is-a-geek.com/
service  user
```

AHORA QUE YA ESTÁ CONFIGURADO, editamos **/etc/openafs/afs.conf.client** y reemplazamos AFS_CLIENT=**false** por AFS_CLIENT=**true** para que se inicie al reiniciar la máquina.

Si seguimos autenticados como afsadmin/admin podemos añadir carpetas y archivos en /afs/monrove.is-a-geek.com/*

si no lo estamos nos dará error por que no tenemos permiso para ello

kdestroy

unlog

```
goku:/afs/monrove.is-a-geek.com# mkdir hola
mkdir: no se puede crear el directorio Â«holaÂ»: Permiso denegado
goku:/afs/monrove.is-a-geek.com#
```

Administración básica de AFS

El comando **fs** es el principal enlace administrativo con el **cache manager**, con el podemos realizar tareas de administración sobre:

- Como interacciona el cache manager con las máquinas servidores.
- Administración de las ACL.
- Administración de los servidores, volúmenes o particiones que contienen un directorio.
- Administración la cache del cliente local y su información.
- Administración de los puntos de montaje de los volúmenes.
- Monitorización y traza(debug).
- Administración de la interacción del cache manager con otros sistemas de ficheros.

Todo esto viene en **man fs** así como los comandos que se pueden ejecutar.

Para ver los permisos sobre los ficheros si tenemos ticket de AFS de administrador y nos colocamos sobre /afs/monrove.is-a-geek.com

```
goku:/afs/monrove.is-a-geek.com# fs la
Access list for . is
Normal rights:
  system:administrators rlidwka
  system:anyuser rl
```

También podemos ver la quota de disco que hay asignada y modificarla:

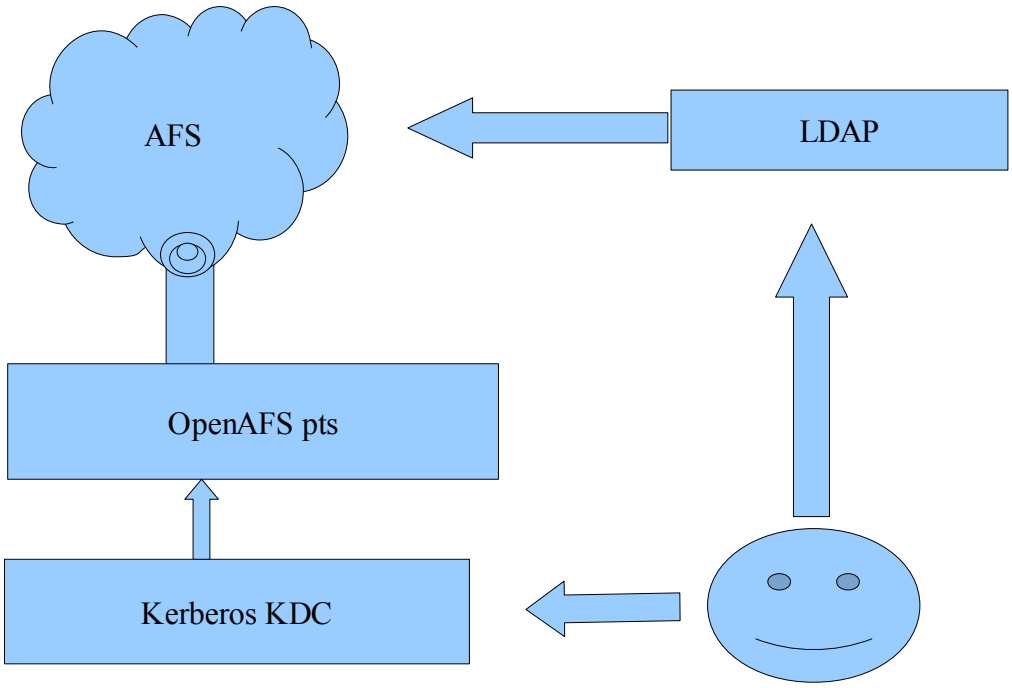
```
goku:/afs/monrove.is-a-geek.com# fs lq
Volume Name                    Quota      Used %Used   Partition
root.cell                       5000         4    0%          2%
```

Cambiando la quota de 5 megas a 100

```
goku:/afs/monrove.is-a-geek.com# fs sq . 100000
fs: You don't have the required access rights on '.'
goku:/afs/monrove.is-a-geek.com# kdestroy
goku:/afs/monrove.is-a-geek.com# unlog
goku:/afs/monrove.is-a-geek.com# kinit afsadmin/admin
Password for afsadmin/admin@MONROVE.IS-A-GEEK.COM:
goku:/afs/monrove.is-a-geek.com# aklog
goku:/afs/monrove.is-a-geek.com# fs sq . 100000
goku:/afs/monrove.is-a-geek.com# fs lq
Volume Name                    Quota      Used %Used   Partition
root.cell                     100000         4    0%          2%
```

Gestión de usuarios:

Pues aquí tenemos algo bastante interesante, ya que tenemos 2 métodos de autentificación el de kerberos y el de afs por lo tanto tenemos que tener las credenciales del usuario duplicadas.... y el resto de información del usuario en ldap.



Por lo tanto tendremos que crear el usuario tanto en kerberos (kadmin.local) como en OpenAFS.

En kerberos para crear el usuario lo que haremos será crear un principal para este:

```
#kadmin.local
#kadmin.local:addprinc pruebau
```
—-- le ponemos una contraseña y salimos.
```
#kadmin.local:q
```

En OpenAFS tenemos que hacer lo siguiente:
#kinit afsadmin/admin (para autentificarnos con kerberos)
#aklog (para autentificarnos con afs usando el ticket que tenemos retenido)

Una vez que estamos identificado como el administrador de afs tenemos acceso a las gestiones de la celda.
Creamos el usuario con:
pts createuser pruebau 10000
(le asignamos el mismo uid que tiene en ldap)
Para este usuario tenemos que asignarle un volumen dentro de la partición , tenemos que crear el volumen.
 # vos create goku a user.pruebau 10000

sintaxis: vos create servidor partición user.usuario uidnumber

Y ahora creamos el directorio propiedad del usuario:
```
#fs mkm /afs/monrove.is-a-geek.com/user/pruebau user.pruebau -rw
#cd /afs/monrove.is-a-geek.com/user/
```

#fs la pruebau/

con **fs la** nos muestra información sobre la lista de acceso y los privilegios, haciendo esto podemos ver que aun no puede acceder nuestro nuevo usuario así que tendremos que asignárselos con:

#fs sa pruebau/ pruebau all

Si después de esto listamos de nuevo los privilegios podemos ver que pruebau ya tiene privilegios sobre su home.

Home de los usuarios:

Esta información se encuentra en ldap, podemos hacer dos cosas:
1ª hacer un simlink del /home a /afs/monrove.is-a-geek.com/user.
(lo malo que habría que hacerlo en todas las máquinas)
2ª cambiar el home del usuario en ldap

En mi caso he optado por la segunda opción ya que me parece mas adecuada.

Añadiendo un servidor extra

Primero instalamos y configuramos el cliente de kerberos en el nuevo servidor.
 # aptitude install krb5-clients krb5-user

 # nano /etc/krb5.conf
configurar como corresponde el cliente kerberos

Configurar la red y el dns del servidor, ponerle ip estática.
Instalar module-assistant
 # aptitude install module-assistant

hacemos lo mismo que con el servidor principal, preparamos una partición en ext3 con fdisk y mkfs.ext3 y la montamos en /vicepa
Después añadimos en los ficheros de configuración del servidor los datos de nuestra celda tal y como hicimos en el servidor principal.

Una vez hecho esto tenemos que exportar a este servidor el keytab de la celda e importarlo con asetkey.
Lo copiamos desde el otro servidor:
scp /etc/krb5.keytab.afs piccolo:/etc/krb5.keytab.afs

Lo importamos:
asetkey add 3 /etc/krb5.keytab.afs afs

Editamos /etc/openafs/server/CellServDB
y añadimos dentro los servidores
Después creamos el archivo /etc/openafs/server/UserList y añadimos nuestro usuario admin afsadmin.admin
Iniciamos el fileserver con /etc/init.d/openafs-fileserver start
Ahora configuramos los servicios en la nueva máquina servidor haciendo lo siguiente:

bos create -server piccolo -instance ptserver -type simple -cmd /usr/lib/openafs/ptserver -cell monrove.is-a-geek.com -localauth

bos create -server piccolo -instance fs -type fs -cmd /usr/lib/openafs/fileserver -cmd /usr/lib/openafs/volserver -cmd /usr/lib/openafs/salvager -cell monrove.is-a-geek.com -localauth

Ahora solo tenemos que reiniciar el servicio:

/etc/init.d/openafs-fileserver restart

Y ya tenemos dos servidores afs, dos particiones separadas, ahora vamos a usar a piccolo como replica de goku ya que leyendo la documentación de afs he observado lo siguiente que denota la importancia de replicar los volúmenes root.afs y root.cell para que todo vaya mas fluido.

"If you are replicating any volumes, you must replicate the **root.afs** and **root.cell** volumes, preferably at two or three sites each (even if your cell only has two or three file server machines). The Cache Manager needs to pass through the directories corresponding to the **root.afs** and **root.cell** volumes as it interprets any pathname. The unavailability of these volumes makes all other volumes unavailable too, even if the file server machines storing the other volumes are still functioning.

Another reason to replicate the **root.afs** volume is that it can lessen the load on the File Server machine. The Cache Manager has a bias to access a read-only version of the **root.afs** volume if it is replicate, which puts the Cache Manager onto the *read-only path* through the AFS filespace. While on the read-only path, the Cache Manager attempts to access a read-only copy of replicated volumes. The File Server needs to track only one callback per Cache Manager for all of the data in a read-only volume, rather than the one callback per file it must track for read/write volumes. Fewer callbacks translate into a smaller load on the File Server."

Por ello deberíamos replicarlos y tener algo como lo siguiente:

```
goku:/# vos listvol piccolo
Total number of volumes on server piccolo partition /vicepa: 3
datos.readonly                  536870925 RO          2 K On-line
root.afs.readonly               536870913 RO        200 K On-line
root.cell.readonly              536870916 RO          4 K On-line
```

Replicación y creación de volúmenes

La creación y replicación de volúmenes es algo muy simple con openafs.
Para crear un volumen basta con ejecutar lo siguiente:

vos create goku vicepa datos
vos create (server) (particion) (nombrevolumen)

```
goku:/# vos create goku vicepa datos
Volume 536870924 created on partition /vicepa of goku
goku:/# vos listvol goku
Total number of volumes on server goku partition /vicepa: 7

datos                           536870924 RW          2 K
 On-line
root.afs                        536870912 RW        200 K
 On-line
root.afs.readonly               536870913 RO        200 K
 On-line
root.cell                       536870915 RW          4 K
 On-line
root.cell.readonly              536870916 RO          4 K
 On-line
service                         536870921 RW          2 K
 On-line
user                            536870918 RW          2 K
 On-line

Total volumes onLine 7 ; Total volumes offLine 0 ; Total bu
sy 0
```

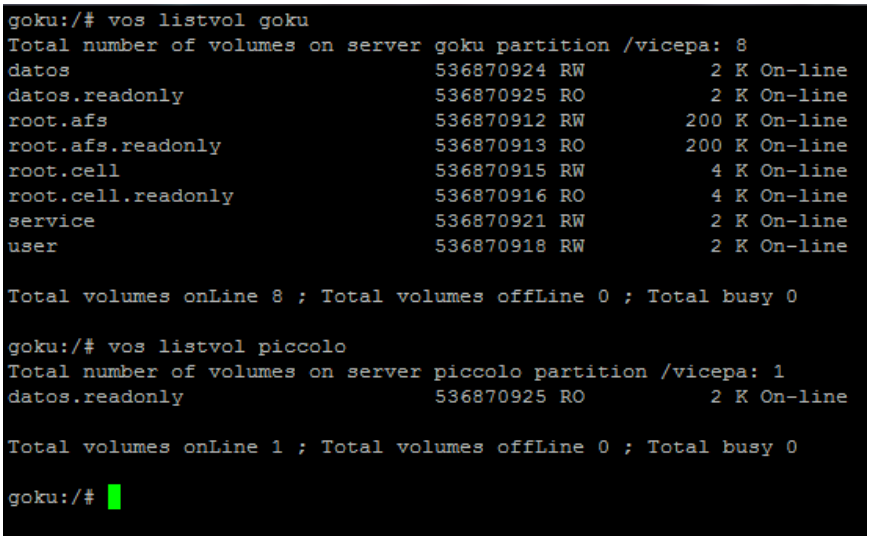
y para replicarlo simplemente añadir un sitio de replicación con el comando:

vos addsite -server goku -partition vicepa -i datos
vos addsite -server piccolo -partition vicepa -i datos

Ademas para que se actualicen hay que hacerlo mediante el comando:

vos release datos
vos release (nombrevolumen)

Podemos ver en la siguiente captura que los dos sitios que hemos creado son de solo lectura (RO)



Para que los cambios se apliquen en las copias hay que usar el **vos release** hay que tener esto **muy en cuenta**.

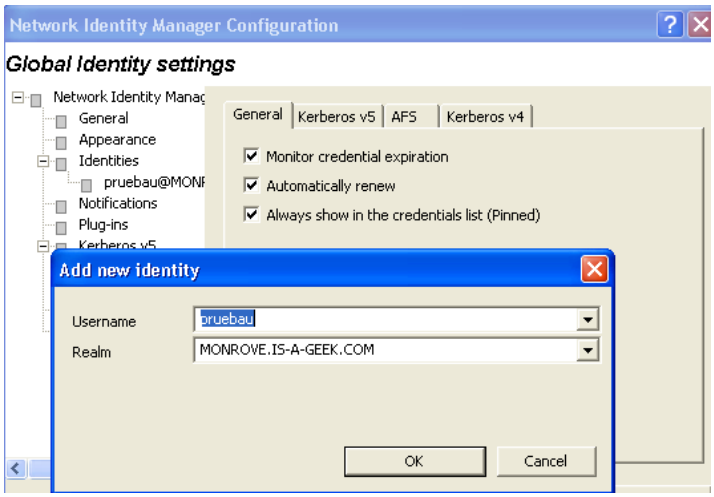
Configuración de los clientes:

Windows XP:

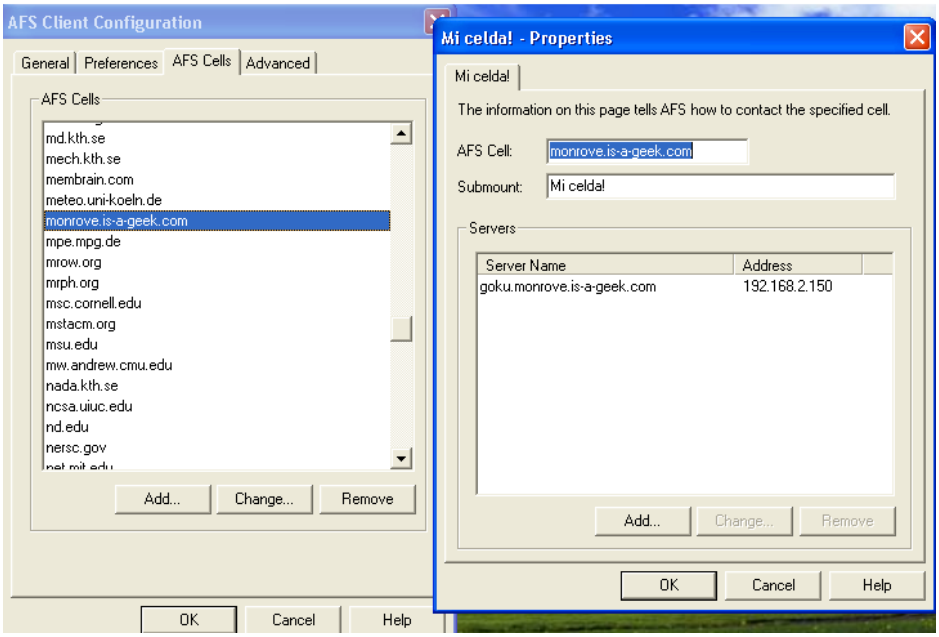
Descargamos kerberos y openafs para windows xp desde aquí http://www.openafs.org/windows.html y los instalamos, despues tendremos que configurarlos para que se adapten a nuestra celda y a nuestro reino kerberos.

Editamos con el **notepad el fichero c:/windows/krb5.ini**
veremos que es como el /etc/krb5.conf en unix por lo que realizamos la misma configuración en el.

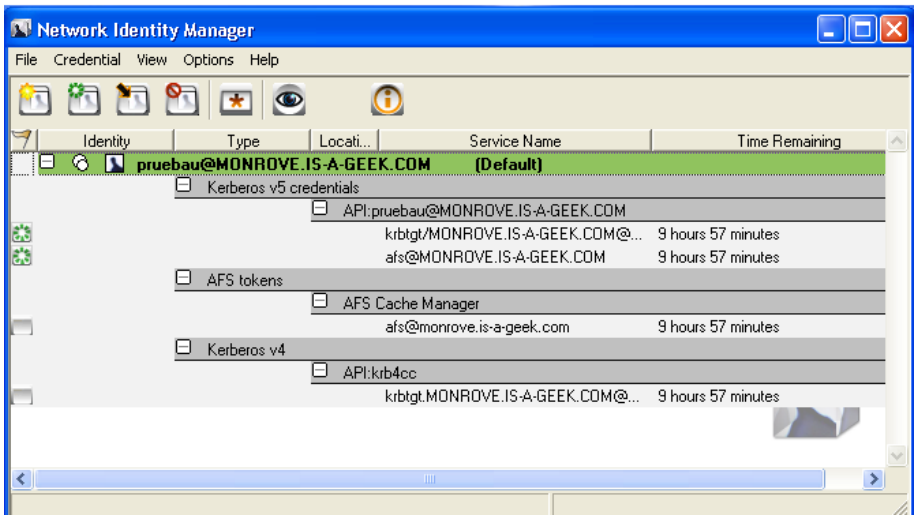
Después de configurar las opciones de kerberos le damos a añadir nueva identidad y usamos el usuario pruebau que tenemos ya creado



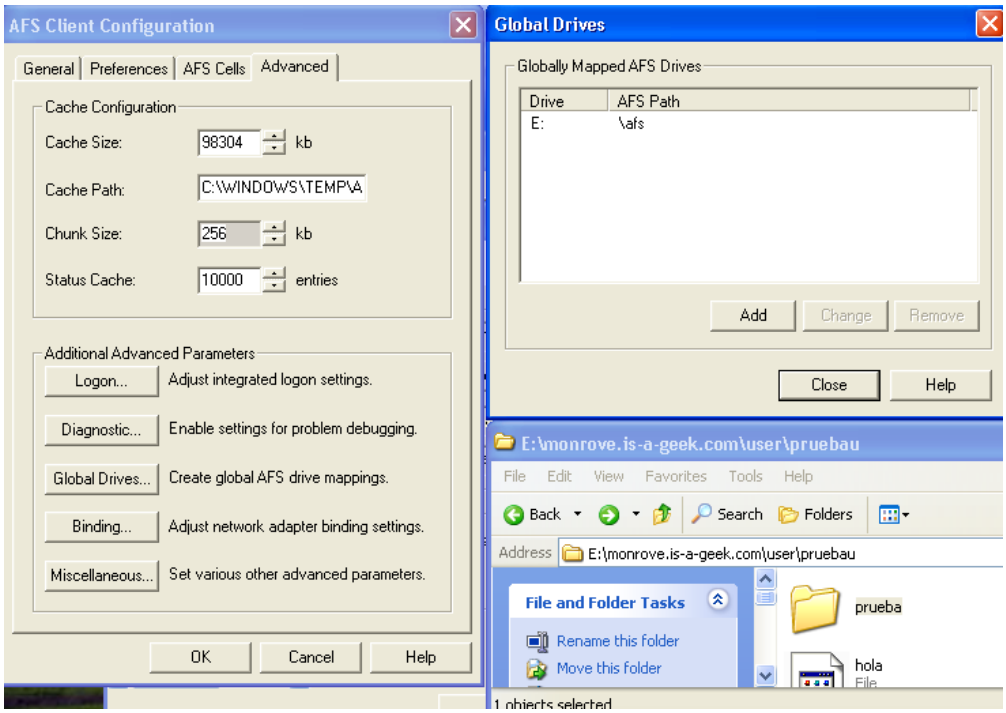
Desde el gestor de identidades podemos abrir la configuración de afs y ahí tendremos que añadir la celda y sus datos en las pestañas correspondientes



Luego ya podemos iniciar sesión y adquirir los tickets correspondientes para poder acceder a la celda afs



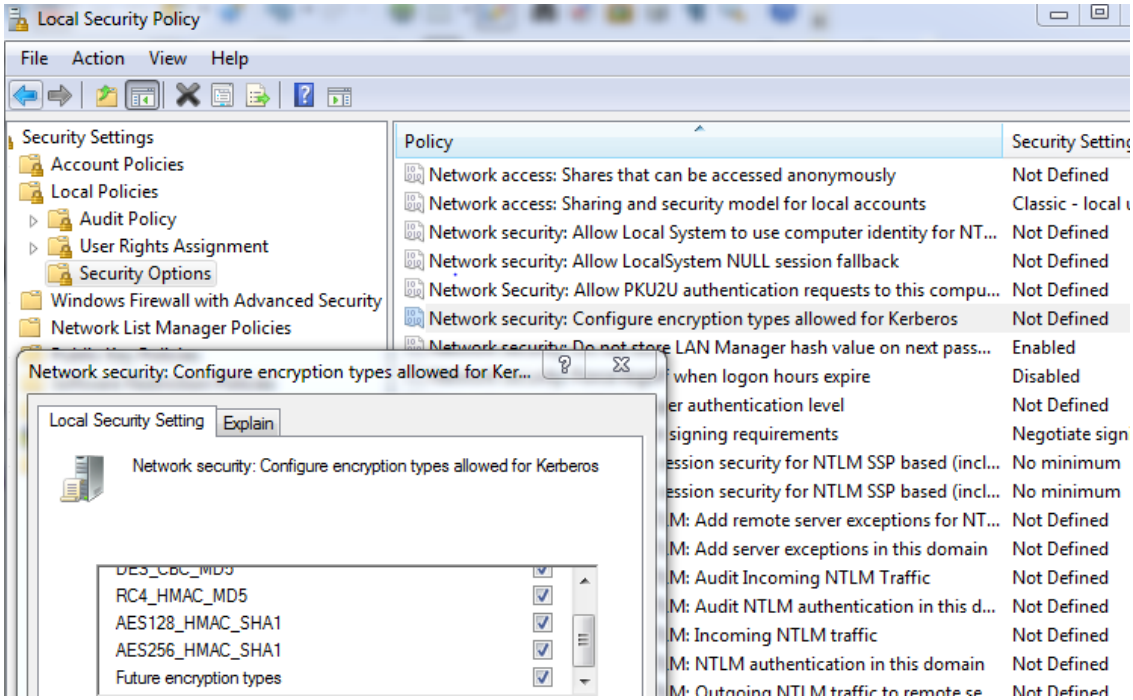
Para montar el directorio /afs tendremos que pulsar en la pestaña advanced sobre Global drives y añadir la unidad sobre la que vamos a montar /afs



Podemos ver en la parte inferior de la captura como hemos podido acceder al directorio afs del usuario pruebau, con el que nos hemos identificado mediante tickets kerberos.

Windows 7

Primero tenemos que activar los tipos de encripción soportados para kerberos así que hacemos lo siguiente:
Escribimos en buscar “local security policy”



Y asignamos los tipos de encripción permitidos para kerberos en nuestro caso des-cbc-crc para afs y el resto para los principales de kerberos.

Entramos en http://www.openafs.org/windows.html
Descargamos el instalador de kerberos y de openafs y los ejecutamos, elegimos la configuración típica nos pedirá reiniciar la máquina, como no.....

Despues simplemente seguir las mismas instrucciones que para windows xp

Debian:

Necesitamos una máquina con debian y con el **openafs-client** instalado, para ello tendremos que compilar el módulo de openafs e instalar openafs-client con apt para esta máquina cliente y configurar el pam.d de manera que pueda iniciar sesión, además intentaremos que sea transparente para el usuario.

Instalamos los módulos necesarios de pam y kerberos para el cliente:

`aptitude install libpam-openafs-session openafs-krb5 libpam-krb5 libpam-openafs-session libpam-ldap libnss-ldap krb5-user`

Importante no olvidarse de hacer un dpkg-reconfigure libpam-ldap para poner la información en condiciones, sino seguro que nos da algun fallo.

Instalamos las herramientas de ldap para poder recoger la información de los usuarios cuando inicien sesión:

`aptitude install openldap-utils`

las configuramos para que usen a goku y lo probamos con un ldapsearch -x

```
client:/home/javi# nano /etc/ldap/ldap.conf
client:/home/javi# slapcat
bash: slapcat: command not found
client:/home/javi# ldapsearch -x
# extended LDIF
#
# LDAPv3
# base <dc=monrove,dc=is-a-geek,dc=com> (default) with
# filter: (objectclass=*)
# requesting: ALL
#

# monrove.is-a-geek.com
```

Una vez probado esto, añadimos en /etc/nsswitch.conf las lineas que harán que los datos de la máquina no se cojan solo del fichero passwd y group sino que se puedan obtener tambien por ldap

```
# /etc/nsswitch.conf
#
# Example configuration of GNU Name Se
# If you have the `glibc-doc-reference
# `info libc "Name Service Switch"' fo

passwd:         compat ldap
group:          compat ldap
shadow:         compat
```

Modificamos el fichero que contiene las direcciones de los servidores en la máquina del cliente para que afs-client sepa con quien hablar, podemos copiarlo de uno de nuestros servidores o añadirlas a mano.

```
  GNU nano 2.0.7        Fichero: /etc/openafs/CellServDB

>monrove.is-a-geek.com
192.168.2.150           # goku.monrove.is-a-geek.com
192.168.2.151           # piccolo.monrove.is-a-geek.com
```

Después de eso podemos arrancar nuestro openafs-client
`/etc/init.d/openafs-client start`
y podemos decirle que se arranque al inicio modificando /etc/openafs/afs.conf.client y poniendo a true el valor de AFS_CLIENT:

```
AFS_CLIENT=true
AFS_AFSDB=false
AFS_CRYPT=false
AFS_DYNROOT=true
AFS_FAKESTAT=true
```

Para conseguir que se nos asignen tickets de Kerberos y AFS tokens cuando se hace login tendremos que tocar los ficheros de configuración del pam del cliente que están en /etc/pam.d

modificamos los /**etc/pam.d/common-***

common-auth:
```
auth    sufficient      pam_unix.so nullok_secure
auth    sufficient      pam_krb5.so use_first_pass
auth    optional        pam_afs_session.so program=/usr/bin/aklog
auth    required        pam_deny.so
```

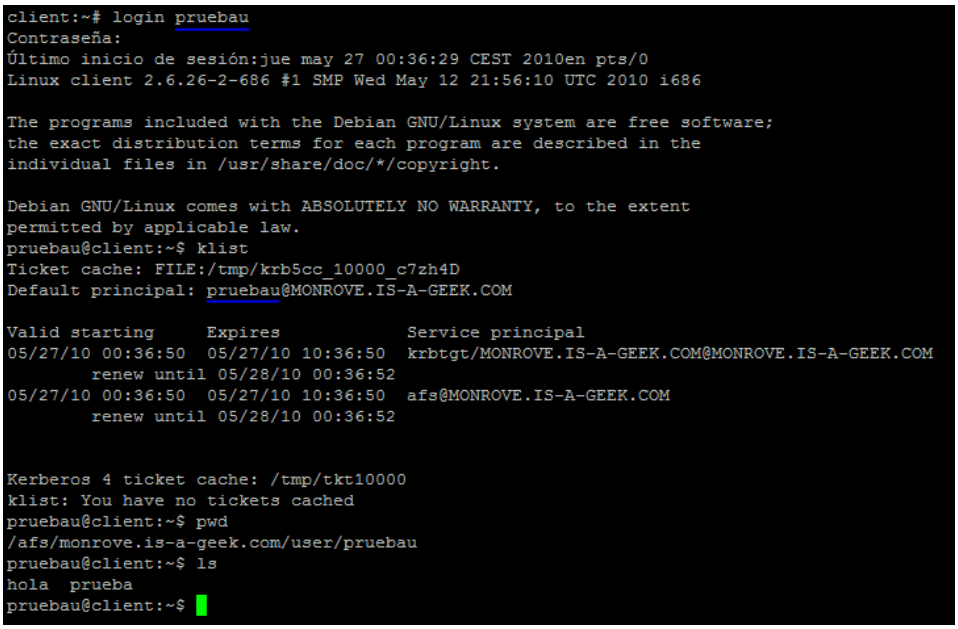
common-account:
```
account sufficient pam_unix.so
account sufficient pam_krb5.so
account required pam_ldap.so
account required pam_deny.so
```

common-password:
```
password   sufficient   pam_unix.so nullok obscure md5
password   sufficient   pam_krb5.so use_first_pass
```

common-session:
```
session required        pam_limits.so
session optional        pam_unix.so
session optional        pam_krb5.so minimum_uid=9000
session optional        pam_ldap.so
session optional        pam_afs_session.so program=/usr/bin/aklog
```

Usamos el modulo pam_afs_session para que ejecute aklog después de autentificarse con kerberos, con lo que establecemos los tickets necesarios para el acceso a /afs/monrove.is-a-geek.com/user/pruebau
ya que a este directorio solo podemos acceder si tenemos ticket de kerberos y de afs.

Comprobación:



Enlaces:

AFSMONITOR: http://docs.openafs.org/AdminGuide/ch08s04.html

AFS en general:
http://matienzo.org/project/setting-up-umich-afs-on-ubuntu // cliente

http://help.unc.edu

http://www.rjsystems.nl/en/2100-kerberos-openldap-openafs-client.php

http://techpubs.spinlocksolutions.com/dklar/afs.html#id2453178

http://www.debian-administration.org/article/OpenAFS_installation_on_Debian#id2455666

http://en.gentoo-wiki.com/wiki/OpenAFS_with_MIT_Kerberos#Kerberos_Installation

Administración AFS:
http://www-01.ibm.com/software/stormgmt/afs/manuals/Library/unix/en_US/HTML/AdminGd/auagd007.htm

http://www.debian-administration.org/article/OpenAFS_installation_on_Debian

autentificación kerberos+windows 7:

http://www.mcplusa.com/blog/2009/10/authentication-with-kerberos-on-windows-7-and-the-google-search-appliance/

Documentación AFS: http://docs.openafs.org/

Instalación de AFS+kerberos fecha 2003
http://www.scode.org/afs/openafs-install.txt

AFS Global Namespace:
http://docs.openafs.org/AdminGuide/ch02s03.html