JUNTA DE ANDALUCIA

CONSEJERIA DE EDUCACION
CEP Lora del Río

Sistema de cuentas de usuario centralizadas con Kerberos 5, OpenLDAP y NFSv4 en Debian GNU/Linux (lenny)

Alberto Molina Coballes, José Domingo Muñoz Rodríguez y José Luis Rodríguez Rodríguez.

6 de abril de 2010

En este documento se describen los pasos necesarios para configurar un equipo GNU/Linux para gestionar de forma centralizada las cuentas de usuarios de una red mediante Kerberos, LDAP y NFSv4. Este documento forma parte del curso *Servicios en GNU/Linux. Portal Educativo*, organizado por el CEP de Lora del Río (Sevilla) en 2010.

Usted es libre de copiar, distribuir y modificar este documento de acuerdo con las condiciones de la licencia Attribution-ShareAlike 3.0 de Creative Commons. Puede ver una copia de ésta en:
http://creativecommons.org/licenses/by-sa/3.0/es/

Índice

1. Introducción

La forma clásica de almacenar información sobre las cuentas de los usuarios en sistemas UNIX es mediante la información existente en los ficheros */etc/passwd*, */etc/shadow* y */etc/group*, método que funciona bien para gestionar las cuentas de usuarios en un solo equipo pero no es útil para tener un sistema centralizado de cuentas. Entendemos como sistema centralizado de cuentas aquel que permite a un usuario de una red usar diferentes equipos de la misma con los mismos datos de autenticación y pudiendo acceder siempre a sus ficheros, para ello tanto los datos de autenticación como los ficheros del usuario deben estar ubicados en servidores de la red en lugar de en cada equipo.

En los sistemas que utilizan cuentas centralizadas se suelen utilizar dos tipos de cuentas: lo que se denominan cuentas locales para los que se emplea el método tradicional de ficheros y mediante un segundo método para las cuentas de usuarios que puedan autenticarse en cualquier equipo de la red, lo que podríamos denominar *usuarios de red*.

En entornos UNIX durante bastante tiempo se ha utilizado NIS/NFS para la gestión centralizada de cuentas de usuarios en una red local y su configuración es muy sencilla, pero es una opción cada vez más en desuso por diferentes problemas que presenta, principalmente porque no funciona sobre TCP/IP y la comunicación entre el cliente y el servidor no es cifrada, lo que puede provocar problemas de seguridad.

Este documento se centra en explicar la forma de configurar un sistema centralizado de cuentas de usuario, gestionando la información de autenticación con Kerberos, guardando el resto de la información de la cuenta en un directorio OpenLDAP y utilizando *Network File System version 4* (NFS4) para los ficheros de los usuarios.

Aviso: Este documento se utiliza con fines educativos y no tiene como objetivo utilizarse en una implantación real. Algún lector interesado podría utilizarlo como primer paso para la implantación en un entorno real, pero debería realizar posteriormentes algunos ajustes para la administración y seguridad de los servicios utilizados.

2. Características del montaje

Todo el montaje se va a realizar en dos equipos con sistema Debian GNU/Linux (lenny), uno que va a actuar como servidor, en el que se realizarán la mayoría de pasos y otro que actuará como cliente. Los nombres y direcciones IP de los equipos son:

Servidor	Cliente	Red
`avatar`	`cliente`	`example.com`
`192.168.2.1`	`192.168.2.2`	`192.168.2.0/24`

donde hemos utilizado *example.com* como nombre de dominio para los equipos de la red, siendo éste uno de los nombres de dominio reservado para documentación de acuerdo con la RFC-2606 [1].

3. Configuración previa

Es un requisito previo para el correcto funcionamiento de los servicios posteriores (en particular de Kerberos) la configuración de un servidor DNS con todos los nombres de los equipos

de la red y un servidor de hora para sincronizar los relojes de todos los equipos[1]. En esta sección explicaremos de forma sucinta la manera de configurar estos servicios en la red, por simplicidad todos los servicios se instalarán en *avatar*.

3.1. Instalación de bind9

En primer lugar instalamos el paquete `bind9`:

```
avatar:~# aptitude install bind9
```

Definimos las zonas de resolución local:

/etc/bind/named.conf.local

```
zone "example.com" {
        type master;
        file "/var/cache/bind/example.com.db";
};

zone "2.168.192.in-addr.arpa" {
        type master;
        file "/var/cache/bind/192.168.2.db";
};
```

y creamos los ficheros para la resolución directa e inversa, donde se han definido además alias para las direcciones ldap y Kerberos que se utilizarán posteriormente y entradas SRV [2] correspondientes a los servicios LDAP y Kerberos:

/var/cache/bind/example.com.db

```
$TTL 86400
@   IN  SOA  avatar.example.com.  hostmaster.example.com. (
                                        1 ; Serial
                                        604800 ; Refresh
                                        86400  ; Retry
                                        2419200 ; Expire
                                        604800 ) ; TTL
;
@                   IN      NS      avatar
cliente             IN      A       192.168.2.2
avatar              IN      A       192.168.2.1
kerberos            IN      CNAME   avatar
ldap                IN      CNAME   avatar

_kerberos           IN      TXT     "EXAMPLE.COM"

_kerberos._udp      IN      SRV     0 0 88  avatar.example.com.
_kerberos_adm._tcp  IN      SRV     0 0 749 avatar.example.com.
_ldap._tcp          IN      SRV     0 0 389 avatar.example.com.
```

/var/cache/bind/example.com.db

```
$TTL 86400
@   IN   SOA   avatar.example.com.  hostmaster.example.com. (
```

[1] Realmente es necesario que todos los equipos de la red puedan hacer resolución directa e inversa de los nombres y que tengan la misma hora

[2] Algunas aplicaciones utilizan entradas DNS tipo SRV y por eso la hemos incluido, aunque no son necesarias para lo expuesto en este documento

```
                        1 ; Serial
                        604800 ; Refresh
                        86400 ; Retry
                        2419200 ; Expire
                        604800 ) ; Default TTL

@       NS      avatar.example.com.
1       PTR     avatar.example.com.
2       PTR     cliente.example.com.
```

Después de reiniciar el servicio, comprobamos el correcto funcionamiento del servidor DNS en los registros del sistema y realizando una consultas con *dig* como:

```
avatar:~$ dig ldap.example.com
```

```
; <<>> DiG 9.5.1-P3 <<>> ldap.example.com
;; global options:  printcmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 17880
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 1, ADDITIONAL: 0

;; QUESTION SECTION:
;ldap.example.com.                IN    A

;; ANSWER SECTION:
ldap.example.com.          86400   IN    CNAME   avatar.example.com.
avatar.example.com.        86400   IN    A       192.168.2.1

;; AUTHORITY SECTION:
example.com.               86400   IN    NS      avatar.example.com.

;; Query time: 4 msec
;; SERVER: 127.0.0.1#53(127.0.0.1)
;; WHEN: Tue Jan  5 17:02:51 2010
;; MSG SIZE  rcvd: 85
```

3.2. Configuración de los clientes DNS

En todos los equipos de la red habrá que definir el dominio de búsqueda y la dirección del servidor DNS, esto es algo tan sencillo como hacer en avatar:

/etc/resolv.conf

```
domain example.com
search example.com
nameserver 127.0.0.1
```

En el resto de equipos habría que hacer la misma modificación, aunque considerando que en ese caso la dirección IP del servidor de nombres es `192.168.2.1`.

Tras hacer todas estas modificaciones, es conveniente verificar que el FQDN está bien configurado en todos los equipos de la red[3]:

```
avatar:~$ hostname -f

avatar.example.com
```

[3] Hay que comprobar que no haya entradas mal definidas en el fichero */etc/hosts*

3.3. Instalación de ntp

Para instalar un servidor de hora en el equipo hay que hacer:

```
avatar:~# aptitude install ntp
```

Para comprobar que se ha realizado una conexión a servidores de hora públicos y se ha sincronizado el reloj local, verificamos la salida de la instrucción:

```
avatar:~# ntpq -np
```

remote	refid	st	t	when	poll	reach	delay	offset	jitter
+84.88.69.10	193.67.79.202	2	u	140	512	277	31.764	−2.308	2.665
+163.117.131.239	219.76.239.59	2	u	389	512	377	14.545	0.161	1.546
*158.227.98.15	.GPS.	1	u	93	512	377	25.295	−0.178	2.387
−77.226.252.14	150.214.94.5	2	u	390	512	377	65.936	−12.789	0.301

En los clientes de la red instalaremos el mismo paquete, pero habrá que definir el servidor ntp de la red y obviar los servidores de hora públicos:

/etc/ntp.conf

```
15 # You do need to talk to an NTP server or two (or three).
16 server avatar.example.com
17
18 # pool.ntp.org maps to about 1000 low-stratum NTP servers.  Your server
19 # will pick a different set every time it starts up.  Please consider
20 # joining the pool: <http://www.pool.ntp.org/join.html>
21 #server 0.debian.pool.ntp.org iburst dynamic
22 #server 1.debian.pool.ntp.org iburst dynamic
23 #server 2.debian.pool.ntp.org iburst dynamic
24 #server 3.debian.pool.ntp.org iburst dynamic
```

Al comprobar ahora el funcionamiento de ntp comprobamos que sólo se realizan consultas al servidor ntp local:

```
cliente:~$ ntpq -np
```

remote	refid	st	t	when	poll	reach	delay	offset	jitter
*192.168.2.1	81.19.96.148	3	u	24	64	77	0.632	−5.317	1.499

4. Instalación del servidor OpenLDAP

El servidor LDAP es uno de los componentes fundamentales del sistema de cuentas que estamos organizando, de las diferentes implementaciones libres de LDAP que hay la que se utiliza habitualmente en Debian es OpenLDAP que viene en el paquete *slapd*, por lo que procedemos a instalarlo:

```
avatar:~# aptitude install slapd
```

Dependiendo de la configuración del paquete *debconf* nos pedirá algunos parámetros durante la instalación, en nuestro caso sólo fue:

- Contraseña del Administrador del directorio

y creará un directorio con dos entradas que podemos volcar a la consola con la instrucción *slapcat*:

```
avatar:~# slapcat
```

```
dn: dc=example,dc=com
objectClass: top
objectClass: dcObject
objectClass: organization
o: example.com
dc: example
structuralObjectClass: organization
entryUUID: 695d13f6-8bd4-102e-8841-11275fe7c8e6
creatorsName:
createTimestamp: 20100102102202Z
entryCSN: 20100102102202.072322Z#000000#000#000000
modifiersName:
modifyTimestamp: 20100102102202Z

dn: cn=admin,dc=example,dc=com
objectClass: simpleSecurityObject
objectClass: organizationalRole
cn: admin
description: LDAP administrator
userPassword:: e2NyeXB0fWppRFFoTzNuQXUuQS4=
structuralObjectClass: organizationalRole
entryUUID: 695dc684-8bd4-102e-8842-11275fe7c8e6
creatorsName:
createTimestamp: 20100102102202Z
entryCSN: 20100102102202.077087Z#000000#000#000000
modifiersName:
modifyTimestamp: 20100102102202Z
```

En la salida anterior podemos ver dos entradas en formato ldif, identificadas cada una con un nombre distintivo (*Distinguished Name*) que es único para cada entrada y una serie de atributos. *slapcat* está incluída en el paquete *slapd* y muestra los objetos del directorio sin establecer una conexión LDAP propiamente, aunque sólo se puede ejecutar desde el propio servidor.

En caso de que la base del directorio no esté correctamente definida se podría reconfigurar de nuevo con:

```
avatar:~# dpkg-reconfigure slapd
```

y en este caso nos preguntará más detalles de la configuración, en concreto:

- Nombre de dominio DNS: example.com
- Nombre de la Organización: Example
- Contraseña del administrador: (la que corresponda)
- Motor de base de datos a utilizar: BDB
- ¿Permitir el protocolo LDAPv2?: No

En caso de que esto no fuese suficiente, podríamos purgar el paquete, ir al directorio */var/lib/ldap* y borrar todo su contenido y a continuación instalar de nuevo el paquete *slapd*.

Ya por último, recordar que todos los valores que hemos introducido y otros para los que se asumen los valores, se guardan en el fichero */etc/ldap/slapd.conf*

4.1. Estructura básica del directorio

La estructura del árbol del directorio variará en función de la información que queramos almacenar y de la complejidad de ésta, en nuestro caso vamos utilizar el directorio para almacenar información de las cuentas de los usuarios y los grupos a los que pertenecen. En estos casos habitualmente se utiliza una estructura del estilo a la que aparece en la figura 1.

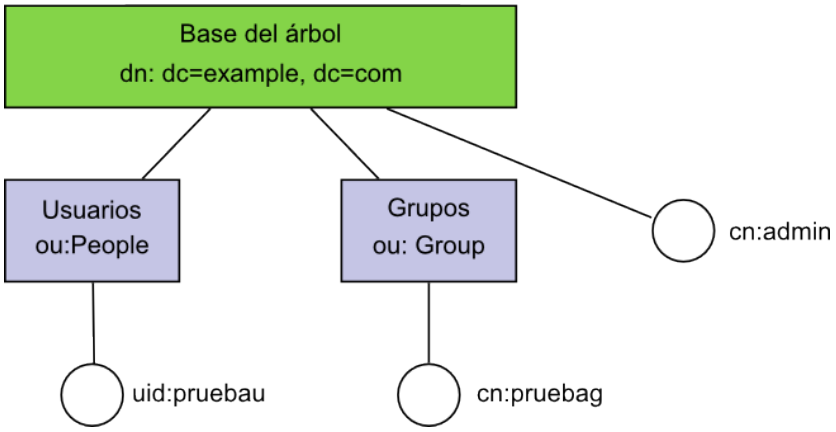

Figura 1: Esquema básico del árbol LDAP.

Como puede verse en la figura, se han creado dos unidades organizativas (*ou*) llamadas *People* y *Group* donde obviamente se incluirán las entradas de los usuarios y los grupos respectivamente, además de un objeto en cada unidad organizativa (el usuario *pruebau* y el grupo *pruebag*). En el caso de que quisiéramos almacenar información de otros objetos como impresoras, ordenadores, etc. deberíamos añadir nuevas unidades organizativas.

Para añadir las entradas anteriores, debemos crear un fichero en formato ldif, que denominaremos *inicio.ldif*, con el siguiente contenido:

inicio.ldif

```
dn: ou=People,dc=example,dc=com
ou: People
objectClass: top
objectClass: organizationalUnit

dn: ou=Group,dc=example,dc=com
ou: Group
objectClass: top
objectClass: organizationalUnit

dn: cn=pruebag,ou=Group,dc=example,dc=com
objectClass: posixGroup
objectClass: top
cn: pruebag
gidNumber: 2000

dn: uid=pruebau,ou=People,dc=example,dc=com
objectClass: account
objectClass: posixAccount
objectClass: top
cn: pruebau
uid: pruebau
loginShell: /bin/bash
uidNumber: 2000
```

```
25 gidNumber: 2000
26 homeDirectory: /home/users/pruebau
```

donde se ha optado por utilizar *uid* para identificar la entradas correspondientes a los usuarios, aunque también se podría utilizar *cn*. El atributo *uidNumber* se corresponde con el número de usuario o UID del sistema y hemos utilizado un número a partir de 2000 para evitar solapamiento con los usuarios locales.

Para insertar esta entrada en el directorio es necesario utilizar algún cliente LDAP desde cualquier equipo de la red [4], por lo que instalaremos el paquete *ldap-utils*:

```
avatar:~# aptitude install ldap-utils
```

y configuraremos de forma apropiada todos los clientes ldap:

/etc/ldap/ldap.conf

```
8 BASE    dc=example,dc=com
9 URI     ldap://ldap.example.com
```

Para insertar el fichero ldif:

```
avatar:~# ldapadd -x -D "cn=admin,dc=example,dc=com" -W -f inicio.ldif
Enter LDAP Password: (se introduce la contraseña)
```

```
adding new entry "ou=People,dc=example,dc=com"

adding new entry "ou=Group,dc=example,dc=com"

adding new entry "cn=pruebag,ou=Group,dc=example,dc=com"

adding new entry "uid=pruebau,ou=People,dc=example,dc=com"
```

si no se ha producido ningún error, podremos ver las nuevas entradas utilizando *slapcat* desde el propio servidor o *ldapsearch -x* desde cualquier cliente ldap.

Puesto que no existe el directorio home del usuario *pruebau*, tendremos que crearlo y asignarle el usuario y grupo propietario adecuado:

```
mkdir -p /home/users/pruebau
cp /etc/skel/.* /home/users/pruebau
chown -R 2000:2000 /home/users/pruebau
```

Si hacemos un listado del directorio home del usuario:

```
avatar:~# ls -al /home/users/pruebau/
```

```
total 20
drwxr-xr-x 2 2000 2000 4096 ene  2 21:26 .
drwxr-xr-x 4 root root 4096 ene  2 21:26 ..
-rw-r--r-- 1 2000 2000  220 ene  2 21:26 .bash_logout
-rw-r--r-- 1 2000 2000 3116 ene  2 21:26 .bashrc
-rw-r--r-- 1 2000 2000  675 ene  2 21:26 .profile
```

podemos ver que el sistema no es capaz de asociar los UID/GID a ningún usuario existente puesto que todavía no está configurado para hacer la búsqueda en el directorio.

[4] desde el propio servidor es posible utilizar *slapadd* pero es necesario parar el servicio

4.2. Definición de entradas destacadas

Como hemos mencionado anteriormente cada entrada del directorio se define por un *dn* único, que la identifica dentro del árbol y define la ruta que hay que seguir hasta ella. Hay algunas entradas que se utilizan frecuentemente y por ello las definimos a continuación:

base dn Nombre distintivo de la raíz del directorio.

root dn Nombre distintivo del administrador del directorio LDAP.

bind dn Nombre distintivo del usuario que establece la conexión, si el usuario que se conecta es el administrador, entonces coincide con el *root dn*.

relative dn (rdn) Nombre distintivo relativo, se identifica con el atributo único que define el objeto, como por ejemplo *uid=pruebau*. El *rdn* no identifica la ubicación en el árbol del objeto, eso se hace mediante el *dn*.

5. Configuración del cliente LDAP. Name Service Switch (nss)

Debemos configurar el sistema para que pueda obtener del directorio LDAP los siguientes servicios[5]:

Información de la cuenta del usuario Asignando a un usuario que ha ingresado en el sistema su directorio home, UID, shell, etc. tal como haría el sistema con un usuario local a través del fichero */etc/passwd*.

Name service switch Estableciendo la relación entre el UID/GID de un usuario o un grupo y su correspondiente nombre, como por ejemplo al crear un fichero, hacer un listado, etc. tal como haría el sistema con un usuario local a través de los ficheros */etc/passwd* y */etc/group*.

Asignación de grupos Asignando los grupos adicionales a los que pertenece cada usuario tal como haría el sistema con un usuario local a través del fichero */etc/group*, que se realiza añadiendo líneas con el atributo *memberUid* dentro del objeto de cada grupo.

El sistema encargado de establecer la correspondencia entre los UID/GID y los nombres de los usuarios y los grupos que estén en el directorio LDAP se denomina *name service switch* (nss) y vamos a configurarlo a continuación para que consulte esta correspondencia en el directorio LDAP además de en los ficheros *passwd* y *group*.

5.1. libnss-ldap

Para que el nss sea capaz de consultar a un directorio LDAP, debemos instalar el paquete *libnss-ldap*, aunque lo haremos añadiendo el parámetro *--no-install-recommends* para que no se instale el paquete recomendado *libpam-ldap* para la autenticación de usuarios con LDAP, ya que como hemos mencionado anteriormente la autenticación de usuarios se va a realizar con Kerberos:

```
avatar:~# apt-get install --no-install-recommends libnss-ldap
```

y configurar los siguientes puntos:

[5] Del servicio de autenticación debe encargarse Kerberos

- Identificador del servidor LDAP: *ldap:///ldap.example.com*
- Nombre distintivo (dn) de la base: *dc=example,dc=com*
- Versión de LDAP: 3
- Cuenta del administrador: (ignorar)
- Contraseña del administrador: (ignorar)

Recomendamos ignorar los dos últimos puntos en lugar de rellenarlos, ya que no es necesario tener privilegios de administrador para que puedan realizarse consultas nss al directorio, es mucho más sencillo realizar consultas anónimas y además es mucho más seguro al no tener que incluir la contraseña del administrador del LDAP en todos los clientes de la red.

Para anular el valor de la cuenta del administrador es necesario comentar la siguiente línea:

/etc/libnss-ldap.conf

```
53 #rootbinddn cn=manager,dc=example,dc=net
```

y borrar la contraseña del administrador de LDAP si existiera:

```
avatar:~# rm -f /etc/libnss-ldap.secret
```

5.2. Modificación de /etc/nsswitch.conf

Editamos este fichero y modificamos las líneas correspondientes a *passwd* y *group*[6], que tienen originalmente el siguiente aspecto (indica al sistema que tiene que buscar correspondencia entre UID/GID y nombres en servidores NIS y en los ficheros *passwd* y *group*):

/etc/nsswitch.conf

```
7 passwd: compat
8 group: compat
```

Para indicar al sistema que también busque en directorios LDAP hay que añadir:

/etc/nsswitch.conf

```
7 passwd: compat ldap
8 group: compat ldap
```

5.3. Prueba de funcionamiento de nss

Para comprobar el correcto funcionamiento de nss con ldap, repetimos el listado del directorio home del usuario de pruebas:

```
avatar:~# ls -al /home/users/pruebau/
```

```
total 20
drwxr-xr-x 2 pruebau pruebag 4096 ene  2 21:26 .
drwxr-xr-x 4 root    root    4096 ene  2 21:26 ..
-rw-r--r-- 1 pruebau pruebag  220 ene  2 21:26 .bash_logout
-rw-r--r-- 1 pruebau pruebag 3116 ene  2 21:26 .bashrc
-rw-r--r-- 1 pruebau pruebag  675 ene  2 21:26 .profile
```

[6] habría que incluir *shadow* si estuviesen definidos en LDAP valores asociados a las contraseñas, que no es nuestro caso ya que estos datos se van a gestionar con Kerberos

donde comprobamos que se ha cambiado el UID/GID por el correspondiente nombre de usuario y grupo, información que se ha obtenido mediante una consulta al directorio LDAP.

Otra forma de comprobación, quizás más elegante, es mediante la utilización de *getent*:

```
avatar:~# getent passwd pruebau
```

```
pruebau:*:2000:2000:pruebau:/home/users/pruebau:/bin/bash
```

```
avatar:~# getent group pruebag
```

```
pruebag:*:2000:
```

Como último punto en la configuración de nss, es recomendable la instalación del paquete *nscd*, que no necesita configuración, y que es simplemente un demonio que cachea las correspondencias entre los UID/GID y los nombres, con el fin de evitar consultas repetidas al directorio y agilizar así la respuesta.

6. Instalación del servidor MIT Kerberos 5

Kerberos es un protocolo de autenticación utilizado en redes no seguras para permitir autenticación recíproca entre un cliente y un servidor y se utiliza principalmente para centralizar la autenticación de usuarios y equipos en una red.

Kerberos proporciona lo que se denomina SSO (*Single Sign-On*), de manera que un usuario sólo tiene que autenticarse una vez en cada sesión, pudiendo servir esta autenticación para todos los servicios y equipos de la organización[7].

Existen diferentes implementaciones y versiones de Kerberos en Debian, siendo la versión recomendada la 5 y las implementaciones más conocidas la del *Massachusetts Institute of Technology* (MIT) y heimdal. En este caso vamos a utilizar MIT Kerberos 5, pero sería posible realizar el mismo proceso utilizando la implementación heimdal.

6.1. Defincione previas

Kerberos utiliza su propia terminología que es necesario conocer previamente:

- Realm (reino): Define el dominio de autenticación del que se hará cargo el servidor Kerberos. Para un realm se suele utilizar el dominio DNS de la organización, pero se pone en mayúsculas. En nuestro caso será *EXAMPLE.COM*
- Principal: Es una entrada en la base de datos Kerberos que puede incluir definiciones de usuarios, equipos o servicios entre otros, por ejemplo se definiría el principal *usuario@EXAMPLE.COM* para un usuario o *imap/correo.example.com@EXAMPLE.COM* para definir el servidor imap de la organización.
- Ticket: Los tickets son datos cifrados que el servidor Kerberos facilita a los clientes para su autenticación y que estos almacenan durante la sesión. Los principales tipos de tickets son:

  ***Ticket Granting Ticket* (TGT)** Ticket de autenticación de un usuario en la red y que se solicita al iniciar la sesión. Normalmente los TGT tienen una validez de 10 horas.

[7] Hay cierta controversia en esta denominación ya que hoy por hoy no es posible utilizar Kerberos en todos los servicios, por lo que hay quien prefiere denominar lo que hace Kerberos como *Reduced Sign-On*

***Ticket Granting Service* (TGS)** Ticket que solicita un usuario para autenticarse frente a un servidor que también esté en la base de datos de Kerberos.

- Key Distribution Center (KDC): Servidor Kerberos encargado de la autenticación, realmente compuesto por un *Authentication Server* encargado de repartir los TGT y un *Ticket Granting Server* encargado de hacer lo correspondiente con los TGS.
- kadmin-server: Servidor maestro de Kerberos, que se utiliza para administrar los principales.

Una descripción más detallada de todos los componentes se encuentra en [4].

6.2. Instalación de paquetes en el servidor Kerberos 5

El kdc y el kadmin-server no tienen que estar necesariamente en el mismo equipo, aunque en nuestro caso estarán ambos en *avatar*:

```
avatar:~# aptitude install krb5-kdc krb5-admin-server
```

Dependiendo de la configuración de *debconf*, habrá que contestar a varias cuestiones, en este caso nos pidió el nombre del servidor Kerberos y el servidor administrativo, que en ambos casos es *kerberos.example.com* (alias DNS de *avatar*). Al finalizar la instalación de estos paquetes obtenemos los siguientes mensajes:

```
krb5kdc: cannot initialize realm EXAMPLE.COM - see log file for details
...
kadmind: Cannot find/read stored master key while initializing, aborting
```

Porque no hay todavía un realm definido y no existe la clave maestra. Vamos a comprobar los parámetros que hay en los diferentes ficheros de configuración y a realizar unas pequeñas modificaciones para poder iniciar después los dos servicios.

/etc/krb5kdc/kdc.conf

```
[kdcdefaults]
    kdc_ports = 88

[realms]
    EXAMPLE.COM = {
        database_name = /var/lib/krb5kdc/principal
        admin_keytab = FILE:/etc/krb5kdc/kadm5.keytab
        acl_file = /etc/krb5kdc/kadm5.acl
        key_stash_file = /etc/krb5kdc/stash
        kdc_ports = 88
        max_life = 10h 0m 0s
        max_renewable_life = 7d 0h 0m 0s
        master_key_type = des3-hmac-sha1
        supported_enctypes = aes256-cts:normal arcfour-hmac:normal \
         des3-hmac-sha1:normal des-cbc-crc:normal des:normal des:v4\
         des:norealm des:onlyrealm des:afs3
        default_principal_flags = +preauth
    }
```

donde la única modificación hecha ha sido dejar sólo el puerto *88/udp* ya que el puerto *750/udp* se utiliza en Kerberos4.

/etc/default/krb5-kdc

```
KRB4_MODE=disable
```

```
2 RUN_KRB524D=false
```

Para deshabilitar por completo la utilización de Kerberos4.

/etc/krb5.conf

```
1  [libdefaults]
2          default_realm = EXAMPLE.COM
3  ...
4  [realms]
5          EXAMPLE.COM = {
6                  kdc = kerberos.example.com
7                  admin_server = kerberos.example.com
8          }
9  ...
10 [domain_realm]
11         .example.com = EXAMPLE.COM
12         example.com = EXAMPLE.COM
```

Que corresponde a la configuración del cliente Kerberos de *avatar*.

6.3. Puesta en marcha de los servicios

Para definir el realm de nuestro servidor Kerberos, ejecutaremos la siguiente instrucción[8]:

```
avatar:~# krb5_newrealm
```

que nos pedirá introducir la clave maestra de Kerberos y nos permitirá iniciar ya los dos servicios:

```
avatar:~# /etc/init.d/krb5-kdc start
avatar:~# /etc/init.d/krb5-admin-server start
```

Ahora podemos comprobar el funcionamiento de ambos servicios:

```
avatar:~# netstat -putan |grep "krb5kdc\|kadmind"
```

```
tcp   0   0   0.0.0.0:749          0.0.0.0:*     LISTEN      3105/kadmind
udp   0   0   0.0.0.0:464          0.0.0.0:*                 3105/kadmind
udp   0   0   192.168.2.1:88       0.0.0.0:*                 3101/krb5kdc
```

kdc escucha en el puerto `88/udp`, mientras que *kadmind* escucha en los puertos `749/tcp` para utilizar la aplicación *kadmin* y `464/udp` para *kpasswd*. Es también interesante notar que *kdc* sólo acepta peticiones a través de la dirección `192.168.2.1` por lo que no está accesible ni siquiera a través de *localhost*.

Ahora es posible realizar una conexión local con el servidor kadmin donde podemos crear o modificar los principales:

```
avatar:~# kadmin.local
```

```
Authenticating as principal root/admin@EXAMPLE.COM with password.
```

Vamos a comprobar los principales que se generan de forma automática al instalar el servidor kadmin:

```
kadmin.local:  list_principals
```

```
K/M@EXAMPLE.COM
```

[8] En el caso de utilizar una máquina virtual es muy recomendable la lectura de [5]

```
kadmin/admin@EXAMPLE.COM
kadmin/avatar.example.com@EXAMPLE.COM
kadmin/changepw@EXAMPLE.COM
kadmin/history@EXAMPLE.COM
krbtgt/EXAMPLE.COM@EXAMPLE.COM
```

Todos los usuarios, equipos y servicios que queramos que se validen contra Kerberos deber tener su principal en la base de datos de Kerberos. Los usuarios se autenticarán utilizando una contraseña y los equipos o servicios se crearán utilizando una contraseña aleatoria y almacenándola en un fichero *keytab*.

En nuestro caso vamos inicialmente a crear principales para el usuario *pruebau*, los equipos *avatar* y *cliente* y el servicio *ldap*, donde sólo el primero tendrá una clave propia y los otros se generarán con una clave aleatoria:

```
kadmin.local:  add_principal pruebau
kadmin.local:  add_principal -randkey host/avatar.example.com
kadmin.local:  add_principal -randkey host/cliente.example.com
kadmin.local:  add_principal -randkey ldap/avatar.example.com
```

6.4. Ficheros keytab

Los ficheros keytab contienen una lista de principales con sus correspondientes claves cifradas y se utilizan para autenticarse contra un servidor Kerberos de forma no interactiva. Estos ficheros son utilizados en procesos en los que no puede haber interacción de un usuario como la autenticación de servidores o la ejecución de tareas programadas. Un atacante que tuviese acceso a un fichero keytab podría autenticarse en la red con cualquiera de los principales que en el fichero hubiera, por lo que es realmente importante controlar el acceso a estos ficheros.

Para almacenar los principales de avatar y sus claves cifradas en */etc/krb5.keytab*, ejecutamos:

```
kadmin.local:  ktadd host/avatar.example.com
kadmin.local:  ktadd ldap/avatar.example.com
```

El contenido del fichero keytab puede consultarse y modificarse utilizando la instrucción *ktutil*:

```
avatar:~# ktutil
ktutil: rkt /etc/krb5.keytab
ktutil:  l
```

```
slot KVNO Principal
---- ---- ---------------------------------------------------------------------
   1    3             host/avatar.example.com@EXAMPLE.COM
   2    3             host/avatar.example.com@EXAMPLE.COM
   3    3             host/avatar.example.com@EXAMPLE.COM
   4    3             host/avatar.example.com@EXAMPLE.COM
   5    3             ldap/avatar.example.com@EXAMPLE.COM
   6    3             ldap/avatar.example.com@EXAMPLE.COM
   7    3             ldap/avatar.example.com@EXAMPLE.COM
   8    3             ldap/avatar.example.com@EXAMPLE.COM
```


6.5. Creación de usuarios para la administración de Kerberos

El acceso a *kadmind* está permitido para el usuario *root* de la máquina que ejecuta el servicio utilizando *kadmin.local*, dependiendo de la organización la administración de Kerberos puede

tener diferentes roles, aunque aquí sólo crearemos un rol de administración con todos los permisos.

Para crear un rol de administración con todos los permisos hay que descomentar la siguiente línea:

/etc/krb5kdc/kadm5.acl

```
6 */admin *
```

que le otorga todos los permisos a los principales del rol */admin*.

Accedemos a *kadmin.local* y creamos un principal para un usuario administrador:

```
kadmin.local:  add_principal jefe/admin
```

Ahora es posible acceder a la interfaz de administración de Kerberos desde otro equipo de la red:

```
cliente:~$ kadmin -p jefe/admin
Authenticating as principal jefe/admin with password.
Password for jefe/admin@EXAMPLE.COM: (se introduce la contraseña)
kadmin:
```

7. Configuración del cliente Kerberos

En todos los equipos de la organización hay que instalar inicialmente los paquetes:

```
cliente:~# aptitude install krb5-config krb5-user
```

y configurar el fichero */etc/krb5.conf* igual que el del servidor.

7.1. klist

Esta instrucción muestra los tickets de la sesión de usuario, si se ejecuta antes de autenticarse se obtiene esta salida (el parámetro *-5* es para que sólo utilice Kerberos 5):

```
cliente:~$ klist -5
```

```
klist: No credentials cache found (ticket cache FILE:/tmp/krb5cc_0)
```

7.2. kinit

Se utiliza para autenticarse de forma manual contra un servidor Kerberos definido en */etc/krb5.conf*, por ejemplo:

```
cliente:~$ kinit pruebau
Password for pruebau@EXAMPLE.COM: (se introduce la contraseña)
```

Ahora la salida de *klist -5* muestra lo siguiente:

```
Ticket cache: FILE:/tmp/krb5cc_0
Default principal: pruebau@EXAMPLE.COM

Valid starting     Expires            Service principal
01/07/10 19:55:21  01/08/10 05:55:21  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 01/08/10 19:55:20
```

Esta salida muestra que el usuario *pruebau* está autenticado correctamente contra Kerberos. Como anteriormente pudimos sacar toda la información de la cuenta del directorio LDAP, ahora sólo queda unirlo todo para que el acceso sea transparente al usuario.

8. SASL/GSSAPI

En las secciones anteriores, para conectarnos (*bind*) al directorio LDAP debíamos utilizar autenticación simple, de manera que en cada consulta o modificación había que incluir toda la información del usuario que se conectaba al LDAP. Sin embargo LDAP permite autenticarse utilizando *Simple Authentication and Security Layer* (SASL), que a su vez soporta autenticación Kerberos a través de *Generic Security Services Application Program Interface* (GSSAPI). Todo esto puede parecer muy complicado, pero vamos a resumir los pasos que pretendemos que dé el usuario para acceder a una cuenta en el sistema Kerberos/LDAP:

- El usuario introduce su nombre de usuario y contraseña que se corresponde con un principal de Kerberos. Si la contraseña es válida el usuario obtiene un TGT y está correctamente autenticado pero no tiene datos de la cuenta ni está autorizado en el sistema.
- Una vez que el usuario tiene el TGT solicita a kerberos el TGS de LDAP y realiza una consulta autenticada a LDAP (SASL/GSSAPI) de la que obtiene el resto de sus datos (home, shell, uid, ...) sin necesidad de volver a autenticarse contra el directorio.

Por tanto el primer paso es configurar LDAP para que se puedan realizar consultas autenticadas con SASL/GSSAPI, para lo que instalamos el módulo gssapi de sasl2:

```
avatar:~# aptitude install libsasl2-modules-gssapi-mit
```

Para que el servicio *slapd* pueda consultar el fichero keytab y autenticarse contra kerberos, es necesario modificar sus permisos y grupos, en el caso de una configuración sencilla como la nuestra se puede hacer:

```
avatar:~# chmod 640 /etc/krb5.keytab
avatar:~# chgrp openldap /etc/krb5.keytab
```

Editamos el fichero de configuración de slapd y añadimos las siguientes líneas:

/etc/ldap/slapd.conf

```
141 sasl-realm EXAMPLE.COM
142 authz-regexp
143         uid=([^,]*),cn=EXAMPLE.COM,cn=gssapi,cn=auth
144         uid=$1,ou=People,dc=example,dc=com
```

Creamos el siguiente fichero con la línea:

/etc/ldap/sasl2/slapd.conf

```
1 mech_list: GSSAPI
```

y reiniciamos el servidor ldap:

```
avatar:~# /etc/init.d/slapd restart
```

Para comprobar si está activado SASL/GSSAPI realizamos una consulta al directorio de los mecanimsos SASL soportados:

```
avatar:~# ldapsearch -x -b "" -s base -LLL supportedSASLMechanisms
```

```
dn:
supportedSASLMechanisms: CRAM-MD5
supportedSASLMechanisms: NTLM
supportedSASLMechanisms: GSSAPI <---
supportedSASLMechanisms: DIGEST-MD5
```

Ahora el usuario *pruebau*, una vez se haya autenticado contra Kerberos, puede hacer todas las consultas que desee al directorio sin necesidad de introducir la datos de autenticación en cada una de ellas:

```
cliente:~$ ldapsearch "gidNumber=2000"
```

```
SASL/GSSAPI authentication started
SASL username: pruebau@EXAMPLE.COM
SASL SSF: 56
SASL data security layer installed.
# extended LDIF
#
# LDAPv3
# base <dc=example,dc=com> (default) with scope subtree
# filter: gidNumber=2000
# requesting: ALL
#

# pruebag, Group, example.com
dn: cn=pruebag,ou=Group,dc=example,dc=com
objectClass: posixGroup
objectClass: top
cn: pruebag
gidNumber: 2000

# pruebau, People, example.com
dn: uid=pruebau,ou=People,dc=example,dc=com
objectClass: account
objectClass: top
objectClass: posixAccount
uid: pruebau
cn: pruebau
loginShell: /bin/bash
uidNumber: 2000
gidNumber: 2000
homeDirectory: /home/pruebau

# search result
search: 5
result: 0 Success

# numResponses: 3
# numEntries: 2
```

y para comprobar cómo el usuario identifica la entrada de LDAP que incluye sus datos:

```
cliente:~$ ldapwhoami
```

```
SASL/GSSAPI authentication started
SASL username: pruebau@EXAMPLE.COM
SASL SSF: 56
SASL data security layer installed.
```

```
dn:uid=pruebau,ou=people,dc=example,dc=com
```

9. PAM

Pluggable Authentication Module (PAM) es el sistema modular que se encarga de las tareas de autenticación en el sistema, cada aplicación que necesite comprobar la autenticación de usuarios tendrá que hacer uso de él y habitualmente tendrá un fichero de configuración en el directorio */etc/pam.d*; podemos ver el contenido de este directorio en un sistema muy simple:

```
chfn   common-account   common-password   cron    other   ssh   sudo
chsh   common-auth      common-session    login   passwd  su
```

En los cuatro ficheros *common-** se incluyen las directivas comunes para todas las aplicaciones, mientras que en el resto de ficheros se incluyen las directivas que son aplicables sólo a ese programa.

Para que el sistema pueda utilizar el sistema de autenticación de kerberos es necesario instalar la biblioteca de PAM para Kerberos en todos los equipos de la red:

```
avatar:~# aptitude install libpam-krb5
```

```
cliente:~# aptitude install libpam-krb5
```

Recomendación: Puesto que vamos a tocar los ficheros de autenticación del sistema es posible dejar el sistema inaccesible, por lo que es recomendable guardar una copia de todo el directorio */etc/pam.d*, por ejemplo haciendo:

```
cp -r /etc/pam.d /etc/pam.d.old
```

9.1. Modificación de los ficheros common-*

Vamos a permitir que todas las aplicaciones sean capaces de autenticar contra Kerberos, por lo que haremos modificaciones sólo en los anteriormente mencionados ficheros *common-** del directorio */etc/pam.d*. La configuración de estos ficheros puede incluir otras muchas opciones, aquí sólo mostramos la configuración mínima para poder realizar autenticación de usuarios locales y de usuarios de la base de datos de Kerberos:

/etc/pam.d/common-auth

```
10  auth  sufficient  pam_krb5.so minimum_uid=2000
11  auth  required    pam_unix.so try_first_pass nullok_secure
```

/etc/pam.d/common-session

```
9   session  optional  pam_krb5.so minimum_uid=2000
10  session  required  pam_unix.so
```

/etc/pam.d/common-account

```
9   account  sufficient  pam_krb5.so minimum_uid=2000
10  account  required  pam_unix.so
```

/etc/pam.d/common-password

```
24  password  sufficient  pam_krb5.so minimum_uid=2000
25  password  required    pam_unix.so nullok obscure min=4 max=8 md5
```

En todos los casos existe una línea para la biblioteca *pam_unix.so* que se utiliza para los usuarios locales y otra *pam_krb5.so* para los usuarios de red. Se ha añadido el parámetro opcional *minimum_uid=2000* para que no se intente utilizar Kerberos para usuarios con UID menor que este número.

9.2. Prueba de funcionamiento

Una vez configurado PAM en todos los equipos de la red, es posible utilizar los usuarios de Kerberos como usuarios de esos equipos. Supongamos un proceso de *login* con el usuario *pruebau*:

```
cliente login: pruebau
Password: (se introduce la contraseña de Kerberos del usuario)
```

```
pruebau@cliente:~$ klist -5
```

```
Ticket cache: FILE:/tmp/krb5cc_2000_Qhe5k6
Default principal: pruebau@EXAMPLE.COM

Valid starting     Expires            Service principal
01/21/10 10:58:00  01/21/10 20:58:00  krbtgt/EXAMPLE.COM@EXAMPLE.COM
        renew until 01/22/10 10:58:00
```

Si ese usuario cambia la contraseña, esta modificación debe hacerse sobre la base de datos de Kerberos:

```
pruebau@cliente:~$ passwd
Current Kerberos password: (se introduce la contraseña actual)
Enter new Kerberos password: (se intriduce la nueva contraseña)
Retype new Kerberos password: (se introduce la nueva contraseña)
```

```
passwd: contraseña actualizada correctamente
```

Una vez llegado a este punto un usuario de red que se entre en el sistema se está autenticando contra Kerberos, del que obtiene un TGT para la sesión. Utiliza este TGT para solicitar un TGS del servidor LDAP de la organización y obtener de él los datos de la cuenta, acción que realiza de forma segura y transparente gracias a GSSAPI. Aunque todo esto pueda parecer complicado, para el usuario no lo es, ya que simplemente constata que puede utilizar el mismo nombre de usuario y contraseña en todos los equipos de la red, sin necesidad de crear cuentas locales. Ahora sólo falta que tenga siempre sus ficheros disponibles, de lo que se encargará NFS y que veremos a continuación.

10. Network File System 4 (NFS4)

Uno de los problemas importantes que tenía la versión 3 de NFS era que no empleaba ningún método de autenticación de los clientes, simplemente se definía la dirección IP de los equipos que podían montar un directorio remoto mediante NFS lo que puede permitir usos ilegítimos del directorio compartido.

NFS4 soluciona esto de forma drástica ya que incluye soporte para Kerberos, de manera que sólo los clientes de la red que estén en la base de datos de kerberos y tengan acceso a la clave correspondiente (fichero keytab) podrán montar el directorio remoto.

10.1. Configuración del servidor

En nuestro caso, el equipo avatar va a funcionar como servidor NFS, por lo que tendremos que instalar el paquete correspondiente al servidor NFS:

```
avatar:~# aptitude install nfs-kernel-server
```

Que instala por dependencias el paquete *nfs-common*. Ahora tendremos que configurar ambos paquetes para que utilicen Kerberos a través de GSSAPI:

/etc/default/nfs-common

```
16 NEED_IDMAPD=yes
17
18 # Do you want to start the gssd daemon? It is required for Kerberos mounts.
19 NEED_GSSD=yes
```

/etc/default/nfs-kernel-server

```
17 NEED_SVCGSSD=yes
```

/etc/idmapd.conf

```
5 Domain = example.com
```

Es necesario definir principales en kerberos para cada equipo que va a utilizar NFS y exportar el fichero keytab correspondiente a cada equipo:

```
avatar:~# kadmin.local
kadmin.local:  add_principal -randkey nfs/avatar.example.com
kadmin.local:  add_principal -randkey nfs/cliente.example.com
kadmin.local:  ktadd nfs/avatar.example.com
kadmin.local:  ktadd -k /tmp/krb5.keytab nfs/cliente.example.com
```

Enviamos a cliente el fichero keytab de forma segura (por ejemplo utilizando ssh) y lo borramos:

```
avatar:~# scp /tmp/krb5.keytab cliente.example.com:/etc/krb5.keytab
avatar:~# rm /tmp/krb5.keytab
```

En nuestro caso vamos a utilizar NFS para exportar el directorio con las cuentas de los usuarios de red (/home/users). En NFS4 se define un directorio base a partir del cual se pueden exportar directorios que cuelguen de él y como es lógico este directorio no debe ser el directorio raiz. En nuestro caso vamos a utilizar el directorio /srv/nfs4 como directorio base para exportar y sobre él creamos y montamos los directorios a exportar:

```
avatar:~# mkdir -p /srv/nfs4/homes
avatar:~# mount --bind /home /srv/nfs4/homes
```

Para definir los directorios que se van a exportar con NFS, utilizamos como es habitual el fichero /etc/exports, aunque con algunos cambios en la sintaxis:

/etc/exports

```
8 /srv/nfs4         gss/krb5i(rw,sync,fsid=0,crossmnt,no_subtree_check)
9 /srv/nfs4/homes   gss/krb5i(rw,sync,no_subtree_check)
```


gss/krb5i El cliente debe estar autenticado mediante kerberos5 para poder montar el directorio

rw Modo lectura y escritura.

sync Montaje síncrono, de manera que cualquier modificación desde el cliente se transmite inmediatamente al servidor[9].

fsid=0 Define este directorio como la base de NFS4.

crossmnt Se monta este directorio al montar uno que cuelgue de él.

no_subtree_check Evita chequeos completos del árbol de directorios que se hacen para mantener la consistencia, pero que pueden ralentizar mucho el proceso.

Ahora es el momento de reiniciar los demonios de NFS:

```
avatar:~# /etc/init.d/nfs-common restart
```

```
Stopping NFS common utilities: gssd idmapd statd.
Starting NFS common utilities: statd idmapd gssd.
```

```
avatar:~# /etc/init.d/nfs-kernel-server restart
```

```
Stopping NFS kernel daemon: mountd svcgssd nfsd.
Unexporting directories for NFS kernel daemon....
Exporting directories for NFS kernel daemon....
Starting NFS kernel daemon: nfsd svcgssd mountd.
```

Se suele utilizar la instrucción *showmount* para mostrar los directorios exportados:

```
avatar:~# showmount -e
```

```
Export list for avatar:
/srv/nfs4       gss/krb5i
/srv/nfs4/homes gss/krb5i
```

10.2. Configuración del cliente

Hay que instalar y configurar el paquete *nfs-common*:

```
cliente:~# aptitude install nfs-common
```

/etc/default/nfs-common

```
16 NEED_IDMAPD=yes
17
18 # Do you want to start the gssd daemon? It is required for Kerberos mounts.
19 NEED_GSSD=yes
```

/etc/idmapd.conf

```
5 Domain = example.com
```

Y reiniciar el demonio:

```
cliente:~# /etc/init.d/nfs-common restart
```

Ahora podremos montar cualquiera de los directorios que ofrece avatar ya que tenemos acceso al fichero keytab que nos autentica frente a Kerberos. Como lo que nos interesa es montar el directorio con todas las cuentas de los usuarios:

```
cliente:~# mount -t nfs4 -sec=krb5i avatar.example.com:/homes /home
```

Para comprobar que se ha montado correctamente:

[9] Esta opción exige una buena comunicación entre el cliente y el servidor.

```
cliente:~# mount |grep nfs4

avatar.example.com:/homes on /home type nfs4 (rw,sec=krb5i,addr=192.168.2.1)
```

10.3. Modificaciones permanentes

Puesto que queremos tener un sistema centralizado de cuentas, el montaje de directorio se debe realizar de forma automática al iniciar el equipo, por lo que habrá que añadir las siguientes líneas a los ficheros */etc/fstab* de avatar y cliente:

/etc/fstab (avatar)

```
1 /home              /srv/nfs4/homes    none    rw,bind         0    0
```

/etc/fstab (cliente)

```
1 avatar.example.com:/homes    /home    nfs4    rw,sec=krb5i    0    0
```

Y como es lógico, será necesario arrancar el servidor antes que el cliente y que ambos estén interconectados.

Ya sólo falta reiniciar las dos máquinas de forma ordenada (primero el servidor y luego el cliente) y comprobar que la cuenta del usuario *pruebau* se puede utilizar de forma síncrona en los dos equipos.

Referencias

[1] D. Eastlake y A. Panitz. Request For Comments 2606.
http://www.rfc-editor.org/rfc/rfc2606.txt

[2] Bart Trojanowski. Ldap Authentication on Debian
http://www.jukie.net/~bart/ldap/ldap-authentication-on-debian/index.html

[3] Alberto Molina Coballes. Autenticación LDAP en GNU/Linux.
http://albertomolina.files.wordpress.com/2008/07/autenticacion_ldap.pdf

[4] MIT Kerberos Consortium. Kerberos Protocol Tutorial.
http://kerberos.org/software/tutorial.html

[5] Daniel Kahn Gillmor. Dealing with entropy on a virtual machine.
http://www.debian-administration.org/users/dkg/weblog/56

[6] Jean-Yves Migeon. The MIT Kerberos Administrator's How-to Guide
http://kerberos.org/software/adminkerberos.pdf

[7] Davor Ocelic. Debian GNU: Setting up MIT Kerberos 5
http://techpubs.spinlocksolutions.com/dklar/kerberos.html

[8] Crysol. En busca de la solución definitiva para la autenticación
http://crysol.org/es/node/743