Apéndice B. Kerberos

"El mejor profeta del futuro es el pasado."

Lord Byron

KDC

El KDC recibe requerimientos de tickets, si el usuario y el host que lo solicitan están en las bases de datos crea un ticket de acceso, lo encripta con la llave del usuario y lo envía de regreso. También se encarga de mantener las bases de datos, en el KDC se dan de alta, baja, y se modifica a los usuarios. Aquí se encuentra la llave maestra de la base de datos.

El KDC da el servicio de encriptamiento en general, que por red y gracias a un guardián especial (krb5kdc), los demás host clientes y usuarios pueden establecer sus sesiones. De ahora en adelante el servidor de Kerberos será llamado KDC. [BOR93]

Kerberos sigue de el protocolo de Needham y Schroeder [VIL00] con clave secreta, utilizando timestamps como pruebas de frescura con dos propó sitos: evitar reenvíos de viejos mensajes o reutilización de viejos tickets obtenidos [GAR91]

Los tickets que extiende el KDC tienen un tiempo de vida, de tal forma que si un atacante captura un ticket sólo tiene poco tiempo para ejercer su ataque, el tiempo de vida es especificado por el administrador, con base a políticas de seguridad.

De igual forma, Kerberos trabaja con Time Stamps, es decir que si un atacante captura un paquete de servicio, cuando intente hacer una requisición al servidor, el atacante necesitará enviar dicho paquete (el cual está encriptado con la llave de sesión que el KDC envía al cliente y al servidor), dentro del cual existe un Time Stamp, es decir, una estampa con la hora exacta en el cliente, que tiene que coincidir con la del servidor con un margen de error muy pequeño (calculado el tiempo de transmisión) de tal forma que si el atacante atrapó este paquete, el Time Stamp no coincidirá y el atacante no tiene forma de modificarlo, ya que no conoce la llave de sesión repartida al cliente y al servidor encriptada con sus respectivos passwords. Este sistema se estudiará a detalle más adelante [GAR91]

Realm

El KDC necesita un nombre, que para efectos de Kerberos, es el nombre del dominio que controla el conjunto de clientes y servidores. Ese dominio se llama Realm, y por lo general es el nombre del dominio de la red, pero escrito con mayúsculas para no tener ningún problema con el servidor de dominios. [GAR91]

Por ejemplo para la UDLA el dominio es pue.udlap.mx, entonces un nombre candidato para el realm puede ser UDLAP.MX.

Principals

Kerberos utiliza este término para cada entrada en la base de datos, ya sea un servidor, un cliente, usuario, etc.

Los principals pueden tener dos o tres elementos que son: nombre, instancia y realm.

Por ejemplo, un usuario administrador: sandra/admin@UDLA.MX.

Donde **sandra** es el usuario, **admin** es una característica del usuario que se utiliza para efectos de control, por ejemplo para darle permisos de modificación, creación y borrado en la base de datos a todos los principals que tengan la característica admin, por último UDLA.MX es el realm.

Cualquier usuario normal debe tener una entrada como la siguiente: sandra@UDLA.MX

Unicamente con el nombre de usuario y el realm.

Un principal para servidor debe tener la palabra host, el nombre del servidor y el nombre del realm.

Por ejemplo: host/udlakerb.pue.udlap.mx@UDLA.MX

Cada servicio debe tener una entrada en la base de datos, por ejemplo ftp:

ftp/udlakerb.pue.udlap.mx@UDLA.MX

Por último, existen dos principals por default que son **kadmin/admin** con el cual un cliente se autentifica con el servidor de Kerberos para poder tener acceso a la base de datos para modificarla y **kadmin/changepw** con el cual un cliente puede accesar para que un usuario cambie su password desde su propia máquina.

Permisos para principals

Existe un archivo en /usr/local/var/krb5kdc/kadm5.acl en donde cada entrada es un principal seguida de los permisos que tiene con respecto a la base de datos.

Por ejemplo:

sandra/admin lam

*/admin *

enrique lm

host/udlakerb.pue.udlap.mx@UDLA.MX *

Este archivo acepta cartas comodín como el *, que significa, todos los permisos, todos los usuarios o todas las instancias, según en donde esté colocado.

Del lado de los permisos una letra en mayúscula los quita y con una minúscula lo habilita.

Los permisos son los siguientes:

1. A,a Permiso para añadir principals.
2. D,d Permiso para borrar principals.
3. M,m Permiso para modificar principals.
4. C,c Permiso para cambiar passwords.
5. I,i Permiso para hacer peticiones a la base de datos.
6. L,l Permiso para listar principals.
7. Todos los permisos.
8. x Igual que *.

Para poder modificar la base de datos, nosotros entramos al proceso **kadmin** , el cual activa al programa o daemon kadmind en la máquina administradora (que se declara en el archivo **krb5.conf** sección REALM : admin_server), por lo cual, después de hacer modificaciones en el archivo **kadm5.acl** es necesario reinicializar dicho daemon.

El programa **kadmin** siempre aumenta la instancia /admin al usuario que lo invoca, por lo cual cualquier usuario que quiera accesar deberá tener un principal con la instancia /admin, de otra forma necesitará entrar con la opción −p principal.

Es posible también entrar con la autentificación hecha por el principal de la máquina, con lo cual se utilizará la opción:

−k /etc/krb5.keytab que indica la tabla donde se encuentra la llave del host. Con esta opción, kadmin no preguntará por un password para desencriptar la respuesta como lo haría con cualquier otro principal, debido a que la llave la busca en la tabla específica.

Los permisos para host se declaran también en el archivo **kadm5.acl,** como se vio en el ejemplo anterior con el host "host/udlakerb.pue.udlap.mx@UDLA.MX *".

Esto podría causar algunos problemas de seguridad (si se dan demasiados permisos para el host) en caso de que algún atacante tenga acceso a la máquina.

Organización de principals en el Realm

Cada usuario debe tener una entrada en la base de datos en el KDC, se puede dar de alta principals en el servidor mismo o en una máquina cliente, siempre y cuando el usuario que vaya a modificar la base de datos tenga permisos para hacerlo.

Para que un usuario pueda tener un ticket, éste tiene primero que estar dado de alta en la base de datos como un principal.

Para que un servidor pueda dar un servicio primero debe estar dado de alta el servidor como un principal de la siguiente forma:

host/servidor.dominio@Realm

Para que un servicio pueda ser prestado por un servidor tiene también que estar dado de alta de la siguiente forma:

servicio/servidor.dominio@Realm.

Para poder dar de alta un principal se ejecuta el programa /usr/local/bin/kadmin, después para dar de alta:

kadmin: addprinc sandra@UDLA.MX

kadmin: addprinc host/udlakerb.pue.udlap.mx

kadmin: addprinc ftp/udlakerb.pue.udlap.mx

Después de esto el usuario sandra puede tener derecho a un ticket, sin embargo, para que el servidor udlakerb.pue.udlap.mx pueda dar algún servicio necesita tener una tabla localmente con el nombre del servicio que presta y una llave con la que se va a autentificar en el KDC, el cual va a darle un ticket y una llave de sesión para que pueda comunicarse con el usuario.

La forma de crear la base de datos local para autentificarse es con el comando **/usr/local/sbin/kadmin** , ejecutado localmente en el host:

kadmin: ktadd host/udlakerb.pue.udlap.mx

kadmin: ktadd ftp/udlakerb.pue.udlap.mx.

Tickets del Servidor

Para poder accesar a un servicio de alguna otra máquina, el cliente lo solicita como si no existiera Kerberos, sin embargo, esta requisición llega al KDC [VIL00], el cual entonces envía una llave de sesión y un ticket para el servidor (que debe estar dado de alta en la base de datos), al cliente. Después el cliente envía una requisición al servidor para poder tener acceso al servicio, dicha requisición contiene el ticket encriptado con la llave del servicio/servidor (enviada anteriormente por el KDC al cliente). Cuando la requisición llega al servidor, éste la desencripta con la llave que se encuentra en la tabla de servicios local vista anteriormente, checa el ticket del cliente, que sea actual y verídico.

Después, si es correcta la información, el servidor regresa al cliente una respuesta de aceptación del servicio encriptada con la llave de sesión. De esta forma el usuario "no tiene que volver a introducir su username y su password, evitando que éste viaje sobre la red". [GAR91]

.k5login

En ocasiones es necesario dar permisos a alguna otra persona para que entre a alguna cuenta, en una situación común sería necesario darle el password a la otra persona, sin embargo, con Kerberos existe el archivo .k5login, dentro del cual únicamente necesitamos incluir a los usuarios a los que vamos a permitir la entrada a nuestra cuenta. Estos usuarios únicamente tendrán que hacer un rlogin o telnet a dicha cuenta, o bien un ksu, de la siguiente forma:

acadaplic:enrique > more .k5login

enrique@UDLAP.MX

sandra@UDLAP.MX

Por telnet:

Aquí enrique le da permiso a sandra de entrar a su cuenta

lara : sandra > kinit

Password for sandra@UDLAP.MX:

lara : sandra > telnet −l enrique acadaplic

Trying 140.148.155.177...

Connected to acadaplic (140.148.155.177).

Escape character is ´^]´.

[Kerberos V5 accepts you as "sandra@UDLAP.MX"]

Last login: Thu Apr 5 18:28:58 from lara

Sun Microsystems Inc. SunOS 5.6 Generic August 1997

acadaplic:enrique >

O bien en el mismo servidor, cambiando su UID: sandra adquiere la identidad de enrique

acadaplic:sandra > ksu enrique

Authenticated sandra@UDLAP.MX

Account enrique: authorization for sandra@UDLAP.MX successful

Changing uid to enrique (109)

acadaplic:enrique >

En estos ejemplos el usuario sandra no introdujo ningún login o password para accesar a la cuenta enrique, debido a la presencia del archivo .k5login en la cuenta enrique.

Es necesario si se tiene la presencia de este archivo, incluir en él al dueño de la cuenta, de otra forma cuando quiera entrar a su cuenta con telnet o rlogin, tendrá que reteclear su password, lo cual es ineficiente y molesto.

Otra ventaja muy importante de este archivo es que se puede incluir en el directorio raíz, y de esta forma limitar a los usuarios que pueden hacerse root a sólo los que están dentro de este archivo.

Por otro lado y mucho más importante es que dichos usuarios no tendrán que teclear el password de root, lo cual es de gran importancia, porque ni siquiera tendrán que saberlo y podrán conectarse a root remotamente (con telnet o rlogin) sin el temor de que el password viaje sobre la red, y con la seguridad de tener una comunicación encriptada. [GAR91]

Rsh

Remote Shell normal (sin Kerberos) puede estar desactivado o activado por medio del archivo **host.equiv o .rhost** , para un determinado cliente o conjunto de clientes, sin embargo, con rsh Kerberos es distinto. Mientras un cliente (sin ticket de Kerberos) no puede entrar con rsh normal a un servidor (ya que es inseguro habilitar este servicio de forma natural), si el mismo cliente tiene un ticket de acceso Kerberos, y el servidor al cual quiere accesar está dado de alta en la base de datos Kerberos (en forma de principal), la entrada del cliente es transparente (con su mismo UID en la máquina remota), y la comunicación estará segura ya que será encriptada y desencriptada sólo en el cliente y en el servidor. Esto es seguro ya que con Kerberos se tiene un ambiente de autentificación y de confianza dentro del realm, en donde la comunicación es confiable, al igual que los clientes, siempre y cuando estos tengan un ticket adecuado [GAR91].

Rlogin

Hay un dato importante que mencionar para rlogin de Kerberos, y es que no es necesario que el cliente esté dado de alta en el archivo .rhost o host.equiv, archivos que sirven para controlar el acceso en un escenario normal (sin Kerberos). Si el cliente cuenta con un ticket de autentificación Kerberos y el servidor está dado de alta en la base de datos de Kerberos la entrada es transparente, encriptada e inmediata, de tal forma que el usuario no se da cuenta de que existe Kerberos, y el administrador no tiene que preocuparse por los agujeros de seguridad que significan tener estos archivos (.rhost y host.equiv) ya que "no son necesarios" para tener una comunicación eficiente y transparente dentro de la red, quedando protegida de ataques exteriores e interiores, de personas no autorizadas y crackers [GAR91].

Por otro lado la transmisión como en todos los demás servicios es encriptada, viajando segura, y sin necesidad de introducir password.

Ksu, Kpasswd

Kerberos cuenta con sus propios programas para la administración como son ksu, que es el análogo de su, solamente que éste toma en cuenta el archivo .k5login para cambiar el UID de un usuario a otro, y principalmente antes de hacer algún cambio de UID verifica que el usuario que lo solicita tenga su correspondiente ticket de acceso.

Kpasswd por otro lado cambia el password de un principal en la base de datos de Kerberos en lugar de en el archivo /etc/passwd [GAR91].

Funcionamiento técnico

Para aclarar el funcionamiento y especificar los detalles de seguridad, es necesario hacer algunas descripciones técnicas de como funciona internamente Kerberos [VIL00].

La comunicación se da en tres pasos:

- Comunicación del cliente con el KDC para requerir ticket de acceso (Servidor de Autentificación). Respuesta del KDC al cliente enviando el ticket de acceso.
- Comunicación del cliente con el KDC para requerir ticket de autorización de servicio. Respuesta del KDC enviando ticket de autorización de servicio.

- ♦ Comunicación del cliente con el servidor para requerir aplicación(telnet, ftp, rlogin, etc.) Respuesta del servidor con la aplicación.

Que hablando técnicamente se traducen en:

1. : KRB_AS_REQ : Autentication Server Request
2. : KRB_AS_REP : Autentication Server Reply
3. : KRB_TGS_REQ : Ticket Granting Service Request
4. : KRB_TGS_REP : Ticket Granting Service Reply
5. : KRB_AP_REQ : Aplicaction Request
6. : KRB_AP_REP : Aplication Reply

Enseguida se presenta una tabla para analizar los detalles de comunicación entre el servidor de Kerberos, el cliente y el servidor de aplicaciones.

1. : C −> AS : U, TGS, L1 , N1
2. : AS −> C : U, TGT, [TGS, K, T1, T2, N1]Ku
3. : C −> TGS : S, L2, N2, TGT, Atgs
4. : TGS −> C : U, TS, [S, K1, T11, T21, N2]K
5. : C −> S : TS, Ac
6. : S −> C : [T1]K1

Tabla. Comunicación : cliente (C) − KDC (AS, TGS) y cliente (C) − servidor (S).

3. **AS** se refiere al Servidor de Autentificación, que es el que da los tickets de acceso a un usuario para que pueda solicitar servicios encriptados como telnet, ftp, etc. Y principalmente para que dicho usuario pueda entrar al sistema.

4. **TGS** es el servidor que da los tickets para que cierto usuario que requiera un servicio y presentando su respectivo ticket de acceso pueda tener dicho servicio seguro y encriptado.

5. **AS y TGS** se encuentran juntos en el **KDC** , aunque podrían aparecer separados.

6. **U** es el identificador del usuario.

7. **C** será tratado como el Cliente, máquina del usuario.

8. **S** será el servidor de las aplicaciones.

Integrantes del sistema Kerberos

Es necesario hacer la aclaración de que "S" es un principal dado de alta en la base de datos de la forma servicio/servidor.dominio@REALM, es decir es el servidor, pero contiene también el servicio requerido.

Por ejemplo ftp/udlakerb.pue.udlap.mx@UDLA.MX.

Los servicios de telnet, rsh, rlogin se hacen automáticamente cuando se da de alta el host con el principal de forma host/servidor.dominio@REALM.

Por ejemplo host/udlakerb.pue.udlap.mx@UDLA.MX, con lo cual quedarían dados de alta los servicios telnet, rlogin, rsh en la máquina udlakerb.

Otros servicios de rpc pueden ser dados de alta con el nombre del servicio en lugar de host como se vio anteriormente.

Un paquete entre corchetes "[XXX] W" significa que lo que está en XXX será encriptado con la llave W.

TGT es el ticket de acceso que se da al usuario.

Ts es el ticket de acceso a un servicio/servidor que se da al usuario, encriptado con la llave del servicio/servidor.

En el **paso 1** , "C" selecciona un TGS y envía un mensaje KRB_AS_REQ al AS. El mensaje contiene el identificador de usuario "U", el identificador del TGS (realm seleccionado), un tiempo de vida deseado L1, y un número N1 llamado en ingles nonce, este número es generado aleatoriamente por el cliente y sirve para evitar que algún atacante que haya capturado un paquete respuesta (KRB_AS_REP) pueda generar algún problema, ya que en la respuesta, el AS envía dicho número junto con el ticket, todo encriptado con la llave del usuario (password), de tal forma que el usuario la desencripta y tiene que coincidir el número que el envió (nonce) con el que recibe, asegurando de esta manera la autenticidad del ticket que recibe.

Después de recibir el KRB_AS_REQ, el AS verifica en su base de datos al usuario y al TGS, si alguno no existe envía un mensaje de error al cliente, si existen extrae las llaves del usuario y del TGS.

En el **paso 2** , el AS envía un mensaje KRB_AS_REP, que contiene el identificador del usuario "U", el ticket de acceso TGT, que es un paquete que contiene U, C, TGS, K, T1, T2, todo esto encriptado con la llave del TGS. "K" es la llave de sesión, T1 es el tiempo de creación del ticket, T2 es el tiempo de expiración del ticket.

También envía un paquete encriptado con la llave del usuario, el paquete contiene el TGS, K , T1, T2, N1. Después de recibir la respuesta, el cliente aplica una función al password que provee el usuario (el resultado de esta función es la llave del usuario) para desencriptar el segundo paquete, de donde obtiene la llave de sesión, el número nonce N1 para verificar la autenticidad de la respuesta, el tiempo de expiración, etc. En este momento el cliente tiene en su poder el TGT que lo puede usar hasta la fecha de expiración, la llave de sesión, y está listo para requerir un ticket de servicio al TGS [GAR91].

Cuando el cliente necesita un servicio (telnet, ftp, etc.) éste envía un KRB_TGS_REQ, lo cual pasa desapercibido por el usuario. En esta requisición (paso 3) el cliente envía S que es el servidor requerido, L2 tiempo de vida del ticket de servicio (TS), N2 (nonce 2), el TGT (su ticket de acceso al sistema recibido del AS), y un paquete más, encriptado con la llave de sesión (recibida del AS) el cual contiene "C" (el cliente) y Tstamp que es un Time Stamp con la hora exacta de requisición (en segundos desde el 1 de Enero de 1970) [GAR91], esto nuevamente para evitar ataques por captura de tickets los cuales obviamente tendrán distinto Tstamp cuando sea hecha la requisición promiscua.

El TGS revisa la requisición verificando que el servidor esté en la base de datos, para posteriormente extraer la llave del servidor (Ks). Desencripta el ticket TGT del cliente encriptado desde el AS con la llave del TGS, extrae de ahí la llave de sesión para poder desencriptar el paquete con el Tstamp, y sigue al cuarto paso, en el cual envía un

KRB_TGS_REP, que contiene "U", TS que es un ticket de permiso para el servidor, y un último paquete encriptado con la llave de sesión que contiene a "S", K1 que es una nueva llave de sesión para el servicio requerido, T11, T12 (tiempo de creación y expiración del Ts), y N2. El TS contiene a U, C, S, K1, T11, T12 encriptado con la llave del servidor (Ks).

El siguiente **paso (5**) es requerir el servicio, el cliente envía un KRB_AP_REQ, que contiene el TS (ticket de acceso al servidor), y un paquete encriptado con la llave de sesión K1, que contiene a "C" y un Tstamp1 (Time Stamp al momento de la requisición). El servidor autentifica al cliente dependiendo del Tstamp, del ticket TS que el mismo desencripta con su llave Ks, y del cliente mismo. El servidor envía un KRB_AP_REP en el cual solamente envía Tstamp1 encriptado con la llave de sesión para que el cliente esté seguro de que está hablando con el servidor auténtico.

Todos estos pasos no son sino un modelo de autentificación hasta la fecha totalmente eficiente, dentro del cual, se envían en los requerimientos, la información para autentificar al cliente; el servidor revisa sus datos y regresa un mensaje con la autorización y tickets de acceso. En el siguiente paso de autentificación, el servidor envía información dentro de los mensajes que lo autentifica con el cliente, para que este último esté seguro de que está comunicándose auténticamente con quien él desea (Time Stamps y números nonce), todo lo anterior encriptado con llave de sesión y dichas llaves de sesión encriptadas con llaves de cliente o servidor como ya se explicó anteriormente, permitiendo una comunicación totalmente segura y auténtica [VIL00].

Slave KDC

Los Slave KDC, que son servidores de respaldo que contienen la base de datos, y constantemente se están actualizando (esto para caso de recuperación en caso de daños)

Estos servidores de preferencia hay que tenerlos en diferentes lugares físicos del KDC maestro, tiene que ser también un lugar seguro para evitar que cualquier persona tenga acceso a los mismos y en consecuencia a la base de datos.

Para echar a andar estos servidores es necesario configurar el archivo /etc/inetd.conf añadiendo una línea para el programa kpropd:

kprop stream tcp nowait root /usr/local/sbin/kpropd kpropd

Esto en todos los servidores KDC, incluyendo el KDC maestro. Después es necesario dar de alta los servidores como un host/servidor.dominio@REALM, para todos los KDC incluyendo el maestro, es decir si el KDC principal es udlakerb, y el Slave KDC es solar18, se necesitan tener las siguientes entradas en la base de datos (con el comando addprinc del programa kadmin, ya visto anteriormente) [GAR91].

addprinc −randkey host/udlakerb.pue.udlap.mx@UDLA.MX

addprinc −randkey host/solar18.pue.udlap.mx@UDLA.MX

Se necesita crear el archivo:

/usr/local/var/krb5kdc/kpropd.acl

Este va a contener los servidores a los cuales se puede propagar la base de datos, esto suena razonable, ya que otro servidor cracker podría solicitar dicho servicio. El archivo debe contener según el ejemplo anterior, lo siguiente:

host/udlakerb.pue.udlap.mx@UDLA.MX

host/solar18.pue.udlap.mx@UDLA.MX

Propagación de la base de datos

Primero se compacta la base de datos en el KDC maestro con el comando:

/usr/local/sbinkdb5_util dump /usr/local/var/krb5kdc/base_dat

Segundo, crear un archivo stash en cada KDC, este archivo contiene la llave maestra encriptada y se encuentra por default en:

/usr/local/var/krb5kdc/.k5.REALM

Con esta llave podrá autentificar un servidor en otro, para que el último le preste un servicio al primero, de otra forma el servicio de transmisión de la base datos será rechazado.

El último paso será (en el KDC maestro) enviar la base de datos, con el comando:

/usr/local/sbin/kprop −f /usr/local/var/krb5kdc/base_datos solar18.pue.udlap.mx

Por último hay que echar a andar el programa krb5kdc en cada KDC.

Interacción KDC, Slave KDC

Si llegara a fallar el KDC principal, el cliente comienza a buscar autentificación en los KDC secundarios, los cuales se encuentran declarados en el archivo /etc/krb5.conf.

Para deshabilitar por completo el KDC maestro, es necesario matar los procesos kadmind y krb5kdc.

En el nuevo KDC se echará a correr el programa kadmind, (el programa krb5kdc ya lo debería tener corriendo desde que se dio de alta como Slave KDC), se creará una base de datos kadmin y por último se modificará el archivo /etc/krb5.conf cambiando el default KDC antiguo substituyéndolo por el nuevo.

En ese momento se puede apagar, modificar o componer, el antiguo KDC maestro.

krb5.conf, kdc.conf

Es mucha la importancia del archivo /etc/krb5.conf ya que es aquí en donde se registran los valores defaults de Kerberos, este archivo debe estar en todas las máquinas cliente, ya que es aquí en donde se distribuye y se asignan los realms para cada máquina, es decir, a qué máquina tiene que hacer la requisición de los tickets de acceso, al igual que los defaults para las aplicaciones, etc.

El archivo cuenta con las siguientes secciones [TOM94]:

- **libdefaults:** Contiene los valores por default que necesitan las librerías, como el realm default, el tiempo de vida default que asigna kinit a cada usuario, etc.
- **appdefaults:** Los valores default para las aplicaciones como telnet, ftp, kinit, etc.
- realms: Sección muy importante, ya que aquí se declaran los KDC para cada realm en donde debe buscar un cliente.
- **domain realm** : En esta sección se declara cómo se llama el realm para cada dominio, es decir se puede establecer un realm diferente para cada dominio. Y después establecer un distinto KDC para cada realm en la sección realms.
- **loggin:** Aquí se encuentran las direcciones de los archivos en donde se va a registrar las acciones de Kerberos y los resultados, para poder detectar causas de errores, ataques, problemas, etc.
- **capaths:** Cuando el realm no lo puede alcanzar directamente el cliente, en esta sección se establece el camino que debe seguir el request de autentificación para llegar al realm final en donde será autentificado para requerir determinado servicio remoto.
- **kdc:** Contiene el path del archivo kdc.conf.

El archivo /usr/local/var/krb5kdc/kdc.conf contiene información de configuración para el KDC maestro, como por ejemplo, qué puertos utilizará cada realm para transmitir la información de administración y en qué puerto autentificarán los clientes en el servidor.

Este archivo contiene tres partes:

- **kdcdefaults:** Aquí se introducen los números de puertos del servidor en donde autentificarán los clientes.
- **realms:** Contiene valores por default para cada realm. Por ejemplo, el archivo en donde se encuentra la base de datos, cuáles serán los puertos del servicio de administración, qué tipo de encriptamiento tendrán los
- servicios, el nombre del archivo de la base de datos para la administración, la localización del archivo kadmin.acl, etc.
- **loggin:** Es el archivo donde se registrarán las acciones del KDC.

En caso de dejar la máquina sola, esta deberá:

9 .Eliminar el ticket antes de dejar la maquina con el comando kdestroy.

mailweb:sandra > kdestroy

mailweb:sandra > klist

klist: No credentials cache file found while setting cache flags (ticket cache /tmp/krb5cc_241)

mailweb:sandra >

NOTA: Para una mayor profundización en la utilización e instalación de estos archivos se debe consultar los manuales de instalación y de administración que vienen con los archivos fuentes de Kerberos.

Políticas

Como en cualquier organización la administración necesita políticas que todos deben cumplir, ya sea por grupo o individualmente.

Kerberos cuenta con políticas para los passwords y para el manejo en general de cada usuario.

Es posible definir políticas por medio de la administración, dichas políticas después se seleccionan y se les asignan las adecuadas a cada principal. Es posible también crear una política default, la cual se va a asignar automáticamente a cada principal cuando sea creado.

Las políticas se crean, destruyen, asignan y modifican con el programa kadmin. Los comandos disponibles son los siguientes [TOM94] :

kadmin: addpol [opciones] usuarios

kadmin: delpol default

kadmin: modpol [opciones] politica3

kadmin: modprinc −policy usuarios

Las opciones pueden ser [TOM94] : maxlife, minlife, minlength, minclasses, history.

La opción minclasses crea una política interesante, debido a que cuando un usuario desea cambiar su password, el nuevo password necesita tener el número de clases de caracteres declarado en minclasses, como mínimo. Los tipos de

clases son los siguientes: minúsculas, mayúsculas, números, signos de puntuación y otros caracteres.

Importante: Con este tipo de políticas se resuelve el problema de la mayoría de los administradores, que es cuando los usuarios desean cambiar su password por algunos fáciles de recordar, pero también de crackear.

Existe también un archivo llamado /usr/local/var/krb5kdc/kadm5.dict (que se declara en kdc.conf), el cual es un diccionario que contiene todas las palabras prohibidas como passwords. Es decir que aun si el usuario respeta la política al escoger su nuevo password,

pero éste se encuentra en el archivo kadm5.dict, el password es rechazado.

Esto le da al administrador un ambiente de tranquilidad y de confianza en los usuarios, y a su vez, le abre las puertas al usuario para que sea capaz de realizar actividades propias del mismo, reduciendo el trabajo y la intranquilidad del administrador.

Problemas

En un principio se tiene un problema poco grave si se sabe controlar, y es que Kerberos tiene su propia tabla de passwords, entonces cuando un usuario intenta entrar, ya que no entra directamente por la tabla de passwords de la máquina sino vía Kerberos, este último necesita autentificar en la máquina, y lo hace como root, por lo cual necesita tener permisos de root sobre los filesystems en donde se encuentre el directorio de los usuarios, para después cambiar su UID de root al del usuario, de tal forma que para entrar a una máquina, Kerberos

no utiliza los passwords del archivo /etc/passwd, ni /etc/shadow.

Sin embargo, cuando el filesystem del usuario está montado por NFS la maquina cliente necesita tener permisos de root sobre dicho filesystem dejando desprotegido dicho filesystem, en caso de que un intruso se apodere de dicha máquina cliente lo cual es peligroso porque el atacante estaría también como root dentro del filesystem en la maquina servidor de NFS.

Para ejecutar el comando Kadmin para dar de alta o baja usuarios es necesario ejecutarlo como root porque de otra forma no permite modificaciones a las tablas, lo cual no debería ser, ya que un usuario que tiene permiso por Kerberos de modificar la base de datos, no lo podría hacer.

/etc/host necesita tener la entrada completa de cada servidor de telnet, ftp, rpc, etc. con el IP y principalmente el nombre del host con el dominio.

Es decir:

140.148.5.22 lara.pue.udlap.mx

Si esto no sucede ftp para Kerberos, así como kprop para la propagación de la base de datos en un Slave KDC, no autentifican, haciendo que el servicio falle, esto sucede en las redes en donde se tiene el servicio DNS, ya que éste posiblemente no provea el dominio completo.

Con respecto a este servicio, es necesario que el archivo /etc/nsswitch.conf contenga la siguiente entrada [GAR91]:

host: file dns

Si existe el archivo .k5login en alguna cuenta o en el directorio raíz, es necesario antes que incluir a otros usuarios, introducir el principal del dueño de la cuenta, de otra forma su entrada por medio de telnet, ksu o rlogin no será transparente.

Otro de los principales problemas consiste en tener el código fuente de cualquier programa que se desee "kerberizar" . Esto implica un gran esfuerzo en tiempo y operaciones complejas que en ocasiones no se está dispuesto a dar. Este es un problema de implementación [VIL00]. Un problema que sí está relacionado con la seguridad es la centralización que presenta el sistema, esto es, si el servidor de Kerberos falla, la red es inutilizable (contradictorio con la teoría de sistemas distribuidos [TOM94].

Un problema mas: Debido a los timestamps se obliga a todas las máquinas que ejecutan servicios autentificados a tener los relojes mínimamente sincronizados.

Estos pequeños problemas han propiciado que Kerberos no esté muy difundido, aún así se está trabajando muy fuerte para solucionarlos.

Pasos para usar el instalador de Kerberos

1. En acadaplic en /usr/kerberos ya puse un archivo llamado kerberos.tar, ese llevarselo por ftp a la maquina que necesita kerberos bajo /usr/kerberos de tal forma que exista el archivo /usr/kerberos/kerberos.tar

2. En la maquina destino teclear:
   # cd /usr/kerberos
   # tar xvf kerberos.tar
   # chmod 700 make−kerb
   # ./make−kerb
   Si es la primera vez que se le instala kerberos a la maquina contestar a la pregunta de "instalar servicios de inetd.conf" afirmativamente. Si ya tenia kerberos pero estaba mal primero elimine cualquier entrada de servicios de kerberos en /etc/inetd.conf y en /etc/services y contestar afirmativamente.
3. Revivir los procesos haciendo un kill −HUP al inetd
   # ps −ef|grep inetd
   # kill −HUP <numero_de_proceso>

índice figuras introducción 1 2 3 4 5 A B C D referencias