UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

ESCUELA NACIONAL DE ESTUDIOS PROFESIONALES
CAMPUS ARAGÓN

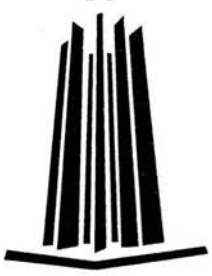
"SERVICIO DE CORREO ELECTRÓNICO CON LDAP Y QMAIL"

T E S I S

QUE PARA OBTENER EL TÍTULO DE:
INGENIERO EN COMPUTACIÓN
P R E S E N T A :

ANDRÉS ARMANDO CASTILLO PAVÍA

ASESOR:
DR. ENRIQUE DALTABUIT GODAS



SAN JUAN DE ARAGÓN, ESTADO DE MÉXICO MAYO 2004

SERVICIO DE CORREO ELECTRÓNICO CON LDAP Y QMAIL

ÍNDICE

CAPÍTULO 4. INSTALACIÓN Y FUNCIONAMIENTO DE LDAP CON QMAIL

Lista de Figuras

Lista de Tablas

HIPÓTESIS

LDAP Y QMAIL SON UNA EXCELENTE OPCIÓN EN SISTEMAS QUE FUNCIONAN COMO SERVIDORES DE CORREO ELECTRÓNICO, SU ALTA SEGURIDAD EN EL ENVIÓ Y RECEPCIÓN DE CORREO, ASI COMO SU FÁCIL CONFIGURACIÓN, ORGANIZACIÓN Y ADMINISTRACIÓN DE USUARIOS LOS HACEN HOY EN DÍA LA ELECCIÓN QUE SUSTITUIRÁ A OTROS SERVIDORES DE CORREO COMO SENDMAIL.

INTRODUCCIÓN

Al crecer la aceptación de las computadoras personales y diseminarse entre las organizaciones, éstas se vieron beneficiadas gracias a los constantes adelantos en la tecnología, obteniendo de ello servicios para almacenar, trasportar y procesar grandes cantidades de información.

A diferencia del correo postal, el correo electrónico se envía a través de una red de computadoras, hasta la computadora donde reside el buzón de la persona a quién va dirigido.

El correo electrónico ha sido y sigue siendo el recurso más utilizado en el mundo de las comunicaciones por su sencillez, agilidad y versatilidad. Con una conveniente interfaz y una dirección de correo electrónico tienes a tu alcance el envió de mensajes de una forma rápida y efectiva a cualquier parte del mundo. Una vez que el mensaje ha alcanzado el destino, el receptor puede leerlo en cualquier momento con su propia interfaz.

El uso del correo electrónico para las comunicaciones entre usuarios distantes es más cómoda, rápida y barata que sistema tradicionales como el correo postal o el teléfono. Mas aún, cuando se trata de comunicaciones transnacionales. Mientras que una carta puede tardar días o semanas en llegar de México a otro país, un mail suele llegar en minutos.

Para que una persona pueda utilizar el servicio de correo electrónico debe disponer de un buzón (mailbox) donde recibir los mensajes. Cada buzón va ligado a una dirección electrónica que deberá ser conocida por la persona que envía el mensaje.

El buzón de correo no es más que un recipiente donde el sistema va depositando cada uno de los mensajes destinados al propietario. Este recipiente esta protegido mediante una contraseña. Solo la persona o personas que la conozcan, podrán accesar a su contenido.

Cuando el propietario del buzón desea consultarlo, debe conectarse con la computadora donde reside utilizando su identificador y su contraseña. Normalmente se utiliza una aplicación que le permite ir leyendo y borrando cada uno de los mensajes. Debemos ser conscientes de que cualquier persona que conozca el identificador y la contraseña tendrá acceso al contenido del buzón.

La dirección electrónica del propietario de un buzón se compone, normalmente, del identificador del usuario, seguido del carácter @ (arroba) y el nombre de la computadora donde reside el buzón del usuario. Así, el usuario cuyo identificador es rodrigo y cuyo buzón se encuentra en la computadora servidor.unam.mx, su correo electrónico será rodrigo@servidor.unam.mx. Cualquier mensaje destinado a esta dirección será depositado en el buzón del usuario rodrigo.

El buzón de correo es el destino final donde queda depositado el mensaje al llegar a su destino. A partir de ese momento, el usuario es el responsable de consultarlo periódicamente y de borrar aquellos mensajes que ya hayan sido leídos.

Los buzones de correo pueden tener diferentes formatos, el más común es el formato mbox. Las características, en este tipo de archivos, están directamente ligadas al MTA (Agente de Transporte de Mail) con el que estén trabajando.

Actualmente se han impuesto los modelos cliente/servidor para el acceso al buzón de correo. En la máquina donde residen los buzones (servidor), se ejecuta un programa que atiende las peticiones realizadas por las aplicaciones que utilizan los usuarios para consultar su buzón (cliente). Para ello, el usuario debe utilizar un cliente capaz de dialogar con el servidor. Los

términos en los que se realiza este diálogo están impuestos por el protocolo utilizado. Si el servidor utiliza el protocolo IMAP (Protocolo de Acceso a Mensajes de Internet), el programa cliente debe estar configurado con IMAP.

La computadora donde ejecutemos el cliente debe estar conectada a la red. Tanto IMAP como POP (Protocolo de oficina Postal) son protocolos definidos para una red TCP/IP (Protocolo de Control de Transmisión-Protocolo de Internet) y por tanto la computadora debe estar conectada a una red de este tipo. La localización física del servidor es independiente de la del cliente. Podemos estar en otro país y accesar a nuestro buzón en México y viceversa.

Los dos protocolos mas extendidos para el acceso al buzón de correo en TCP/IP son POP e IMAP. Ambos sirven para lo mismo, pero entre ellos existen diferencias que aconsejan el uso de uno u otro según las particularidades del usuario.

Entre los clientes de correo que utilizan POP tenemos algunos de los más conocidos: Eudora, Netscape, Microsoft Mail, Pegasus, etc. Tienen una interfaz sencilla y agradable y cuentan con un conjunto de características que facilitan la recepción, envío y clasificación de mensajes.

Existen varios clientes basados en protocolos IMAP, aunque están mas extendidos los basados en POP.

CAPÍTULO 1

Principales diferencias entre los servidores de Correo Sendmail y Qmail

En este capítulo se explican las funciones que tienen el MDA, MUA y MTA como agentes de correo electrónico.
Se expone la problemática que presenta el buzón de correo mbox, que utiliza sendmail, así como la solución que nos da qmail a través de maildir.
Nos muestra la diferencia entre sendmail y qmail para agregar usuarios, aspectos de seguridad y otras diferencias entre ambas herramientas.

1.1 Servidores de correo electrónico

Probablemente los servidores de correo sean el servicio más extendido en Internet. Siempre hay un camino sencillo y barato para comunicarse por e-mail. Este servicio permite el intercambio de mensajes electrónicos con cualquier usuario de Internet (y de otras redes). Existen extensiones multimedia (MIME) que permiten enviar cualquier tipo de información asociada a un mensaje: datos, texto, sonido, imágenes, etc.

Los tres conceptos principales para el transporte y administración de correo electrónico que se deben conocer son: el agente de usuario de correo (MUA), el agente de transporte de correo (MTA) y el agente de entrega de correo (MDA).

Un MUA es la interfaz de usuario; el software que utilizan los usuarios para leer, organizar y enviar el correo. Diferentes MUA pueden coexistir en la misma máquina, por ejemplo: mailx, elm, pine, mailtool, netscape entre otros.

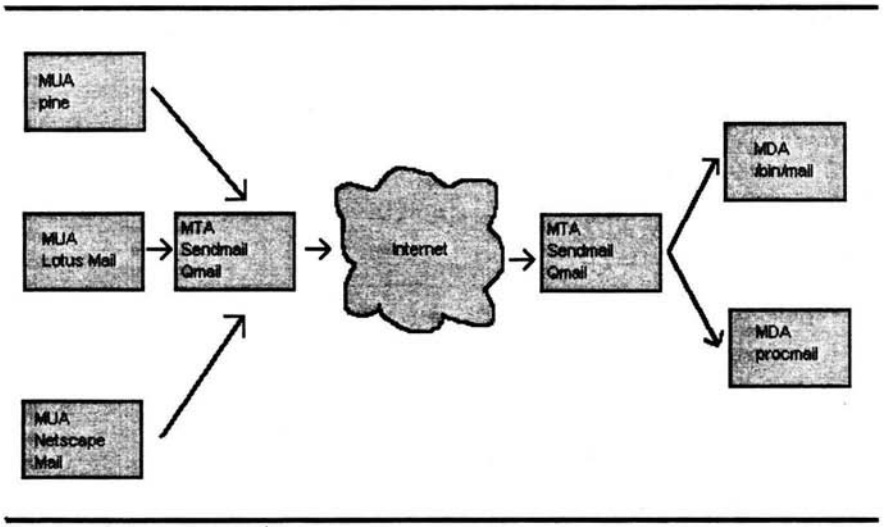
El MTA es el agente que se encarga de transportar el correo que ya tiene formato. El transporte de correo puede darse de forma local (en la misma máquina) o en forma remota (otra máquina o red). Como ejemplo se tiene a sendmail, como el más tradicional en internet, zmail y qmail.

El MDA se encarga de aceptar un mensaje que llegue desde MTA para depositarlo en un buzón de correo previamente establecido, la entrega del mensaje la realiza de acuerdo con las preferencias del usuario.

En una transferencia local de correo, donde tanto el remitente como el destinatario poseen cuentas en la misma máquina, el MTA es el responsable del transporte de correo desde sí mismo hasta el MDA local. Posiblemente en el proceso cambien los protocolos, direcciones y encaminamientos del correo. Así mismo el MTA actúa como pasarela, un mecanismo para obtener el mensaje desde una red a otra red que utilizan diferentes protocolos. En la inmensa mayoría de los casos, sólo habrá un único MTA por cada servidor de correo.

El tratamiento de las peticiones a un MTA recae sobre los agentes de entrega de correo (MDA). Procmail es un ejemplo de un MDA; un programa que nos permite procesar correos de forma sencilla, con el cual, se pueden separar los correos según determinados filtros, eliminar correos spam, reenviar a otras cuentas, activar antivirus de correos, responder automáticamente, ejecutar programas, y muchas funciones más.

La relación entre estos componentes de correo se muestran en la figura F1.1.1



Agentes de correo electrónico F1.1.1

Estableciendo una analogía con la figura anterior.

1. El emisor redacta un mensaje usando su MUA. Para él proceso es totalmente transparente una vez dada la orden de "Enviar".

2. El MUA del remitente pasa el mensaje y la dirección del destinatario al MTA de la misma máquina.

3. El MTA emisor identifica el dominio de la dirección destino y busca usando uno o varios DNS

4. Con esa información, el MTA emisor crea una conexión con el puerto 25 del servidor SMTP, una vez allí, usa los comandos del protocolo SMTP (HELLO, MAIL FROM:, RCPT TO: y DATA, entre otros) para dejar el mensaje. En caso de que no exista el destinatario, el MTA emisor emitirá un mensaje de respuesta del tipo *Returned mail: User unknown*, el cual será transmitido a su vez por el MTA emisor al remitente.

5. El trabajo para el MTA emisor ha terminado. Sin embargo, hay que situar el correo electrónico en el sitio adecuado para que el destinatario lo pueda consultar. El MDA pasará el mensaje al buzón correspondiente. Con eso finaliza el viaje del correo electrónico.

1.2 Buzón de correo electrónico

El buzón de correo electrónico consiste en un archivo que almacena los mensajes de cada usuario que tenga una cuenta de correo en el sistema. En la mayoría de los sistemas de correo se utilizan buzones con formato mbox, aunque existe también el formato maildir.

Sendmail utiliza el buzón de correo con formato mbox. En este formato, los mensajes se van colocando uno detrás de otro en el mismo archivo. El espacio reservado para estos archivos está condicionado por la capacidad de almacenamiento del equipo y por los criterios impuestos por el administrador. Cabe mencionar que por cada cuenta de correo en el sistema existe un archivo mbox.

El problema más común en los archivos con formato mbox es el bloqueo. Este se presenta cuando ocurre una caída en el sistema o en las comunicaciones al momento de accesar al archivo, ya sea por una consulta de correo o por alguna inserción de un nuevo mensaje. En caso de bloqueo, el cliente no podrá seguir utilizando su correo.

También se puede bloquear cuando algún usuario se excede del espacio asignado por el administrador de correo. En ambos casos el administrador requiere desbloquear este archivo para que el usuario pueda enviar y recibir mensajes en su correo.

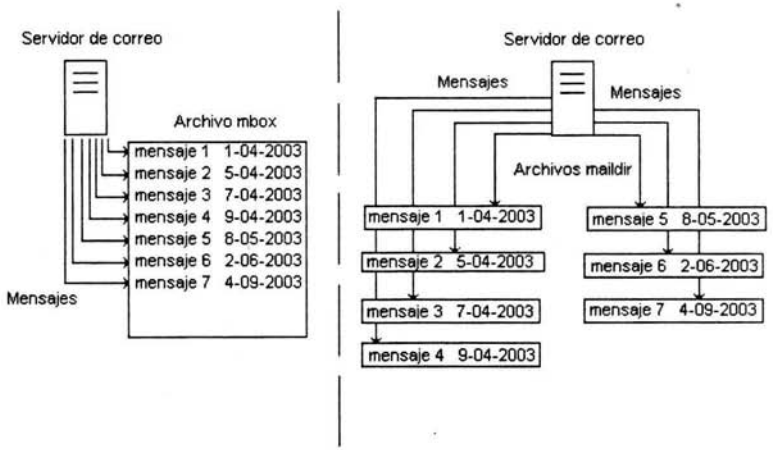
Para evitar la problemática anterior existe el buzón de correo con formato maildir. Este formato por sus características no permite bloqueo alguno.
Maildir consta de tres carpetas donde los correos se almacenarán. Dependiendo del estatus del correo es la carpeta donde este se encuentra. A diferencia de mbox cada nuevo mensaje es un nuevo archivo para maildir, por lo que cualquier daño que sufra algún archivo solo será para ese archivo, lo que hace que se mantenga activo el envío y la recepción de mensajes.

Otro problema que presenta los buzones de correo con formato mbox está en los respaldos. Con mbox los respaldos se realizan completos, es decir, se respalda todo el archivo mbox de cada usuario, en la mayoría de los casos los respaldos se condicionan con la fecha, es difícil saber si el archivo mbox fue sólo consultado o modificado, esto debido a que cualquier operación sobre él modifica la fecha. Como en mbox cada mensaje se agrega uno de tras de otro en el mismo archivo, es muy probable que los respaldos contengan información respaldada con anterioridad.

El tiempo que se lleva en este tipo de respaldos es proporcional al tamaño de los archivos mbox de cada usuario, esto nos indica una problemática muy clara en las horas máquina que se invierte en los respaldos y el gasto creciente que se tendrá que realizar en cintas magnéticas para almacenar la información.

Debido a que en maildir cada correo es un archivo, es posible respaldar sólo archivos que no han sido respaldados con anterioridad, es decisión del administrador del correo si el respaldo lo realiza sobre la tres carpetas que componen a maildir o nada mas alguna de ellas. De lo que puede estar seguro es que el respaldo sólo será con información que contenga la fecha que se requiera, disminuyendo notablemente los tiempos en que se lleva a cabo los respaldos y el número de cintas donde se almacena la información.

A continuación se muestra un ejemplo de cómo almacenan los mensajes los buzones de correo con formato mbox y maildir respectivamente.



Almacenamiento de mensajes en archivos con formato mbox y maildir. F1.2.1

1.3 Cuentas de correo electrónico

Sendmail como tal no maneja cuentas de correo sino que utiliza los archivos /etc/passwd y /etc/shadow de UNIX, como se sabe, ambos archivos tienen el control de altas y bajas de usuarios. UNIX tiene una limitante en el alta de usuarios, cuestión que aplica de igual manera en sendmail.

En UNIX, el constante crecimiento de cuentas de correo puede propiciar problemas en el funcionamiento del sistema, llegando en ocasiones a provocar caídas notables en el rendimiento de los equipos.

Qmail utiliza LDAP (Protocolo de Acceso Ligero a Directorios) para la administración de usuarios. LDAP es una base de datos en donde se pueden realizar las altas, bajas y modificaciones de cada usuario sin utilizar archivos del sistema operativo.

LDAP se encarga de almacenar las características y contraseñas de los usuarios, donde un administrador (diferente a root), podrá realizar todas las operaciones normales que se realizan en una base de datos común.

LDAP cuenta con la capacidad de replicar la información entre varios servidores para garantizar la estabilidad y seguridad en el servicio, además de que permite el uso de clusters para distribuir cargas, lo que evita posibles caídas por fallas en algún servidor de correo.

1.4 Seguridad

Una razón por la que se considera a sendmail extremadamente potente y al mismo tiempo peligroso es el acceso pleno a los archivos de configuración. Sendmail canaliza los mensajes de correo a través de sus archivos de configuración, además de que realiza todas las funciones de encaminamiento incluyendo análisis, reenvío, entrega, respuesta y solicitud de mensajes.

Por lo anterior nadie escribe un archivo sendmail.cf comenzando desde una página en blanco en un editor de texto. Si se está creando un nuevo servidor de correo central para una red, examinando los recursos disponibles, es casi seguro que se puede encontrar un archivo sendmail.cf que sólo requiera de algunas modificaciones para lograr el servidor de correo deseado.

En la mayoría de los casos, las modificaciones que se realicen en los archivos de configuración de sendmail dejan agujeros disponibles para cualquier ataque, siendo estos muy peligrosos debido al control que se puede tener sobre el equipo, ya que todo lo que realiza sendmail lo hace a través del usuario root (usuario que cuenta con todos los privilegios del sistema operativo).

Qmail evita este tipo de ataques ya que la minoría de procesos son ejecutados con root. Además de que cuenta con una gran capacidad para seguir las pistas de los mensajes hasta que llegan a su destino, lo que previene bucles infinitos de reenvío y precisión en la detección de problemas.

Con la incorporación de base de datos LDAP a QMAIL el control de acceso a usuarios está sólo del lado de la base de datos, lo que elimina el uso de archivos de sistema operativo como lo hace sendmail con /etc/passwd y /etc/shadow.

1.5 Otras diferencias entre Sendmail y Qmail.

Entre las principales diferencias de qmail con sendmail encontramos las siguientes:

- Excelente rendimiento gracias a una gestión muy rápida en la cola de espera.
- Configuración muy sencilla, a través de un conjunto de archivos de control y de variables de entorno. Cumple con los RFC-822 y RFC-1123. Gestor SMTP *qmail-smtpd* "8-bit clean" (maneja mensajes con caracteres de 8 bits), gestor *qmail-qmtpd* que implementa el futuro estándar QMTP.
- Desaparición del archivo */etc/aliases*, sustituido por la súper base alias */var/qmail/users/assign*.
- Desaparición de archivos *.forward* en beneficio de los archivos de *.qmail*, que son a la vez más potentes y más sencillos.
- Configuración y gestión del correo muy simplificada, gracias a los archivos *.qmail* interpretados por el MDA *qmail-local*.
- Cualquier usuario puede crear y manejar sus propias listas de distribución,

- Nuevo formato de buzón llamado maildir, más fiable que el formato *mbox* y particularmente útil en caso de utilización de NFS. Se suministran las utilidades *maildir2mbox* y *maildirmake*

CAPÍTULO 2

LDAP (Protocolo Ligero de Acceso a Directorios)

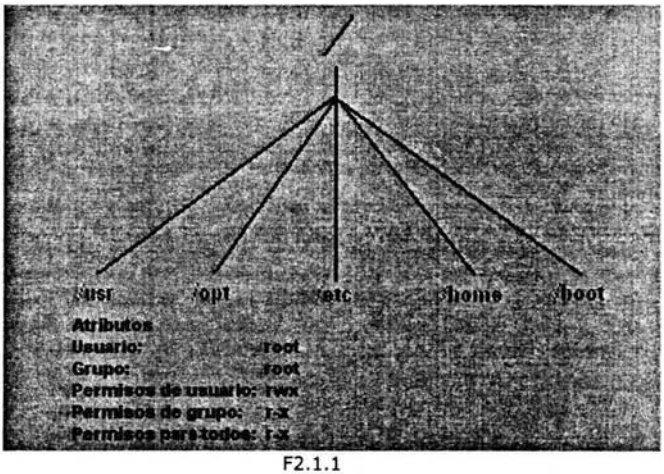
En el presente capítulo se explicará el concepto de LDAP, así como la seguridad, instalación y configuración de archivos Openldap para el diseño de servidores de directorio a través de schemas.
Se especifica el manejo que se lleva a cabo para efectuar operaciones de altas, bajas, modificaciones y consultas sobre las cuentas de los usuarios de correo.
Se expone la habilidad que tiene LDAP para la replicación de información en otros equipos, así como el uso de clusters.

2.1 ¿Qué es LDAP?

LDAP Surge en 1993 en la Universidad de Michigan, sus siglas significan (Lightweight Directory Access Protocol).
Es un modelo que se basa en un servicio de directorio cliente servidor que nos permite establecer las estructuras de nuestros directorios por medio de reglas ya definidas por la RFC1777, las cuales especifican una serie de operaciones que se llevan a cabo para realizar consultas, obtención y actualización de datos a través de la red utilizando TCP/IP como protocolo de comunicación.

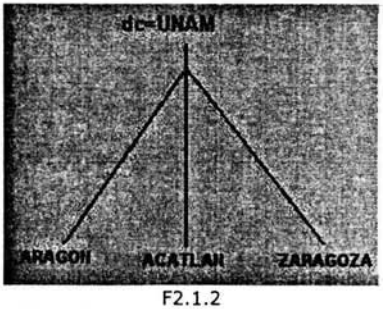
LDAP inicialmente se utilizó como un Front-End ó interfaz final para X.500, pero puede utilizarse con servidores de directorio únicos y con otros tipos de servidores de directorio, muy similares a los directorios del sistema de archivos de UNIX.

La figura F2.1.1 nos muestra una estructura de archivos en UNIX.



F2.1.1
Estructura de archivos en Unix

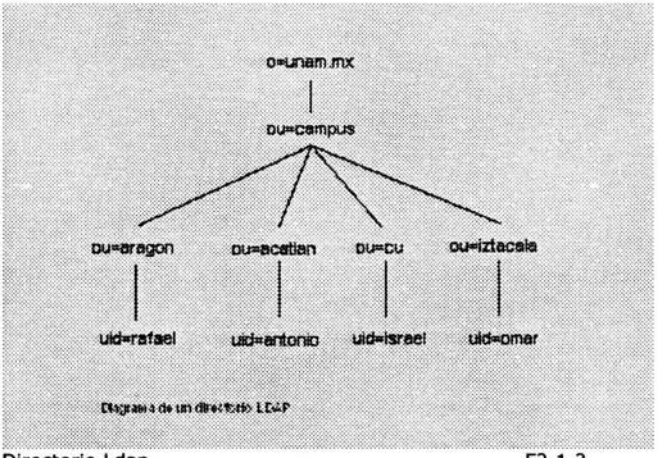
Se puede mostrar la similitud de la estructura de archivos de Unix de la figura anterior con la figura F1.1.2, que representa un LDAP de una organización.



F2.1.2
Estructura de archivos en LDAP

Un servidor de directorio es como una base de datos, está optimizado para que se realicen lecturas, navegaciones y grandes búsquedas. La información se encuentra organizada de modo jerárquico (estructura típica de un árbol) lo que incorpora sistemas de filtrado muy completos que ayudan a realizar búsquedas determinadas en distintas zonas del árbol teniendo una rápida capacidad de respuesta.

En cada servicio de directorio hay un punto inicial desde el cual comenzar nuestro árbol, a este punto en sistemas UNIX normalmente se le llama root, para LDAP es el nodo principal.
Debajo de este nodo principal se encontrarán subdirectorios y directorios, los cuales, tendrán a su vez otros subdirectorios formando una especie de árbol. A continuación se muestra un diagrama de un directorio LDAP.



Directorio Ldap F2.1.3

Normalmente en un servidor de directorio la información se lee mucho más de lo que se escribe, como consecuencia, los directorios no implementan complicados esquemas para transacciones o esquemas de reducción como los que contiene una base de datos común, sin embargo contiene diferentes métodos que nos permiten almacenar en el directorio varios tipos de datos y establecer requisitos para hacer referencia a la información, ya sea para consultas o actualizaciones, así como la forma de proteger al directorio de accesos no autorizados.

Con LDAP los servicios de directorios tienen la capacidad de replicar la información, es decir, generar réplicas de las bases de datos en diferentes computadoras, así al modificar información de alguna base, se actualizarán todas las demás, garantizando la seguridad, disponibilidad y fiabilidad de la información.

Cuando se replica la información de un directorio se puede aceptar inconsistencias temporales entre la información que hay en las réplicas, siempre y cuando exista una sincronización de la información entre las computadoras.

Como ya se mencionó, un servicio de directorio LDAP se basa en un modelo cliente-servidor, donde uno o más servidores LDAP contienen los datos que conforman el árbol de directorio. El cliente se conecta y le solicita un servicio a LDAP, el servidor LDAP le contesta con la respuesta correspondiente o bien con una indicación de donde puede hallar más información. No importa con que servidor LDAP se conecte el cliente ya que siempre observará la misma vista del

directorio; la operación que realice el cliente hace referencia a la misma entrada como la que haría en cualquier otro servidor.

Existen numerosas implementaciones de LDAP. A continuación se nombran algunas de ellas:

1. UMICH Ldap
   UMICH Ldap es una de las primeras implementaciones libres de LDAP, se desarrollo en la Universidad de Michigan convirtiéndose en el precursor de lo que sería el directorio libre por excelencia OpenLDAP. UMICH implementó la versión 2 de LDAP obteniendo un gran logro en el desarrollo de servicios de directorio.

2. iPlanet (Netscape) Directory Server
   El directorio de iPlanet, antiguamente el servidor de directorio de Netscape, es otro de los famosos servidores LDAP. Es un directorio muy robusto, que implementa las dos especificaciones de LDAP (versión 2 y versión 3) con el gran problema de que es un software propietario.

3. OpenLDAP
   Éste es la alternativa hoy en día a los directorios propietarios. Es un directorio LDAP escrito conforme a los estándares que soporta LDAP versión 2 y gran parte de la versión 3.

De las tres implementaciones utilizaremos OpenLDAP ya que su código es libre además de que contiene todos los estándares de construcción para crear un servidor de directorio.

2.2 OpenLdap y su Instalación

OpenLDAP incluye dos demonios para su configuración:

- Slapd
  Es un demonio que sirve para iniciar el servidor de directorio OpenLDAP, tiene la función de recibir, gestionar y servir las peticiones que se realicen a OpenLDAP.
  Slapd es el núcleo y donde se implementan las funciones básicas de LDAP.

- Slurpd
  Es un demonio que sirve para replicar un directorio LDAP. Por el momento OpenLDAP sólo permite la replicación de un árbol entero sin replicar ninguna de sus partes.

Es necesario para llevar a cabo la instalación del directorio iniciar una sesión como súper usuario ó usuario root para tener todos los privilegios de instalación en el sistema.

Se descarga la versión openldap-stable-20021810.tar que se encuentra disponible en http://www.openldap.org. Una vez que se obtiene el archivo tar (archivo comprimido), éste debe descomprimirse en un directorio temporal (/tmp por lo general) para poder iniciar la instalación.

#tar -zxvf /tmp/openldap-stable-20021810.tar.gz

Dentro del directorio /tmp se crea el subdirectorio openldap-2.0.27. A continuación se ejecutarán los siguientes comandos:

#cd *tmp/openldap-stable-20021810.tar.gz*
#./configure

Si se desea se puede configurar el openldap con tcpwrappers ejecutando la siguiente instrucción:

#./configure --enable-wrappers

Este comando configura los archivos de instalación de acuerdo a su sistema. Con la opción --*enable-wrappers* se activa ésta herramienta para OpenLdap.

Se ejecuta el siguiente comando para generar las dependencias:

#make depend

A continuación se compila el servidor mediante la orden:

#make

Se instalan los ejecutables y las páginas de manual mediante el siguiente comando:

#make install

El comando anterior instala OpenLDAP bajo el directorio /usr/local/etc/openldap (en caso de que no se hayan cambiado los parámetros al tiempo de compilar OpenLDAP.

2.3 Configuración de OpenLdap

Para configurar el servidor se tienen que editar los archivos slapd.conf y ldap.conf. Ambos archivos se almacenan en la ruta */usr/local/etc/openldap* .

2.3.1 Archivo Slapd.conf

Slapd.conf es el archivo de configuración principal de OpenLDAP, en él se configuran todos los parámetros necesarios para el servidor LDAP como se muestra en la figura F1.3.1. Para llevar a cabo la configuración es necesario modificar algunos parámetros, cabe mencionar que estos archivos de configuración no permiten los espacios en blanco al inicio de cada línea.

Parámetros básicos del archivo slapd.conf I F2.3.1

- **Include**

Este parámetro nos indica la ruta de otros archivos de configuración utilizados por el servidor LDAP, los cuales sirven para soportar tipos de atributos adicionales y otras clases de objetos que nos permitirán almacenar características de los usuarios, así como de asignar un perfil más personalizado de cada uno de los usuarios.

- **Referral**

Indica un Servidor LDAP alterno en dado caso de no poderse efectuar la búsqueda en el servidor LDAP actual (Desactivado con comentario #).

- **Access to**

Parámetro utilizado para administrar los accesos de los usuarios al servidor LDAP, (Desactivado con comentario #).

- **Pidfile**

Contiene el número de proceso asignado por el sistema operativo al iniciar el servicio de LDAP.

- **Argsfile**

Contiene parámetros que se utilizarán al iniciar el servidor LDAP.

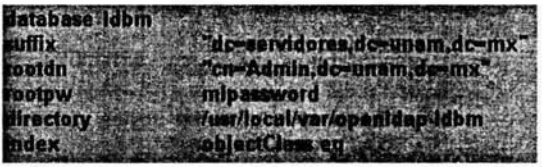
- **Loglevel**

Indica el nivel de registros ("log") producidos por el servidor LDAP.

Posibles valores:

Level	Description	Level	Description
- 1			Desactiva todos los debugs
1			Imprime todas las llamadas a funciones
4			Imprime más datos de depuración
16			Impresión de paquetes enviados y recibidos
64			Archivo de configuración de procesos
256			Estados del log Conexiones/operaciones/resultados
1024			Imprime todas las comunicaciones entre el Cliente y el servidor

Dentro del archivo slapd.conf encontramos parámetros que sirven para configurar la base de datos LDAP, como se presenta en la figura F2.2.2.



F2.3.2
Parámetros básicos del archivo slapd.conf II

- **database**

Indica el tipo de base de datos que se utilizará, generalmente ldbm (otras posibles alternativas: shell,passwd), cada declaración de *database* se considera una base de datos por separado.

- **suffix**

Este parámetro indica el nodo principal o dn (Distinguished Name) de la base de datos, es decir, el nodo sobre el cual se deriva toda la información.

- **rootdn**

Establece el nodo (usuario) que tiene privilegios globales para modificar la base de datos LDAP. Es necesario mencionar que los privilegios se tienen a partir del nodo donde se declare.

- **rootpw**

Indica la contraseña para el usuario rootdn.

- **directory**

Define el directorio donde se encuentra la base de datos LDAP, este directorio debe existir antes de iniciar el servidor LDAP.

- **index**

Define la manera de indexar a los atributos de los objetos.

2.3.2 Archivo Ldap.conf

El archivo de configuración ldap.conf es utilizado para montar un sistema que corre cuando el cliente de LDAP solicita un servicio. Este archivo lee las utilidades de LDAP como ldapsearch, ldapdelete, ldapadd, ldapmodify, etc.

La configuración de este archivo se puede especificar utilizando las variables de ambiente LDAPCONF y LDAPPRC.

La variable LDAPCONF debe contener el path del archivo de configuración, ésta puede ser una ruta absoluta o relativa. La variable LDAPRC debe contener el basename del archivo de trabajo.

En la figura F2.3.3 se muestran algunas opciones de configuración del archivo ldap.conf.

Configuración del archivo ldap.conf F2.3.3

- **BASE (base)**

Se utiliza para especificar el *dn* o nodo principal en las operaciones que realice Ldap.

- **BINDDN (dn).**

Se debe especificar el *dn* en formato ldap. Esta opción es para un solo usuario.

- **HOST (nombre [:puerto]...)**

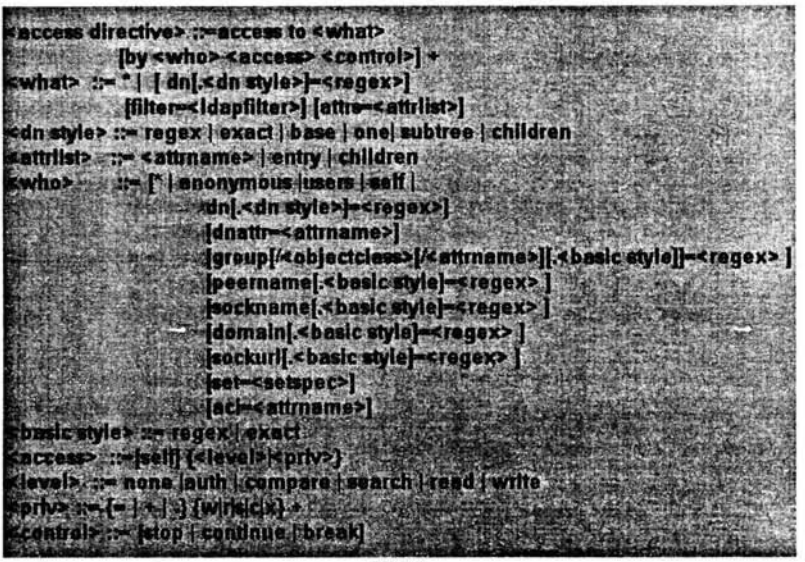
Se especifica el nombre del servidor ldap.

- **PORT (puerto)**

Especifica el puerto que utilizará ldap para sus conexiones.

2.4 Control de Acceso (Access to)

Todas las entradas y atributos que se ingresan en LDAP son controlados por el archivo de configuración slapd.conf. La estructura de las líneas de acceso es la siguiente:



F2.4.1
Directivas para el acceso a la información contenida en la base de datos.

Se entiende por atributos, a las características que tienen los objetos, estos pueden ser o no obligatorios, por ejemplo un objeto usuario necesita de un password para la seguridad de su información, para un objeto persona es necesario el atributo nombre para identificar a la persona, más no así, si tiene mail o no. Es así como a los usuarios se les permite el acceso a los diferentes atributos.

Para crear accesos a los datos se tiene varias directivas. La directiva <what> es la parte donde se seleccionan las entradas, puede ser un *dn*, un atributo ó un filtro. En la directiva <who> se especifica las entidades a las que se tiene acceso, y en la directiva <access> es donde se otorgan los accesos.
Las directivas <who> <access> <control> permiten múltiples y diferentes accesos a los atributos que se encuentran en el mismo grupo de entrada del *dn*.

Como se muestra en un segmento de la figura anterior, la forma más general de las líneas que permiten el acceso a la base de datos es la siguiente:

```
access to <what>
        by <who> <access> <control>
```

2.4.1 El control de Access to

En la directiva <what> es la parte en donde se especifican los accesos, determinando las entradas y atributos, las entradas se pueden seleccionar por una expresión regular que puede corresponder a la misma que contiene el *distinguished name*, por ejemplo:

```
access to dn="dc=unam,dc=mx"
```

Nota: el *dn* especificado debe estar normalizado por la RFC2253, que restringe las formas del dn. En particular dentro del dn o expresión regular no se deben de dejar espacios y se deben utilizar comas para separar los componentes. Como ejemplo de un *dn* no normalizado se tiene:

"cn=daniel, dc=unam,dc= com" (espacio entre daniel, dc=unam)

Los atributos se seleccionan de una lista, donde una coma separa los nombres de cada atributo.

```
attr = <lista de atributos>
```

En el siguiente ejemplo se describe el uso del atributo userPassword.

```
access to attr=userPassword
        by self write
        by * none
```

El acceso al atributo userPassword sólo será de escritura para cada dueño, pero ningún usuario podrá consultar este atributo.

También se pueden generar filtros que corresponden con algunos atributos de entrada, ejemplo:

Filter = <ldap filter>, donde el *ldap filter* es una cadena que representa una búsqueda filtrada

Los filtros se utilizan para realizar búsquedas por un criterio. La sintaxis de un filtro en LDAP es una expresión lógica donde el operador lógico (and / or / not / item) aparece antes de los argumentos de búsqueda.
Un dato que se puede utilizar para realizar una búsqueda es el *sn* ó surename, este atributo, como muchos otros, pueden servir de filtros para la búsqueda de la información. Los atributos pueden combinarse a través de operadores lógicos. El siguiente ejemplo maneja un filtro con dos atributos combinados por un operador lógico.

```
filter= "(& (sn=Fredy) (mail=*))"
```

El filtro traerá como resultado todos los objetos que en su atributo *sn* tengan el valor de Fredy con sus respectivos mails

Últimamente se ha escogido un selector especial que se utiliza para obtener cualquier entrada, este selector se representa por el símbolo * (asterisco) y se usa cuando no hay una selección específica de campos en la base de datos.

Como por ejemplo "dn=.*"

2.4.2 Permisos para los Accesos

La directiva <who> identifica la entidad o entidades a las que se les ha otorgado acceso, el acceso se otorga a "entidades" no a "entradas". A continuación se muestra la tabla 2.4.1 que resume las especificaciones de las entidades.

Entidades y especificaciones para los accesos en LDAP

Especificación	**Entidad**
*	Todos, incluyendo anónimos y autentificación de usuarios
Anonymous	Usuarios anónimos sin autentificación
Users	Usuario autentificado
Self	Usuario asociado con la tarjeta de entrada
Dn=<regex>	Usuario que corresponde con la expresión regular

Tabla 2.4.1

El *dn* especificado de la expresión regular debe de corresponder con el *dn* normalizado de la entidad actual. Se entiende como normalizado la eliminación de espacios y el uso de comas "," como separador de componentes.

Existen otros factores de control, un <what> puede estar restringido por una expresión regular que corresponde con el nombre de dominio del cliente.

```
access to dn="dc=unam,dc=mx"
        by * anonymous auth
```

El ejemplo anterior nos dice que todos los usuarios anónimos tienen que autentificarse para tener acceso al nodo unam.mx y a los nodos que se encuentran debajo de el.

La especificación dnattr se utiliza para dar acceso a todos aquellos que están listados en un atributo de entrada del *dn.*

```
dnattr=<dn_value nombre del atributo>
```

2.4.3 Niveles de Acceso

Los niveles de acceso se presentan en la tabla 2.4.2:

Niveles de Acceso en LDAP

Niveles	**Privilegios**	**Descripción**
None		Sin acceso
Auth	=x	Obligatorio la autentificación
compare	=cx	Se necesita una comparación
search	=scx	Aplicar búsquedas filtradas
read	=rscx	Lectura del resultado de una búsqueda
write	=wrscx	Se necesita para modificar o renombrar

Tabla 2.4.2

Cada nivel inferior implica todos los niveles superiores, esto se puede utilizar cuando se necesita dar a un acceso de entrada con varios niveles como por ejemplo lectura, búsqueda y comparación.

2.4.4 Evaluación del control de Acceso

Cuando se evalúa alguna solicitud de acceso a la base de datos, slapd compara la entrada y/o atributo dado en la directiva <what> del archivo de configuración. Por cada entrada *access to* se comunica con la base de datos, la cual toma esta entrada seguido por la ejecución de las siguientes directivas. Con esta prioridad las directivas son ejecutadas en el orden en que éstas aparecen en el archivo de configuración.

Slapd utiliza para el acceso a la base de datos la primera directiva de acceso, enseguida compara la entidad requerida en la directiva <who>, finalmente compara el acceso otorgado en la directiva <access>. El orden de evaluación de las directivas de acceso dependen de cómo se coloquen dentro del archivo de configuración. Si una directiva de acceso es más específica, como en el caso de atributos, éstas deben aparecer primero en el archivo de configuración. Similarmente si una directiva <who> es más especifica que otra, esta debe aparecer primero.

2.4.5 Ejemplos del Control de Acceso

En esta sección se muestran algunos ejemplos del control de acceso.
Primero se mostrarán ejemplos simples de esto.

```
access to * by * read
```

En el ejemplo anterior la directiva otorga accesos de lectura a todos.

```
access to *
        by self write
        by anonymous auth
        by * read
```

La directiva anterior permite a todos los usuarios la modificación de sus propias entradas, con autentificación de usuarios anónimos y permisos de lectura.

Note que sólo el primer *by* (cláusula <who>) es la que se está aplicando. Los usuarios anónimos tienen garantizado la autentificación no la lectura. La última directiva podría ser como ya se ha visto *by users read*.

El siguiente ejemplo muestra el uso de una expresión regular para seleccionar las entradas por *dn* en dos directivas de acceso, donde el orden es significativo.

```
access to  dn=".*,dc=ejemplo,dc=com"
        by * search
 access to  dn=".*,dc=com"
        by  * read
```

El acceso de lectura se otorga a entradas bajo el nodo dc=com, excepto para las entradas debajo de dc=ejemplo,dc=com, a la cual se le otorga el acceso de búsqueda. Si el orden de las anteriores directivas se da en sentido contrario nunca serán localizadas por lo que nunca se ejecutarán en ese sentido.

Nota. Si las directivas de acceso no encuentran la directiva <who> el acceso será denegado, sólo se garantiza el acceso a todas aquellas que terminen con la directiva implícita access to*, ejemplo:

```
        by  * none              #se niega el acceso
access to *  by  * none         #se permite el acceso
```

El siguiente ejemplo nos muestra nuevamente la importancia del orden de las directivas <who>, también se muestra el uso de atributos.

```
access to dn="(.*,)?dc=ejemplo,dc=com" attr=homePhone
        by self write
        by dn="(.*,)?dc=ejemplo,dc=com" search
        by * none
        by domain=.*\.example\.com read
access to dn="(.*,)?dc=ejemplo,dc=com"
        by self write
        by dn=".*,dc=ejemplo,dc=com" search
        by anonymous auth
```

El ejemplo anterior aplica sobre las entradas que tenga el nodo "dc=ejemplo,dc=com". De estas entradas cada usuario podrá modificar el atributo homePhone, realizar búsquedas debajo del nodo ejemplo.com, otros clientes que se conectan al dominio tendrán permisos de lectura. Todos los usuarios anónimos tendrán que autentificarse para tener acceso (by * none).

En algunas ocasiones se necesita permitir la adición ó eliminación de un *dn* desde un atributo. Esto es, si es necesario crear un grupo y permitir a este grupo que agregue o elimine sus propios *dn* desde un atributo, se añadirá el siguiente código en el archivo de configuración:

```
access to attr=member, entry
        by dnattr=member selfwrite
```

El dnattr nos dice que los accesos se aplican a las entradas listadas en los miembros del atributo, mientras que selfwrite, como miembro, puede añadir o eliminar sus propio *dn* desde el atributo y no otro valor.

El siguiente ejemplo contiene el texto del archivo de configuración slapd.conf, en él, se definen dos bases de datos. Las primeras cuatro líneas se describen a continuación:

1. #ejemplo de configuración del archivo
2. include /usr/local/etc/schema/core.schema
3. referral ldap://root.openldap.org
4. access to * by * read

Línea 1. Comentario.
Línea 2. Incluye dentro del archivo de configuración, el archivo core.schema
Línea 3. Contiene la referencia donde se encuentran las bases de datos definidas además de referirse el servidor LDAP que corre a través del puerto 389 en el host root.openldap.org

Línea 4. Define un control de acceso global. Esto aplica para todas las entradas.

La siguiente sección del archivo de configuración define una BDB(base de datos) que maneja queries a través del nodo de entrada "dc=ejemplo,dc=com". La base se replicara en dos slapd esclavos, uno será una copia fiel de la primera replica, mientras la segunda replica se utilizara para pruebas.
Los índices son atributos, donde el atributo como userPassword deberá estar protegido como un acceso no autorizado.

```
5.  # BDB definición para ejemplo.com
6.  database bdb
7.  suffix "dc=ejemplo,dc=com"
8.  directory  /usr/local/var/openldap-data
9.  rootdn "cn=Administrador,dc=ejemplo,dc=com"
10. rootpw secret
11. # replication directives
12. replogfile /usr/local/var/openldap/slapd.replog
13. replica host=esclavo1.ejemplo.com:389
14.         binddn="cn=replicador,dc=ejemplo,dc=com"
15.         bindmethod=simple credentials=secret
16. replica host=esclavo2.ejemplo.com:389
17.         binddn="cn=replicador,dc=ejemplo,dc=com"
18.         bindmethod=simple credentials=secret
19. #  atributos definidos como índices
20. index  uid pres,eq
21. index  cn,sn,uid pres,eq,approx,sub
22. index objectClass eq
23. #  Acceso de control a la base de datos
24. access to attr=userPassword
25.             by self write
26.              by anonymous auth
27.              by dn="cn=Admin,dc=ejemplo,dc=com" write
28.              by * none
29. access to *
30.              by self write
31.              by dn="cn=Admin,dc=ejemplo,dc=com" write
32.              by * read
```

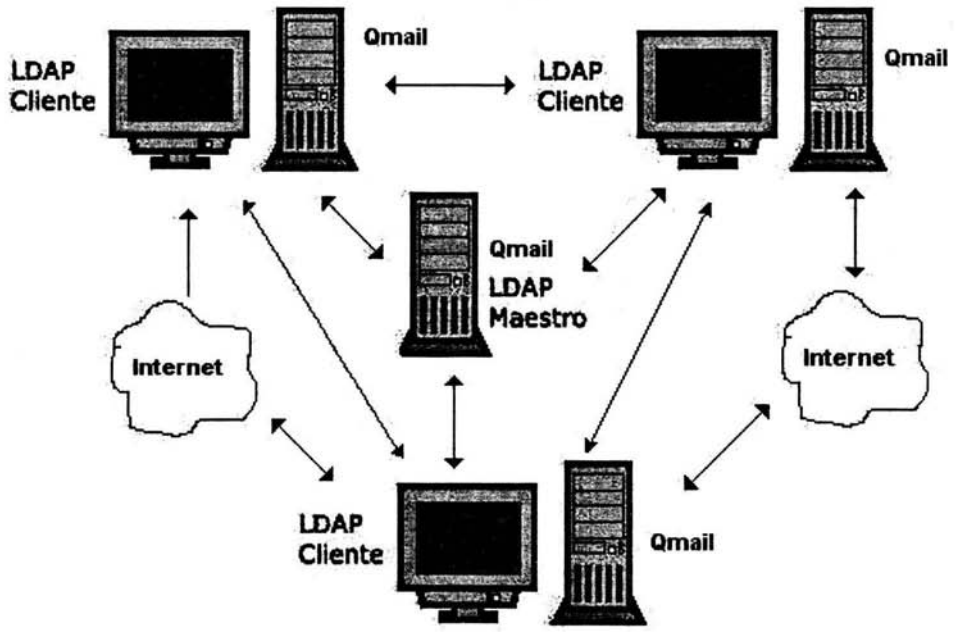
Línea 5. Comentario.
Línea 6.La definición de la base de datos comienza con la directiva *database bdb.*
Línea 7. Se define el nodo principal
Línea 8. Se define el directorio donde se almacenarán los archivos de la base
Línea 9. Se define el Administrador para la base de datos
Línea 10. Contraseña para el administrador
Línea 12-18 Replicación de bases de datos
Línea 19-22 Índices para el acceso a la base de datos
Línea 24-32 Se especifica el control de acceso. El atributo userPassword puede escribirse por las entradas mismas o por la entrada que se definió en el administrador, nadie tiene permisos de lectura en este atributo. Los usuarios anónimos tienen que autentificarse, todos los demás atributos pueden escribirse a través de la entrada de cada usuario o del administrador y leerse por cualquier otro usuario.

La siguiente sección define otra base de datos (BDB). Esta definición toma al nodo principal dc=ejemplo,dc=net pero sigue siendo administrado por la misma entidad de la primera base de datos. Note que sin la línea 39, el acceso de lectura sería permitido debido a la regla del acceso global que está en la línea 4.

```
33.  # BDB base de datos ejemplo.net
34.  database bdb
35.  suffix "dc=ejemplo,dc=net"
36.  directory /usr/local/var/openldap-data-net
37.  rootdn "cn=Administrador,dc=ejemplo,dc=com"
38.  index objectClass eq
39.  access to * by users read
```

2.5 Cluster

Con LDAP se pueden utilizar clusters dentro del sistema de correo. El cluster es un arreglo de computadoras que comparten cuentas de usuarios a través de LDAP. Esto permite que entre varias computadoras se almacenen los correos de los usuarios distribuyendo las cargas. La figura F2.5.1 muestra el funcionamiento del cluster en LDAP.



Funcionamiento del cluster en LDAP F2.5.1

El MTA se encarga de distribuir los correos según lo que esté definido en la base de datos LDAP, debido a que LDAP contiene un campo donde se almacena el *mailhost* ó el nombre de la computadora que contiene los correos del usuario, busca la computadora donde el usuario tiene

almacenado sus correos y ahí es a donde deposita los nuevos correos. Es así, como la capacidad de usuarios se puede incrementar sin saturar un único servidor con sobrecarga de usuarios.

Nota: Para configurar el cluster en Qmail sólo debe de modificarse dos archivos de control de qmail, el archivo me, donde se cambia el hostname por el servidor que contiene la base de datos LDAP y el archivo ldapcluster que debe de contener un 1 como se muestra a continuación:

Archivo me
Servidor.que.contiene.ldap ó dirección IP

Archivo ldapcluster
1

2.6 Ejecución del servidor LDAP

Para llevar a cabo la ejecución del servidor OpenLDAP se ejecuta el comando *slapd,* que se encuentra ubicado en /usr/local/libexec/. Este comando inicia el demonio LDAP bajo el puerto 389 de TCP por default. Para ejecutar el demonio slapd se necesita especificar la ruta donde se encuentra el archivo de configuración más un parámetro como se muestra a continuación:

#./slapd -f /usr/local/etc/openldap/slapd.conf

El parámetro *-f* indica el archivo de configuración que se utiliza para iniciar el servidor LDAP. También es posible indicar ciertos parámetros de arranque como el (los) puerto(s) TCP:

#./slapd -h "ldaps://ldap://127.0.0.1:978",

Lo anterior inicia el servidor LDAP con SSL (Secure Socket Layer) bajo el puerto 978 de TCP.

El indicar estos parámetros en la línea de comandos para cada conexión puede ser tedioso, por lo que se recomienda agregar estos parámetros al archivo *slapd.args* ubicado generalmente en /usr/local/var/ (archivos modificables de *slapd.conf*)

Para cerciorarse que LDAP está en funcionamiento ejecute el siguiente comando.

#ps -fea | grep ldap

En caso de que el servidor LDAP esté funcionando, después de ejecutar el comando anterior en la línea de comandos, nos muestra en pantalla algo similar al siguiente código:

#ps -fea | grep ldap
root 1084 1 0 Nov 1l ? 0:00 /usr/local/libexec/ slapd -f /usr/local/etc/openldap/slapd.conf

Otra forma sería realizando un telnet al puerto TCP en cuestión, si la conexión no es aceptada verifique los registros ("logs") de OpenLDAP.

2.7 Terminación del servidor LDAP

Para terminar el servicio de LDAP se debe ejecutar el siguiente comando:

#kill -INT `cat /usr/local/var/slapd.pid`

Lo anterior asume que el parámetro *pidfile* que se encuentra en el archivo *slapd.conf* está definido como:

pidfile /usr/local/var/slapd.pid

Esto es lo mismo que hacer:

```
#ps –fea ldap
ldap   32170   1  0 12:20 ?      00:00:00 /usr/sbin/slapd -u ldap

#kill –9 32170
```

Donde 32170 es el proceso que corresponde con el servicio de ldap.

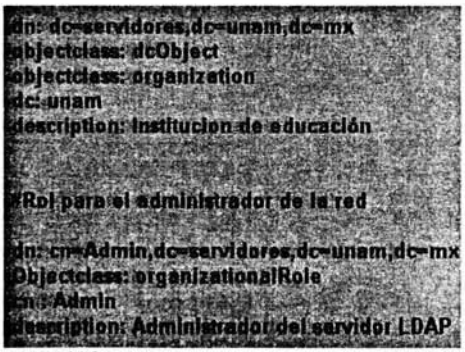
Este identificador de proceso se encuentra en el archivo slapd.pid, y es el que finalmente termina por eliminarse.

2.8 Inserción de datos en LDAP

La manera estándar de insertar datos en LDAP es a través de archivos LDIF (Formato para intercambio de datos en LDAP); estos archivos tienen un formato de intercambio de datos, que se utilizan para representar entradas en LDAP en formato de texto sencillo.

Con los archivos ldif es posible intercambiar información entre dos servidores LDAP, así como motores de bases de datos ejecutándose en diferentes sistemas operativos.

En la figura F2.8.1 se muestra un ejemplo de un archivo de entrada LDIF, en donde cada registro de entrada se separa con una línea en blanco.



UNAM.ldif F2.8.1

El primer elemento de cada estructura es el *dn* (*distinguished name)*.Esta línea define el *dn* para la entrada de nivel superior. Este *dn* será la raíz del árbol de directorios.

El objectclass está directamente relacionado con los schemas que son los que definen el tipo de objeto para el *dn.*

Los schemas no son más que archivos que definen a los objetos y sus atributos, dependiendo del objeto que se necesite será el schema que se utilice. Los objetos a su vez tienen atributos, a los cuales se les puede asignar diferentes valores, siendo algunos obligatorios. Un objeto de clase consta de tres partes (lo que debe tener, lo que requiere y lo que puede tener), además de los atributos que se requieran utilizar.

Existe una definición estándar de schemas (objetos de clases, atributo y sintaxis) contenidas dentro del archivo slapd.conf en LDAP, aunque se pueden crear otras para necesiades específicas.

El atributo *dn* o *Distinguished Name* de la figura F1.8.1 es:

dn: dc=servidores,dc=unam, dc=mx

La composición de cada *distinguished name* puede variar, en este caso se utilizo el vocablo *dc* de "Domain Component", sin embargo, también hubiera sido posible utilizar *c* de "Country", *o* de "Object", *p* de "País" , *cd* de "Componente Dominio" etc....

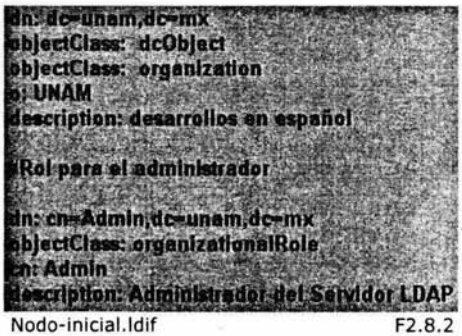
Los vocablos son sólo descriptivos y su única restricción depende de la definición que se haya hecho en el schema.

Además del *DN distinguished name* raíz, también existe un *DN* administrador, el cual posee un acceso global sobre el nodo principal y los nodos hijos que se hayan insertado. Este *DN* se deriva del *DN* raíz, por lo que su sintaxis es:

dn: dc=servidores,cn=Admin,dc=unam,dc=mx

cn se utiliza como vocablo y *Admin* como valor, sin embargo, al igual que el *DN* raíz, este vocablo puede variar junto con su valor.

Aunque los *DN* 's raíz y administradores se encuentren definidos en una base de datos (LDBM) es necesario insertarlos antes de realizar cualquier tipo de operación como se encuentra en el siguiente ejemplo:



Nodo-inicial.ldif F2.8.2

El siguiente comando inserta la información a través del archivo anterior dentro del servidor LDAP.

#ldapadd -f Nodo-inicial.ldif -D " cn=Admin,dc=unam,dc=mx" -x -W

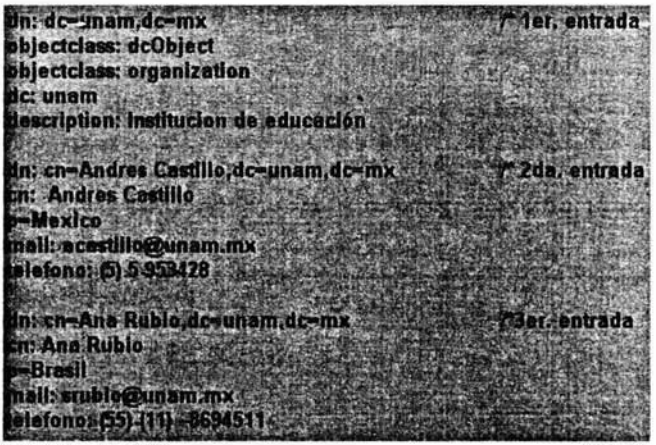
Posteriormente a la inserción aparece el siguiente mensaje en la línea de comandos:

adding new entry dc=unam,dc=mx

el cual nos dice que la inserción fue exitosa, de otra manera, se tiene que revisar el error ocurrido en los registros ("logs").

Se pueden insertar los datos en LDAP de varias maneras, ya sea en un sólo archivo que contenga el nodo principal y los nodos hijos, ó en varios archivos, donde cada uno especifica la posición que le corresponde al nodo en el árbol.

En la figura F2.8.3 se muestra la inserción de tres *DN's* estructurados en un solo archivo LDIF:



tres-entradas.ldif F2.8.3

Para insertar el archivo dentro de nuestro árbol LDAP, se ejecuta el siguiente comando:

#ldapadd -f tres-entradas.ldif -D "cn=Admin,dc=unam,dc=mx" -W

2.9 Búsqueda de datos en LDAP

Ya que los DN's o nodos han sido insertados en la base de datos (LDBM) del servidor LDAP, es posible realizar búsquedas de información sobre ellos mediante el siguiente comando:

ldapsearch [n] [-u] [-v] [-k] [-K] [-t] [-b]

ldapsearch abre una conexión con el servidor LDAP y lleva a cabo una búsqueda utilizando un filtro. Este filtro debe ajustarse a la representación de cadenas de texto para poder filtrar los datos. Si *ldapsearch* encuentra una ó más entradas, se obtienen los atributos especificados y se imprimen por salida estándar los nodos y sus valores. Si no se especifica el atributo que se requiere la búsqueda devuelve por default todos los atributos.

Por Ejemplo:

#ldapsearch -bd 'dc=unam,dc=mx' 'telefono=*52'

El comando anterior busca bajo el DN raíz lo que contenga el atributo teléfono con las cifras 52

#ldapsearch -b 'dc=unam,dc=mx' -x -W

Nos muestra todas las ramas que contiene el nodo unam.mx

2.10 Modificación de datos en LDAP

Para modificar los datos de un nodo del directorio LDAP se utiliza el comando

ldapmodify [-a] [-b] [-c] [-d] [-D] [-x] [-f] [-W]

Este comando abre una conexión con el servidor y modifica o añade entradas. La información sobre la entrada que hay que modificar se lee desde un archivo a través de la opción *-f*

Se muestra un ejemplo de un LDIF que funciona para modificar datos en el directorio LDAP.

```
dn: uid=Andres Castillo,dc=unam,dc=mx
changetype: modify
replace: uid
uid: Rogelio Rojas
-
add: mail
mail: rrojas@unam.mx
-
```

modify.ldif F2.8.4

El comando siguiente modifica el directorio LDAP por medio del archivo anterior.

#ldapmodify -f modify.ldif -D "cn=,Admindc=unam,dc=mx" -x -W

La operación anterior sustituirá el valor *uid* Andrés Castillo por Rogelio Rojas y le añadirá el atributo *mail* con el valor rrojas@unam.mx

2.11 Eliminación de datos en LDAP

Para la eliminación de nodos se necesita el comando

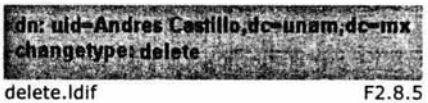
ldapdelete [-n][-v][-k][-c][-d][-D][-f][-W]

Este comando abre una conexión con el servidor LDAP y elimina una o más entradas. Si se proporciona uno o más argumentos en el *dn* se eliminarán las entradas con ese *Distinguished Name* si no se proporciona argumentos de *dn* entonces se leerá desde un archivo con la opción *-f*

#ldapdelete -D "cn=Admin,dc=unam,dc=mx" 'uid=Rogelio Rojas,dc=unam,dc=mx' -x -W

Con el comando anterior se elimina la entrada o nodo Rogelio@unam.mx

También se puede llevar a cabo la eliminación de una entrada a través de un archivo LDIF como se muestra a continuación.



delete.ldif F2.8.5

#ldapdelete -f modify.ldif -D "cn=,Admindc=unam,dc=mx" -x -W

2.12 Replicación de una base de datos LDAP

LDAP tiene la funcionalidad de replicar sus directorios en diferentes equipos, esto para garantizar la seguridad en la información y asegurar un mejor servicio al cliente.
Esto se lleva a cabo través de un servidor maestro con uno o varios servidores esclavos, los cuales, entran en funcionamiento en el momento en que uno de los servidores falle.

Como se mencionó al inicio de este capítulo, el demonio *Slurpd* es el que permite la replicación de bases de datos entre un servidor maestro y uno o varios esclavos; éste es responsable de distribuir los cambios que se realicen en las bases de datos y modificar ya sea los datos de las replicas o del mismo servidor maestro.

Slurpd se encarga también de la sincronización de los datos entre el servidor maestro y sus esclavos, sin embargo se comunican a través de un archivo, que se utiliza para revisar las salidas y encontrar posibles errores en la replicación.

Slurpd tiene la capacidad de enviar las modificaciones del servidor maestro a todas sus réplicas y actualizarlas momentáneamente, para ello se necesita configurar al servidor maestro como a los esclavos.

Para poder realizar una replicación de datos con LDAP se lleva a cabo los siguientes pasos:

1. Tener instalado LDAP en el servidor maestro y en los esclavos.
2. Determinar cual de los servidores será el maestro.
3. No debe estar en funcionamiento LDAP en el servidor maestro ni en sus esclavos (para mayor seguridad).
4. Modificar el archivo slapd.conf del servidor maestro como se muestra en la figura F2.8.6.

```
replogfile    usr/local/var/openldap-ldbm/slurpd.replog
replica       host=máquina.esclavo.com:389
              binddn="cn=Admin,dc=servidores,dc=unam,dc=mx"
              bindmethod=simple credentials=secret
```

Modificación al archivo Slapd.conf para la replicación de datos F2.8.6

Donde,

- *replogfile* es el directorio donde se encuentra la base de datos de LDAP (ldbm).
- *replica* contiene tiene el host con el nombre del servidor esclavo: el número que se muestra después de los dos puntos es el puerto LDAP por donde se realizará la réplica.
- *binddn* es el rootdn (administrador distinghished name). (Se encuentra en el slapd.conf del esclavo).
- *bindmethod* es el método para enlazar los servidores.
- *credential* es el password del ldap del esclavo. (rootpw. Se encuentra en el slapd.conf del esclavo).

Cabe mencionar que el espacio que se encuentra entre el replogfile y la ruta donde se encuentra la base de datos no son espacios en blanco sino un tabulador, éste siempre debe existir entre las líneas de réplica, binddn y bindmethod.

5. Modificar el archivo slapd.conf de los servidores esclavos agregando las siguientes líneas.

```
updatedn      "cn=Admin,dc=unam,dc=mx"
updateref     ldap://ozono.unam.mx
```

Donde,

- *updatedn* es el *rootdn* del slapd.conf maestro.
- *updateref* el nombre del servidor maestro.

6. Debe copiarse la base de datos del maestro a todos sus esclavos, ya que si se tiene alguna diferencia no podrá llevarse a cabo la replicación.
7. Reinicie el proceso slapd del servidor maestro.
8. Reinicie el proceso slapd de cada servidor esclavo.
9. Inicia el proceso slurpd en el servidor maestro con el siguiente comando.

#slurpd -f /usr/local/etc/openldap/slapd.conf

10. Para estar seguros de que la modificación y sincronización de ambos servidores fue satisfactoria ejecute el siguiente comando:

#ps –fea | grep slurpd
ldap 32171 1 0 12:20 ? 00:00:00 /usr/sbin/slurpd

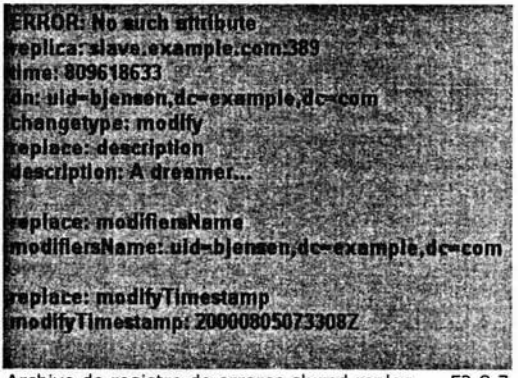
En el momento que se realice alguna modificación a Ldap, ésta se refleja en el archivo log del servidor maestro para posteriormente actualizar los Ldap's esclavos.

Con el siguiente comando se consulta el archivo log:

#more usr/local/var/openldap-ldbm/slurpd.replog

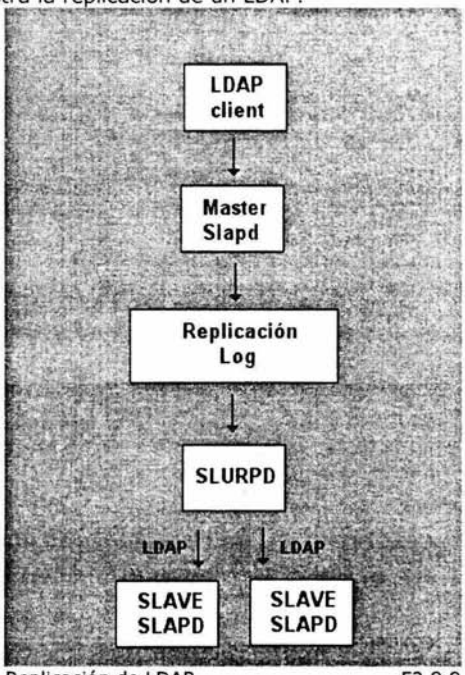
Nota: Recuerde que los Ldap's esclavos deben tener permisos para el acceso remoto a la base de datos.

En caso de que exista un error en la replicación de información, el log file ó archivo de registro de errores, contendrá información como se muestra la figura F2.8.7:



Archivo de registro de errores slurpd.replog F2.8.7

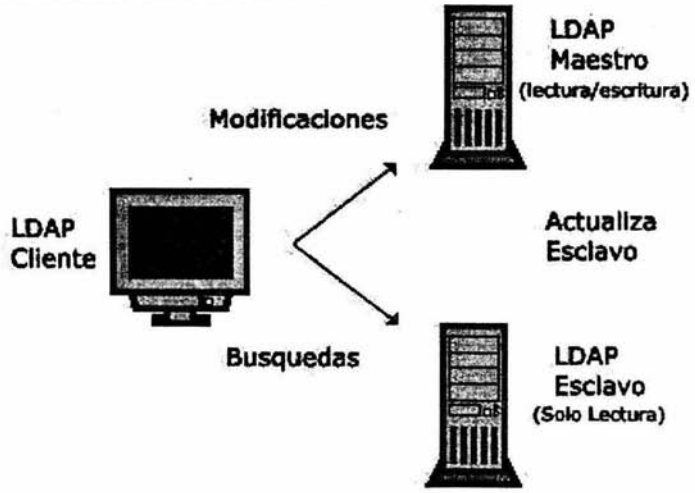
La figura F2.8.8 nos muestra la replicación de un LDAP.



Replicación de LDAP F2.8.8

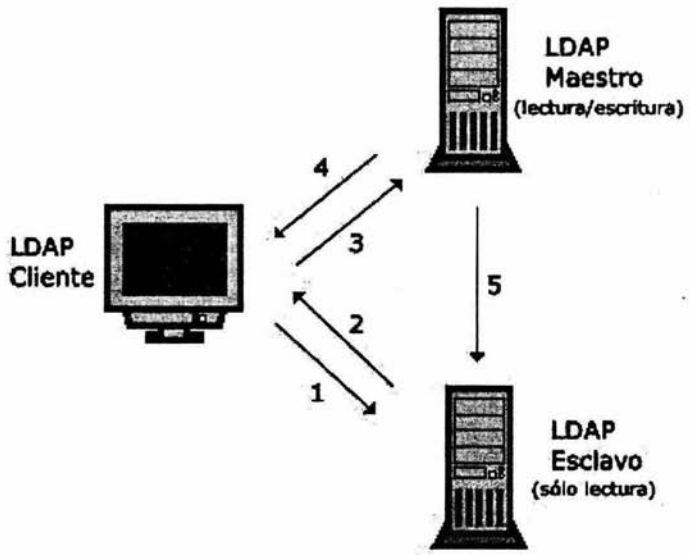
A continuación se mencionan tres formas para llevar a cabo la replicación de bases datos.

- Todas las modificaciones van al servidor maestro



Replicación en LDAP a través de modificaciones al servidor maestro F2.8.9

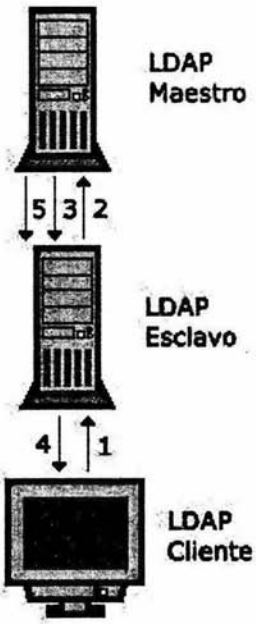
- Utilizando referencias



Replicación en LDAP utilizando referencias F2.8.10

1. Ldap cliente envía modificaciones al esclavo.
2. Ldap esclavo regresa la referencia al maestro.
3. Ldap cliente envía las modificaciones al maestro.
4. Ldap maestro regresa los resultados al cliente.
5. Ldap maestro actualiza la información del esclavo.

- Encadenamiento



Replicación en LDAP en forma de encadenamiento F2.8.11

1. Ldap cliente envía modificaciones al esclavo.
2. Ldap esclavo reenvía la requisición al maestro.
3. Ldap maestro regresa el resultado obtenido al esclavo.
4. Ldap esclavo reenvía el resultado al cliente.
5. Ldap maestro actualiza la información del esclavo.

Las opciones para ejecutar el comando slurpd son las siguientes:

-d Permite que se ejecute slurpd en modo debug
-f Especifica el archivo de configuración para iniciar Slurpd.
-r Especifica el archivo de replicación de slapd. El archivo se debe situar en un directorio con accesos de lectura/escritura/ejecución.
-k Se utiliza para la autentificación de usuarios kerberos en el slapd esclavo.
-o Opera en modo one-shot. En circunstancias normales cuando slurpd termina un proceso de replicación, éste permanece activo y periódicamente revisa para ver si existen nuevas entradas en el archivo log de replicación. Para esta opción se debe especificar el archivo log de replicación con la opción -r.

CAPÍTULO 3

QMAIL (Quick Mail)

En este capítulo se muestra el desarrollo y arquitectura de qmail. Así como el funcionamiento de cada uno de sus módulos.
Se explica la configuración de los archivos de control para obtener los mejor rendimientos y las principales características y ventajas.

3.1 Introducción a QMAIL

Las órdenes cifradas de sendmail y la confusión en su configuración ha ocasionado que algunos administradores de correo electrónico renuncien casi del todo a sendmail. No obstante, estos administradores necesitan aún un MTA que les permita tener variantes sin tener problemas de configuración. Este apartado discute una alternativa disponible a sendmail para el mundo UNIX, llamada Qmail.

Qmail fue escrito por Dan J. Bernstein de Australia. Actualmente es profesor de matemáticas en la Universidad de Ilinois en Chicago, donde es conocido por sus trabajos realizados en el campo de la criptografía.

Bernstein dio a conocer su primer lanzamiento público de Qmail, versión beta 0.70, en enero de 1996, posteriormente liberaría una segunda versión, qmail gamma 0.90, en agosto del mismo año. La primera versión general de qmail, 1.0, se lanzó en febrero de 1997, pero fue hasta el 15 de junio de 1998 cuando se lanzaría la versión (1.03) que a la fecha de este apartado sigue vigente.

Qmail es un MTA (Mail Transfer Agent) que se encarga de transferir el correo electrónico a través de Internet. Se propuso como un sustituto del sistema de correo sendmail en los sistemas UNIX que operan como servidores de correo. Bernstein tenía como objetivo la seguridad, simplicidad, fiabilidad, y eficiencia, de lo cual consideró la seguridad como la de mayor importancia, por lo que el diseño de qmail se realizó para minimizar el uso de *seguid* y *root* (usuario privilegiado en sistemas UNIX), los cuales han sido áreas de constantes ataques en sistemas de correo que utilizan otros MTA.

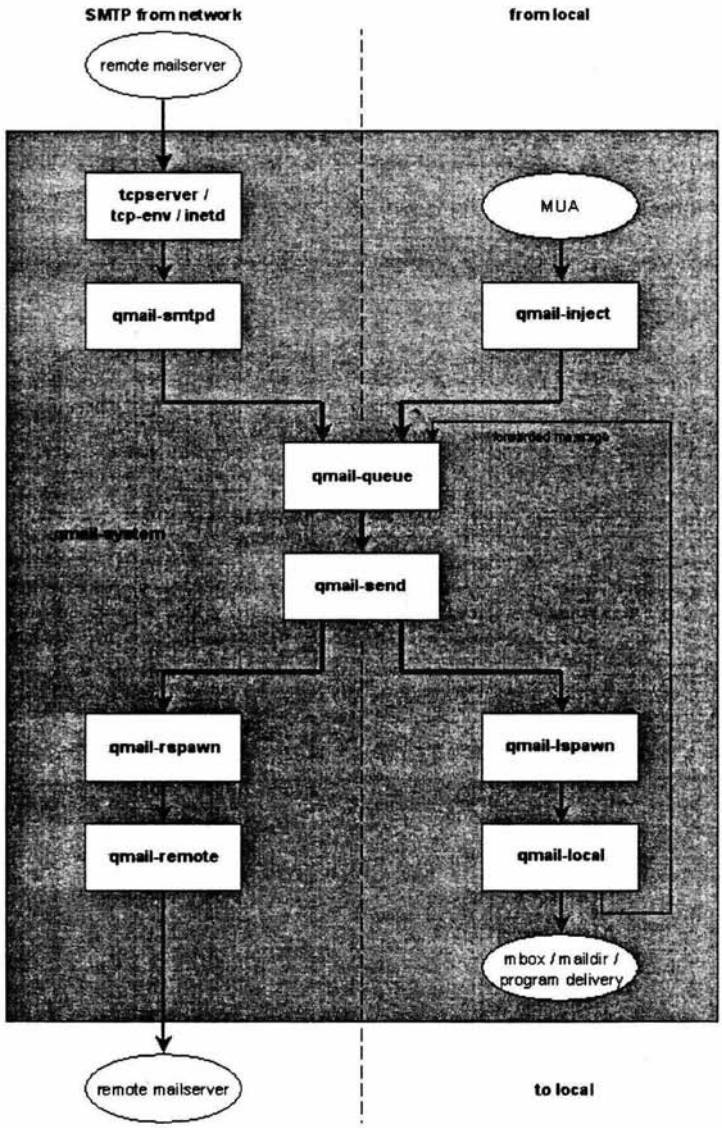
Dentro de qmail se ha tratado con mucho cuidado la gestión de cadenas de caracteres para evitar los problemas de desbordamiento, a menudo, origen de agujeros de seguridad. Un grupo de usuarios ofrece un premio de 1000 dólares a toda persona que encuentre un agujero de seguridad en qmail, premio que hasta la fecha no se ha entregado.

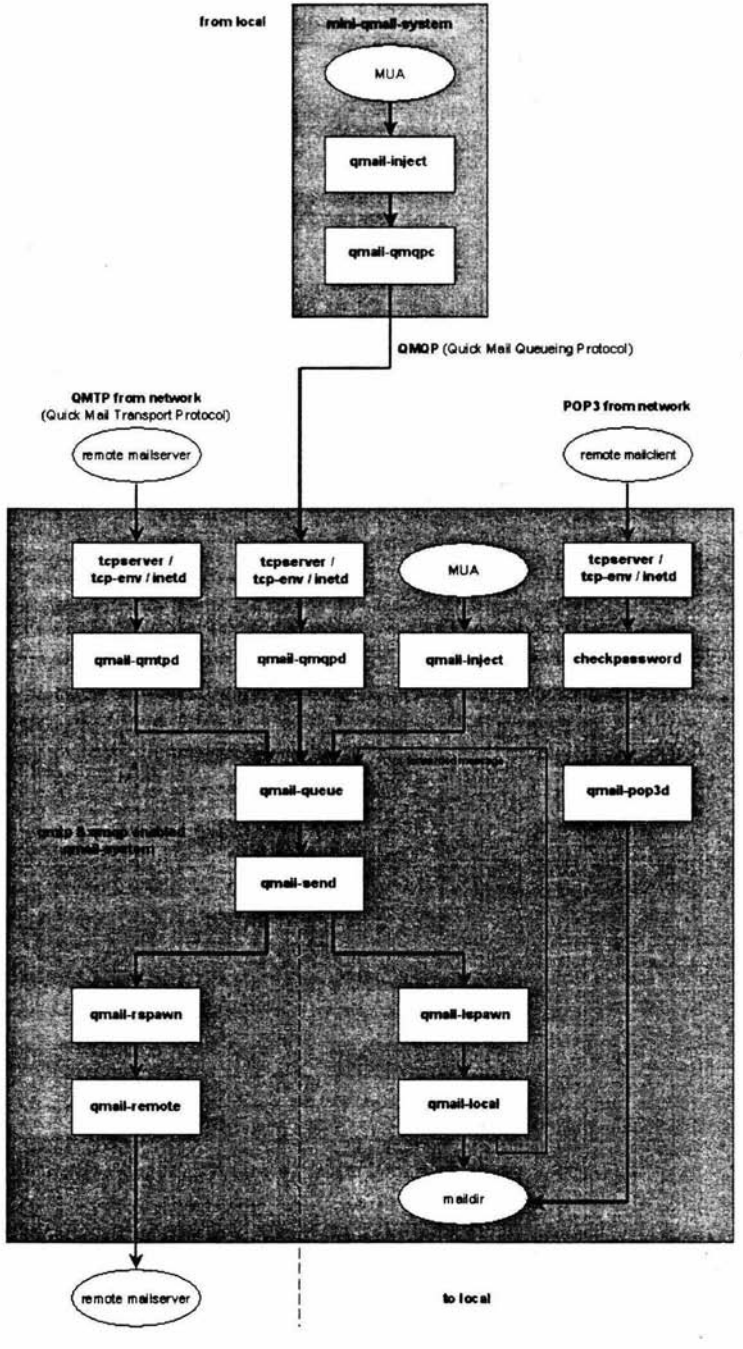
Sendmail ha sido una fuente constante de "bugs" y "vulnerabilidades" para los hackers. Según algunos, esto se debe a que toda la complejidad del sistema de correo recae en un solo programa que necesariamente debe ejecutarse con privilegios de administrador. Qmail intenta eliminar este problema mediante un esquema modular en el cual, diversas etapas del procesamiento del correo electrónico son llevadas a cabo por distintos procesos que mayormente se ejecutan con un usuario no privilegiado.

Qmail está diseñado para ser un sustituto de sendmail. Admite el RFC821 y RFC822, RFC974, RFC1123, RFC1651, RFC1652, RFC1854 y RFC1939 (para los servicios POP3).

3.2 Arquitectura de QMAIL

Imagen del funcionamiento de qmail

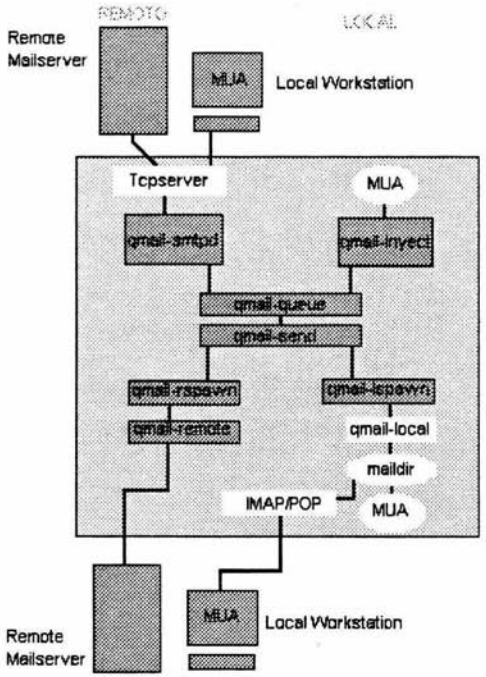



from local
mini-qmail-system
MUA
qmail-inject
qmail-qmqpc
QMQP (Quick Mail Queueing Protocol)
QMTP from network
(Quick Mail Transport Protocol)
remote mailserver
POP3 from network
remote mailclient
tcpserver /
tcp-env / inetd
tcpserver /
tcp-env / inetd
MUA
tcpserver /
tcp-env / inetd
qmail-qmtpd
qmail-qmqpd
qmail-inject
checkpassword
qmail-queue
qmail-pop3d
qmail-send
qmail-rspawn
qmail-lspawn
qmail-remote
qmail-local
maildir
remote mailserver
to local

Como se comentó, qmail no es monolítico, está compuesto por diversos subprogramas o módulos que realizan tareas específicas y que en conjunto constituyen el sistema de correo electrónico.

Para algunos autores existe un inconveniente en el enfoque modular a diferencia del agente de transporte de correo monolítico, las interacciones entre módulos están bien definidas, y los módulos intercambian entre ellos un mínimo de información. Esto generalmente es bueno, sin embargo, puede provocar que sea más difícil de realizar algunas tareas. Por ejemplo, la opción '-v' hace que sendmail imprima un rastreo de sus acciones por salida estándar para permitir la depuración del programa. Puesto que un único binario maneja la inyección, la cola, el procesado de los alias de correo, el procesado de los archivos y el reenvío de mensajes vía SMTP, es capaz de rastrear el envío completo hasta que el mensaje se entrega. No existe capacidad equivalente en qmail, pero si es posible rastrear y asegurar si el mensaje fue enviado satisfactoriamente.

La figura F3.3.2 proporciona una visión más general de los componentes de qmail.



Componentes de Qmail F3.3.2

Note que en la figura anterior la entrega del correo se realiza en formato maildir a través de qmail-local.

Analicemos algunos de los posibles recorridos de un mensaje:

- Origen local: Un usuario operando en el sistema (terminal) remite un mensaje utilizando un MUA como mail. Esto normalmente originará una llamada a *qmail-inyect*, el cual llevará el mensaje al programa de encolamiento *qmail-queue*. Éste lo almacenará en la cola de mensajes /var/qmail/queue. Posteriormente *qmail-send* intentará remitirlo a su destino mediante los programas de *qmail-spam* o *qmail-rspawn* según sea el caso.

- Origen remoto exterior: Un usuario de Internet ha enviado un mensaje hacia nuestra red (un usuario de nuestra organización), este mensaje debe provenir de otro servidor de mail mediante SMTP. *qmail-smtpd* descubre que el destinatario está dado de alta en nuestro servidor, por lo que acepta el mensaje, el cual pasa a la fase de encolamiento.

- Origen remoto de la LAN: Un cliente de correo de nuestra red desea enviar un mensaje. Para esto ha configurado su MUA a fin de remitir hacia el servidor de qmail. Esta remisión normalmente la efectúa usando SMTP, y por lo tanto nuevamente *qmail-smtpd* es el encargado del envío.

- Destino local: Los mensajes con destino local son guardados en el mailbox por *procmail* (o qmail-local) para ser recogidos por el MUA de los usuarios que trabajan con el servidor. En caso de que el usuario no esté conectado con su correo entonces su MUA deberá conectarse a un servidor IMAP o POP para obtener los mensajes. En cualquier caso, *qmail-lspawn* es el encargado de controlar el agente de entrega local.

- Destino remoto: Los mensajes remotos se remiten con SMTP hacia otros MTA. *qmail-rspawn* es el encargado de enviar los mensajes remotamente. Tcpserver aunque no es parte de qmail, es el encargado de comunicarse con *qmail-smtp* de un modo seguro y relativamente inteligente.

3.3 Procmail y QMAIL

Procmail es un Agente de Entrega de Correo (MDA). La función del agente de entrega de correo es aceptar un mensaje que le llegue del MTA para un usuario o buzón de correo dados, y entregar el mensaje de acuerdo con las preferencias del usuario.

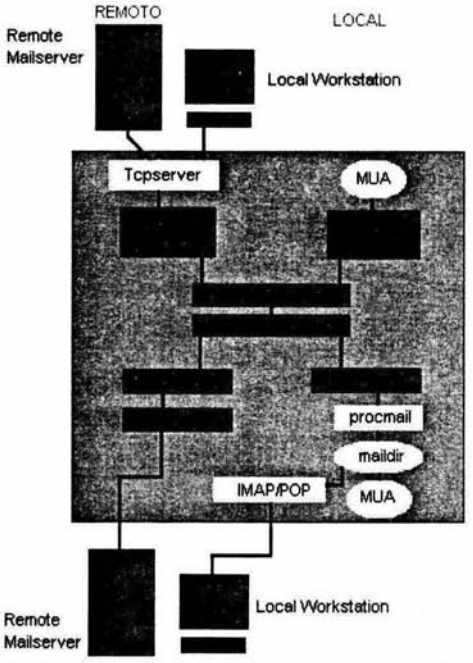
Para que qmail y procmail puedan trabajar juntos es necesario tomar las siguientes consideraciones:

Entre Qmail (MTA) y procmail (MDA) existen algunos problemas para que trabajen juntos. En primer lugar, procmail normalmente está compilado para entregar correo a un buzón de tipo mbox en /var/spool/mail. Qmail utiliza maildir como buzón de correo.

Otro problema es que qmail y procmail no tienen una comprensión común de lo que son los códigos de salida. Procmail usa los códigos de salida estándar de UNIX, en donde cero significa conseguido, no cero significa error, y la causa del error está indicada en /usr/include/sys/errno.h. Por otro lado qmail utiliza otros códigos, no-cero para indicar errores permanentes y el resto se consideran provisionales.

Se puede utilizar un pequeño guión del intérprete de órdenes (shell) como envoltura para traducir los códigos de salida de qmail y procmail. De la misma manera, procmail entregará correos directamente a buzones con formato maildir. Hay un parche disponible que implementa entregas de tipo maildir para procmail (desde la versión 3.14 procmail lleva incorporado el soporte maildir sin necesidad de parches).

La figura F3.4.1 muestra el uso de procmail con qmail.



Qmail y Procmail F3.4.1

3.4 Estructura de archivos binarios

Qmail es un conjunto de archivos binarios situados todos por defecto en */var/qmail/bin.* Están compuestos por una parte de MTA con su faceta SMTP *(qmail-smtpd),* QMTP *(qmail-qmtpd)* e inyección de correos *(qmail-inyect)*, así como una parte de MUA (*qmail-local*).
Mediante *qmail-queue* los mensajes se colocan en la cola, la gestión de ellos lo realizan *qmail-lspawn, qmail-clean* y *qmail-rspawn* respectivamente, todos hijos de *qmail-send.*

A continuación se mencionan los componentes más significativos de qmail.

qmail-queue

Agrega el mensaje en la cola de espera e invoca a qmail-send para el envío.

qmail-send

Toma los mensajes que se encuentran en la cola de espera e invoca a *qmail-lspwan* ó *qmail-rspawn* según sea el caso.

qmail-lspawn

Lleva a cabo la entrega localmente a través de otro binario llamado *qmail-local.*

qmail-local

Realiza la entrega localmente.

qmail-rspawn

Lleva a cabo la entrega remotamente a través de otro binario llamado *qmail-remote.*

qmail-remote

Envía el mensaje remotamente vía SMTP.

qmail-inyect

Lleva el mensaje a *qmail-queue,* quien agrega el mensaje en la cola de espera.

qmail-smtpd

Normalmente trabaja por el puerto 25 donde recibe los mensajes remotos vía SMTP.

qmail-qmqpd

Recibe mensajes remotos vía QMQP sobre el puerto 628. QMQP se utiliza para llevar a cabo la entrega de correos entre clusters.

qmail-popup

Se utiliza para leer usuarios y contraseñas con POP3. Usualmente invoca a auth_pop para la autentificación.

qmail-pop3d

Lo invoca qmail-popup. Maneja las sesiones de POP3 e ingresa a las cuentas de correo pop3 de los usuarios utilizando chekpassword o auth_pop para la autentificación.

auth_pop

Normalmente lo invoca qmail-popup para la autentificación de usuarios. También es responsable de reenviar las conexiones a cualquier miembro del cluster que se tenga configurado en el ldap.

auth_imap

Lo invoca imaplogin en el courier imap. Es responsable de autentificar a los usuarios de IMAP.

qmail-ldaplookup

Es una herramienta que sirve para revisar si el ldap está correctamente configurado.

3.5 Módulos de QMAIL

Los módulos centrales y la estructura estándar del directorio de qmail se muestran en la tabla 3.6.1 y 3.6.2

Módulos Centrales de qmail

Módulo	**Función**
qmail-smtp	Acepta/rechaza mensajes vía SMTP
qmail-inyect	Inyecta mensajes localmente
qmail-rspawn/qmail-remote	Gestión entregas remotas
qmail-lspawn/qmail-local	Gestión entregas locales
qmail-send	Procesa la cola de mensajes
qmail-clean	Limpia la cola de mensajes

Tabla 3.6.1

Directorio /var/qmail

Directorio	**Contenido**
Alias	Archivos .qmail para los alias del sistema
Bin	Binarios y guiones del programa
boot	Guiones de inicio
control	Archivos de configuración
Doc	Documentación (excepto páginas man)
Man	Páginas de ayuda
queue	La cola de los mensajes por enviar
Users	Los archivos de bases de datos de qmail-users

Tabla 3.6.2

He aquí un esquema que muestra la estructura de qmail cuando se carga en memoria:

USUARIO	PROGRAMA	FUNCION
qmails	qmail-send	gestión salida cola de espera
qmaill	_ splogger qmail	registro via syslog
qmailq	_ qmail-clean	limpieza de la cola de espera
qmailr	_qmail-rspawn	gestor de mensajes remotos
root	_ qmail-lspawn	gestor de mensajes locales

3.6 Archivos de Control

La configuración de qmail se lleva a cabo, casi en su totalidad, mediante un conjunto de pequeños archivos situados en el directorio */var/qmail/control/*. Los diferentes programas que componen qmail leen el contenido de estos archivos para configurar el proceso que controla todo el servicio de correo.

Por cada archivo que no exista, se toman sus opciones por defecto. El único archivo que toma el nombre calificado de la computadora de forma predeterminada es el archivo denominado *me*.

El comando *#man qmail-control* proporciona una lista completa de todos los archivos de control y de los programas qmail que los utilizan, así como las opciones que se toman por defecto.

Otra manera de obtener información de los archivos de control es utilizando la orden *qmail-showctl* presente en el sistema qmail.
A continuación se mencionan los archivos de control y su contenido.

me

Contiene el nombre del dominio del servidor de correo.

rcpthosts

Contiene los dominios de qmail-ldap donde llegan los correos.
Permite controlar la retransmisión (relay) de correo por parte de la máquina vía SMTP, es decir, mientras el archivo rcpthosts exista, qmail-smtpd rechazará todo correo cuya parte del dominio del destinatario no se encuentre en este archivo, y al contrario acepta todos los correos en caso que rcpthosts no exista. Es muy importante que rcpthosts autorice al menos a las computadoras que se encuentren dentro del archivo *locals.*

locals

Contiene los dominios por el cual qmail-ldap entrega el correo.

ldapbasedn

Contiene el *distinguished Name* con el cual se configuró ldap.
Nota. Esta información se encuentra en el archivo slapd.conf.

ldapserver

Contiene el (los) host(s) nombre del servidor ldap.

ldaplogin

Contiene el login de autentificación para ldap.
Nota. Esta información se encuentra en el archivo slapd.conf.

ldappassword

Contiene el password de autentificación para ldap.
Esta información se encuentra en el archivo slapd.conf.

plusdomain

Contiene el nombre que el usuario desee para los correos de salida.
Ejemplo: usuario@plusdomain.

defaultdelivery

Contiene el formato para almacenar los correos.

ldapmessagestore

Contiene el path donde se almacenarán los correos del usuario.

ldapgid

Contiene el gid (identificador de grupo) de qmail.

ldapuid

Contiene el uid (identificador de usuario) de qmail.

ldaprebind

Contiene la palabra *ldapobjectclass* junto con cero o un uno, funciona para recobrar el password de un usuario desde el directorio ldap.

ldapdefaultquota

Contiene la quota de correos que recibirán los usuario, puede limitarse por bits ó numero de correos.

ldapwarning

Contiene el mensaje que se desplegará cuando se exceda la cuota.

dirmaker

Contiene la ruta del guión que creará automáticamente los Maildir de los usuarios que se autentiquen en el pop o imap.

Nota. Cabe mencionar que cada vez que sufran alguna modificación estos archivos se debe reiniciar a *qmail-send* enviándole una señal HUP:

 # kill -HUP $(pidof qmail-send)

aliasempty

Contiene el nombre o ruta donde se depositarán los correos, esto mediante un MDA, es decir, *qmail-local* entregará el correo cuando no existe el archivo .qmail. Su valor por defecto es ./Mailbox.

Esta regla de entrega se puede especificar por defecto a *qmail-start* en la línea de comando en el momento de ejecutar qmail.

defaulthost y defaultdomain

Los archivos de control defaulthost y defaultdomain permiten especificar la forma en la que *qmail-inyect* completa las direcciones de destino incompletas. Así, una dirección que no contiene la parte del dominio se complementará a través del defaulthost. El valor por defecto es el mismo que el contiene el archivo de control *me*, y que no corresponde forzosamente al nombre de la máquina.

Las variables de entorno QMAILDEFAULTDOMAIN y QMAILDEFAULTHOST tienen prioridad sobre el contenido de los archivos de control correspondientes.

badmailfrom

badmailfrom es muy útil para evitar el correo entrante no solicitado. Permite especificar las direcciones de remitentes prohibidas. *qmail-smtpd* rechazará todo correo que proceda de ellas. Ejemplo:

```
@contaminacion.com
contaminador@basura.com
```

qmail-smtpd rechazará todo correo que proceda del dominio contaminacion.com o de la dirección contaminador@basura.com.

smtproutes

Permite especificar los puntos de retransmisión (relay) SMTP. Con los que *qmail-remote* deberá contactar para encaminar los correos salientes, ejemplo:

```
primer.dominio:correo.primer.dominio
segundo.dominio:correo.segundo.dominio:24
:correo.otro.dominio
```

Explicando el ejemplo anterior, los correos con destino a primer.dominio se transmiten por SMTP al puerto TCP 25 de correo.primer.dominio, los que tengan destino a segundo.dominio se transmitirán por SMTP al puerto 24 de correo.segundo.dominio. En cuanto al resto de los correos, se transmitirán al puerto TCP 25 de correo.otro.dominio.

3.7 Archivos .qmail

La entrega del correo de un usuario normalmente está controlada por uno o más archivos .qmail los cuales contienen una lista de instrucciones para llevar a cabo la entrega, he aquí un ejemplo de ello:

```
# mi .qmail en mi directorio
# Llegada de correo
|qbiff
# Un forward
&yomismo@otra.cuenta
# Entrega en el buzón maildir
./Maildir/
# Entrega en un buzón mbox
./mbox
```

Como se puede ver en el ejemplo anterior, cada línea contiene una, y sólo una instrucción de entrega. Todas las líneas se analizan para cada correo entregado por *qmail-local*. Pueden contener comentarios prefijados por un octótrofo (símbolo de número #). En este ejemplo, cada correo se enviará por una tubería dada, se reenviará (forwarded) a yomismo@otra.cuenta y se entregará en los buzones de correo ~/maildir/ y ~/mailbox.

A continuación se explica el contenido de las líneas del ejemplo anterior:

- qbiff es un pequeño programa de notificación de llegada de correo con qmail. Anuncia la llegada de un correo por un bip y un pequeño mensaje en la terminal.
- La orden de reenvío (forward) se caracteriza por la anteposición de un ampersand (&). De hecho puede obviarse el ampersand, siempre que la dirección de correo empiece por una letra o una cifra.
- Los nombres de los buzones de correo empiezan siempre por un punto, seguido inmediatamente por una barra inclinada (./). Terminan siempre por una barra inclinada en el caso de los buzones maildir, y nunca terminan con una barra inclinada en el caso de los buzones mbox.

3.7.1 Estructura de un archivo .qmail

Comentarios	Los comentarios se inician con un símbolo de número. `# esto es un comentario`
Programa	Un programa inicia con la línea pipe. `|qbiff` *qmail-local* toma el resto de la línea como un comando como lo hace el comado *sh*
Reenvío	En esta línea reenvía los mensajes.

yomismo@otra.cuenta

qmail-local toma el resto de la línea como una dirección de mail; utiliza *qmail-queue* para reenviar el mensaje a la dirección. La dirección debe contener completamente especificado el nombre de dominio. No debe contener espacios en blanco ni símbolos o comentarios.

En el siguiente ejemplo se especifican algunos errores en la dirección de correo:

&me@new
&<me@new.job.com>
& me@new.job.com
&me@new.job.com (New Address)

Nota: Si la dirección empieza con una letra o un número, puede eliminarse el ampersand (&) al inicio de la línea.

3.8 Servidores POP e IMAP

Pop (Protocolo de oficina de correo) e IMAP (Protocolo de acceso de mensajes de Internet) son protocolos de mensajes basados en Internet. De los dos, POP es el protocolo más antiguo. IMAP fue diseñado para incluir las capacidades de POP y agregar soporte a los modos en línea y fuera de línea o desconectado. POP e IMAP se definen en RFC de IEFT; POP3 se define en RFC1939 e IMAP4 en RFC1730.

POP se diseño para soportar procesamiento de correo fuera de línea. En el paradigma fuera de línea, el correo se entrega a un servidor y carga todo el correo pendiente en la máquina local del usuario. Todo el procesamiento de esos mensajes es local en la máquina cliente de correo.

El modo fuera de línea funciona como un servicio "almacenar y seguir", donde el correo se transfiere a la máquina del cliente cuando se solicita desde el servidor de correo o se conecta a una máquina cliente de correo como único destino, normalmente una computadora personal. Una vez entregados al cliente de correo, los mensajes se eliminan del servidor.

IMAP también puede procesar correos fuera de línea, pero la función principal de IMAP son los modos de operación en línea. En el modo en línea, el correo también se entrega al servidor de correo, pero el cliente de correo no carga todos los mensajes de una vez, actúa al estilo cliente-servidor, el cliente puede pedirle al servidor sólo las cabeceras de los mensajes o los cuerpos de los mensajes seleccionados o buscar los mensajes de acuerdo a un cierto criterio. Los mensajes de correo que se encuentran en el servidor se pueden marcar con varias etiquetas de estado, como "leídos" o "no leídos" y permanecen en el servidor hasta que el usuario los elimine.

IMAP está diseñado para permitir la manipulación de buzones de correo remotos como si fueran locales al usuario. Dependiendo de la implementación del cliente de correo IMAP y del comportamiento del correo definido por el administrador de correo, el usuario puede almacenar directamente los mensajes en la máquina cliente o guardarlos en el servidor, según lo desee.

Los agentes de correo, fuera de línea y en línea, permiten acceder a nuevos mensajes entrantes en el servidor de correo desde diversos clientes de correo. Sin embargo, los modos fuera de línea y en línea encajan con distintos requisitos y estilos de utilización.

El modo en línea se aplica mejor en usuarios que utilizan rutinariamente una única máquina cliente, más no encaja en usuarios que desean acceder a sus mensajes recientes desde máquinas diferentes cada vez. Debido a que el modo en línea es equivalente a la carga y eliminación del correo desde el servidor, el uso de diferentes máquinas cada vez, ocasiona que el correo se disperse entre diferentes escritorios. Esto no es del todo cierto si se trabaja en entornos distribuidos como DCE (entorno de computación distribuida) y se utiliza un sistema común de archivos de red.

Las diferencias entre los modos de acceso en línea y fuera de línea se pueden resumir de la siguiente forma.

El modo en línea es una obtención solicitada por el usuario a una única máquina cliente con las siguientes características:

- Utilización mínima del tiempo de conexión.
- Utilización mínima de los recursos del servidor.

En el modo fuera de línea se tienen las siguientes características:

- Acceso interactivo a múltiples buzones de correo desde múltiples clientes.
- Capacidad para diferentes computadoras.
- Capacidad para utilizar máquinas cliente "sin datos".
- Acceso independiente de la plataforma a múltiples buzones de correo.
- Posibilidad de accesos concurrentes a buzones de correo compartido.

La posibilidad de acceder a los mensajes entrantes y a las carpetas de archivos de mensajes desde diferentes computadoras cada vez, puede ser una ventaja sin importancia para aquellos que utilizan el mismo escritorio para acceder a su correo, pero también es una ventaja significativa para aquellos que utilizan múltiples computadoras.

POP e IMAP comparten varias características comunes como son:

- Ambos pueden admitir operación fuera de línea.
- El correo necesita que se entregue a un servidor de correo compartido "siempre activo".
- El correo nuevo está accesible desde diversos tipos de plataformas de cliente y también está accesible desde cualquier lugar de la red.
- Ambos están definido en las RFC de Internet como protocolos abiertos.
- Los clientes están disponibles para múltiples plataformas, de forma gratuita y comercial.
- Los protocolos sólo tratan con acceso a correo, ambos dejan a SMTP en el envío de correo.

La principal ventaja de POP es que es un protocolo más simple que IMAP y por tanto más sencillo de implementar. IMAP puede acceder y administrar múltiples buzones de correo en el mismo ó en diferentes servidores al mismo tiempo. La capacidad de IMAP de permitir actualizaciones concurrentes y accesos a buzones de correos compartidos es muy útil, si múltiples individuos están procesando mensajes entrantes en un buzón de entrada común, cualquier modificación o cambio de algún usuario estará presente para otros usuarios de correo activos en tiempo real.

IMAP también es bueno para el acceso a datos sin formato de correo como Usenet o documentos. Finalmente, como ya se menciono, IMAP también puede operar en modo fuera de línea para minimizar el tiempo de conexión y/o el impacto sobre el servidor.

Un ejemplo entre POP e IMAP sería, si llega un mensaje con un archivo de video con varios megabytes adjunto a un mensaje de texto con una línea que dice, "Observa la demostración", POP enviaría el mensaje y el archivo adjunto. Utilizar un cliente IMAP, permite terminar de transferir la cabecera o el cuerpo del texto, sin el archivo adjunto de video, dando la posibilidad de decidir al cliente si se desea esperar la carga de video.
Dentro de qmail se incluye un servidor POP, qmail-pop3d, el cual no se instala como parte del proceso de instalación de qmail.

3.8.1 Maildir

Qmail soporta *maildir*, un nuevo formato de buzón de correo de usuario muy fiable.
Los directorios *Maildir* a diferencia de los archivos *mbox* y carpetas *mh*, no se corrompen si el sistema sufre una caída durante el envío, aún mejor, no sólo puede un usuario leer con seguridad su correo a través de NFS, sino que cualquier número de clientes NFS pueden entregarle correo simultáneamente.

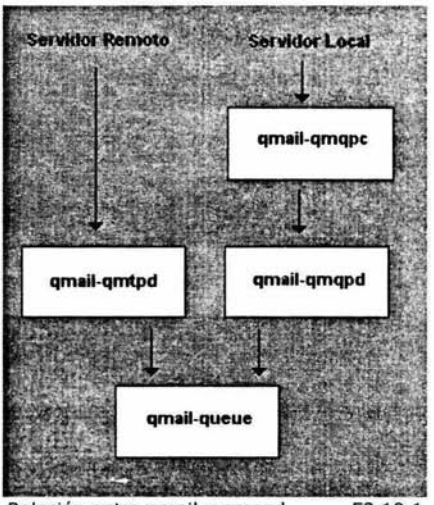
¿Que sucede si al momento de anexar un mensaje al archivo mailbox el sistema falla? Seguramente el archivo se corrompe, para eso existe maildir. Maildir está compuesto por tres subdirectorios (new, current y tmp). Cada correo nuevo se deposita en el directorio new, que a diferencia con Mailbox, es un archivo diferente, esto permite que los correos se puedan cambiar entre directorios, dependiendo del status del correo (nuevo, leído, no leído), se sabe a que subdirectorio pertenece, lo que también ayuda a que los respaldos se realicen sólo en archivos que no se han sido respaldados con anterioridad. Estas características hacen de maildir un formato más potente y con mucho más flexibilidad en el manejo de correos.

3.9 QMTP

Está diseñado por Dan J. Bernstein, es un protocolo rápido de transferencia de correo (Quick Mail Transfer Protocol) que sustituye al protocolo simple de transferencia de correo (SMTP).
QMTP es similar a SMTP, pero más sencillo y rápido. QMTP incluye un servidor llamado *qmail-qmtp* que trabaja sobre el puerto 209 y se ejecuta de forma similar a SMTP, aunque es incompatible con éste.

Qmail no incluye un cliente de QMTP, pero sí incluye un paquete llamado maildir2qmtp, el cual toma un buzón de correo maildir y entrega los mensajes mediante el protocolo QMTP a un servidor QMTP designado, esto generalmente se realiza cuando se tiene un cluster configurado entre varias máquinas.

QMTP no es un sustituto inmediato de SMTP, su uso no es tan extendido en Internet, pero es una alternativa futura para un servicio de transferencia de correo. La figura F3.10.1 muestra el funcionamiento de los componentes de qmail y qmtp.



Relación entre qmail y qmqpd F3.10.1

3.10 Características más importantes de QMAIL.

Configuración

- Adaptación automática a cualquier variante de UNIX; no es necesario trasladar la aplicación.
- Configuración automática para cada máquina.
- Instalación rápida; no hay una larga lista de decisiones que tomar.

Seguridad

- Separación nítida entre direcciones, archivos y programas.
- Reducción al mínimo del código que se ejecuta con setuid.
- Reducción al mínimo del código que se ejecuta como root.
- Registro opcional de hash unívoca para cada mensaje.

Construcción de Mensajes

- Cumple con los *RFC 822* y *RFC 1123.*
- Soporte completo para grupos de direcciones.
- Conversión automática de listas de direcciones en formato antiguo al formato RFC 822.
- Orden *sendmail* para la compatibilidad con los programas clientes de correo de los usuarios.
- Longitud de línea de la cabecera limitada sólo por la memoria disponible.
- Enmascaramiento de máquina.
- Enmascaramiento de usuario.

Servicio SMTP

- Cumple con *RFC 821, RFC 1123, RFC 1651, RFC 1652, y RFC 1854.*
- Transmite mensajes con codificación 8-bit.
- Control de *Relay* (retransmisión): acaba con los reenvíos o retransrnisiones que usuarios externos sin autorización hacen gracias a nuestra máquina.
- No hay interferencia entre el control de *relay* (control de retransmisión) y los alias de correo.
- Reconocimiento automático de direcciones IP locales.
- Caducidad del plazo de envío individual para cada búfer o almacenamiento temporal.
- *Hop counting* (limitación en el número de cabeceras que tiene un mensaje).
- Limitación de paralelismo (a través de *ucspi-tcp*)
- Posibilidad de denegar conexiones a usuarios que se sepa que han abusado (a través de *ucspi-tcp*).
- Retransmisión (*relaying*) y reescritura de mensajes para clientes autorizados.
- Soporte opcional de RBL/ORBS.

Gestión de colas

- Tratamiento instantáneo de los mensajes que se agregan a la cola.
- Límites al paralelismo.
- División del directorio de la cola (no disminuye el rendimiento del sistema cuando la cola aumenta).
- Programación de reintentos: cuanto más antiguo es el mensaje, con menor frecuencia se intenta su reenvío.
- Planificación independiente de reintentos para mensajes.
- Encolamiento automático seguro (no se pierde correo en caso de que el sistema falle).
- Limpieza automática de colas de correo.
- Consulta de colas de mensajes con qmail-qread.

Devoluciones (*bounces*)

- Mensajes de devolución legible para la máquina y legible para el usuario.
- Soporte HCMSSC: códigos de error según RFC 1893 independientes del idioma RFC 1893.
- Devoluciones enviadas al administrador de correo.

Rutados por dominio

- Cualquier número de nombres para la máquina local.
- Cualquier número de dominios virtuales.
- Comodines para los dominios: subdominios enteros pueden gestionarse con una única instrucción de rutado.

Entrega SMTP

- Cumple con RFC 821, RFC 974, y RFC 1123.
- Transmite mensajes con codificación 8-bits.
- La transmisión a un servidor que no responde se detiene o acelera automáticamente.

- Expiración del plazo de envío individual para cada búfer o almacenamiento temporal.

Qmail admite listas de correo, facilitando mucho a los usuarios locales la configuración de sus propias listas por medio de un mecanismo generalizado de reenvío. Aquí se muestran algunas características notables:

- Permite a cada usuario la gestión de sus propias listas.
- Facilita la designación de propietarios (owners) de listas de distribución.
- Evita automáticamente los bucles entre listas de correo, incluso entre distintas máquinas.
- Permite listas de correo de tamaño inconcebibles.
- Gestiona los alias y el reenvío con un solo mecanismo simple. (La detección de los bucles también aplica a los alias.
- Soporta el gestor de listas de correo ezmlm, que de forma sencilla y automatizada se encarga de las devoluciones, peticiones de suscripción y archivado.
- Compatibilidad con archivos .forward de Sendmail (por medio de dot-forward).
- Compatibilidad con el archivo /etc/aliases de Sendmail (por medio de fastforward).
- Comodines para direcciones (qmaildefault).
- Administradores de listas de correo: desvío automático de devoluciones y mensajes de ausencia por vacaciones.
- Cabeceras Delivered-To: prevención automática de bucles, incluso entre máquinas.

Entrega local de correo

- Entrega de correo fiable en NFS.
- Programa de entrega de correo controlado por el usuario: procmail o qmail-local.
- Notificación de nuevo correo opcional (qbiff).
- Acuses de recibo *NRUDT* opcionales (qreceipt).
- Filtrado condicional (condredirect y bouncesaying)

Servicio POP3

- Cumple con RFC 1939
- Soporte UIDL
- Soporte TOP
- Es posible añadir características para el soporte APOP
- Comprobación modular de contraseñas (por medio de checkpassword o auth_pop)

CAPÍTULO 4

Instalación y Funcionamiento de LDAP con Qmail

En este capítulo se explicará la instalación de qmail con ldap y el funcionamiento interno de ambos. Así mismo, el uso de algunas herramientas para su óptimo funcionamiento.

4.1 Requerimientos para la instalación de LDAP con QMAIL

Para llevar a cabo la configuración y ejecución de Qmail-LDAP es necesario del siguiente software:

- daemontools
- ucspi-tcp
- tcp wrappers (opcional)
- dbjdns
- Openldap
- qmail

Se mencionan los pasos para llevar a cabo la instalación de las herramientas necesarias dando sólo una breve descripción de cada una de ellas.

4.1.1 Daemontools

Daemontools son un grupo de programas similares en concepto al demonio inetd pero más potente; daemontools se encarga de iniciar un servicio cuando se detecte un acceso a los puertos de su protocolo asociado, también se encarga de monitorear el estado del servicio a nivel de procesos; si algún servicio muere de forma inesperada, daemontools se encarga de volverlo a iniciar.

Daemontools trabaja con los siguientes comandos:

supervise. Es el que se encarga de monitorear los servicios, iniciarlos, y en caso de que terminen los reinicia. Para iniciar un servicio todo lo que *supervise* necesita es un directorio con un guión de ejecución (script) que sepa iniciarlo.

multilog. Almacena los mensajes de error en uno o varios archivos de registro. Puede etiquetar cronológicamente cada línea y puede incluir o excluir líneas que sigan ciertos patrones que le sean especificados para cada archivo.
Rotan automáticamente los archivos para limitar la cantidad de espacio en disco utilizado. Si el disco se llena, se detiene y reintenta cada cierto tiempo sin perder dato alguno.

svscan. Arranca un proceso de supervisión por cada subdirectorio del directorio actual, si encuentra un nuevo directorio, arranca otro proceso que supervise hasta un límite de 1000 subdirectorios.
Normalmente *svscan* se inicia a través del directorio /service.

svc. Controla servicios que monitorea *supervise.*
Utiliza las siguientes opciones:

- -u: (*Up*) Arrancar. Si el servicio no está ejecutándose, lo inicia. Si el servicio se detiene, lo reinicia.
- -d: Detener. Si el servicio está ejecutándose, envía una señal para terminar su ejecución.
- -o: (*Once*) Una sola vez. Si el servicio no está ejecutándose, lo inicia. No lo reinicia si se detiene.
- -p: Pausa. Envía al servicio una señal STOP

- -c: Continuar. Envía una señal CONT.
- -h: (*Hangup*) Reinicia. Enviar al servicio una señal HUP.
- -a: Alarma. Envía una señal de ALARM.
- -i: Interrumpir. Envía una señal INT.
- -t: Terminar. Envía una señal TERM.
- -k: (*Kill*) Matar. Envía una señal KILL.
- -x: (*Exit*) Salir. supervise detendrá su ejecución tan pronto como detenga al servicio

svstat. Imprime el estado de los servicios monitoreados por *supervise*

svok. Comprueba si *supervise* se está ejecutando.

4.1.2 Instalación de daemontools.

La instalación se hará como usuario root ó superusuario. Cabe mencionar que daemontools funciona sólo bajo UINIX.

Paso 1. Descargue la distribución de daemontools

daemontools-0.61.tar.gz

Paso 2 .Descomprima la distribución de daemontools

#gunzip daemontools-0.61.tar.gz
#tar -xvf daemontools-0.61.tar

Paso 3. Por omisión se instala en */usr/local* pero puede instalarse en cualquier otro directorio.

#cd /admin
#cd /daemontools-0.61

Paso 4. Compilación e Instalación de los programas.

#package/install

4.1.3 UCSPI-TCP

Qmail necesita de un mecanismo que enlace el demonio qmail-smtpd cada vez que llega un intento de conexión SMTP del exterior al servidor de correo. Esto se puede hacer configurando inetd e incluso xinetd para este fin, sin embargo, los creadores recomiendan el uso del programa tcpserver que está disponible como parte del paquete ucspi-tcp de D.J. Bernstein.

Las herramientas o comandos en línea para el uso Ucspi-tcp en aplicaciones cliente-servidor TCP son las siguientes:

tcpserver. Espera conexiones entrantes, y ejecuta un programa por cada conexión. El programa recibe variables de entorno mostrando el nombre del sistema local, del sistema remoto, las direcciones IP y los puertos.

Ofrece un límite de concurrencia para proteger y no agotar el número máximo de procesos o memoria disponible. Cuando se gestionan 40 conexiones simultaneas (por defecto) tcpserver va posponiendo fluidamente las nuevas conexiones.
Este paquete incluye una herramienta que monitorea todas las entradas/salidas de un servidor.

tcpclient. Establece una conexión TCP y ejecuta un programa de su elección con las mismas variables que tcpserver. Este paquete incluye varios ejemplos de clientes elaborados sobre *tcpclient* como who, date, finger, http, tcpcat, y mconnect, los cuales emplean el protocolo ucspi-tcp. Existen herramientas UCSPI para distintos tipos de redes.

UCSPI-TCP compite ahora con varios programas como son:

- inetd, un servidor TCP utilizable sólo por root, distribuido por todos los vendedores de UNIX;
- xinetd, un sustituto de inetd, no funciona en Solaris.
- el cliente mconnect que se incluye como parte de SunOS;
- el programa socket;
- faucet y hose, parte del paquete netpipes;
- El programa netcat que también soporta UDP.

Muchos servidores están sustituyendo inetd por tcpserver, por varias razones:

- inetd es poco fiable bajo grandes cargas. Suspende el servicio si recibe demasiadas conexiones en el lapso de un minuto.
- inetd no proporciona un mecanismo de control de consumo de recursos efectivo. Agotará toda su memoria si ofrece algún tipo de servicio que sea popular.
- inetd tiene problemas para los aumentos súbitos de actividad. Su «lista de tareas pendientes» para *listen()* abarca 5 o 10 como máximo, y no puede incrementarse.

4.1.4 Instalación de UCSPI-TCP

El programa ucspi-tcp-0.88.tar.gz se puede bajar de la página

http://cr.yp.to/ucspi-tcp/install.html

Para instalar UCSPI-TCP debe desempacarse en algún directorio, es recomendable que se realice en el directorio /usr/local.

Paso 1. Se procede a desempacar el programa con el siguiente comando:

#gunzip ucspi-tcp-0.88.tar
#tar -xvf ucspi-tcp-0.88.tar

Paso 2. Se ingresa a la carpeta que se genera con el nombre de ucspi-tcp-0.88/

#cd ucspi-tcp-0.88/

Paso 3. Se compila el programa con el siguiente comando:

#make

Se crean una serie de ejecutables en el mismo directorio. Se recomienda copiar los ejecutables tcpserver y tcprules a un directorio en el PATH, como /usr/sbin o /usr/local/bin.

Paso 4. Para instalarlo debe ser con el superusuario (root), ejecutando el siguiente comando:

#make setup check

4.2 Instalación y Funcionamiento de LDAP con QMAIL

Se debe tener previamente instalado y en funcionamiento el siguiente software.

- daemontools
- ucspi-tcp
- openLDAP

4.2.1 Creación de las cuentas para los usuarios virtuales de correo

Se edita el archivo /etc/passwd y se añade lo siguiente:

 # vmail:x:11184:2110::/var/qmail/maildirs/:/bin/true

Se edita el archivo /etc/group y se añade lo siguiente:

vmail::2110

4.2.2 Proceso de compilación

Es necesario contar con la última versión de qmail y el último parche de qmail-LDAP.
La versión de qmail para este apartado es qmail-1.03.tar.gz, la puedes obtener desde http://cr.yp.to y el parche pertinente desde http://www.nrg4u.com

El archivo *qmail.schema* se encuentra en el directorio donde se desempacó el qmail-1.03.tar.gz. Este archivo se debe copiar en el directorio donde se encuentran los demás *schemas*, así como agregarlo en el archivo slapd.conf de LDAP.

A continuación se descomprime el paquete qmail-1.03.tar.gz.

 # gunzip qmail-1.003.tar.gz
 # tar -xvf qmail-1.03.tar

Se copia el archivo qmail.schema en la ruta donde se encuentran los demás schemas, habitualmente (/etc/openldap/schemas). No se olvide de agregar el archivo en el slapd.conf del openldap.

Se le quita el zip al parche

 # gunzip qmail-ldap-1.03-20010501.patch.gz

Lo siguiente es copiar el qmail-ldap-1.03-20010501.patch en el directorio qmail-1.03 donde se aplica el parche.

 # cp qmail-ldap -1.03-20010501.patch ~/qmail-1.03

4.2.3 Aplicación del parche y compilación

Después de copiar el parche dentro del directorio *qmail-1.03/* se ejecutan los siguientes comandos:

#cd qmail-1.03/
#patch -p1 < qmail-ldap-1.03-20010501.patch

Antes de compilar se edita el Makefile y se añade la línea *-lresolv*, como se muestra a continuación:

LDAPLIBS=-R/usr/local/lib -lldap -llber –lresolv

Después comentaremos las siguientes líneas:

```
# TLS (SMTP encryption) in qmail-smtpd and qmail-remote, see TLS.readme
# You need OpenSSL for this
# TLS enable
#TLSON=-DTLS
# Path to OpenSSL includes
#TLSINCLUDES=-I/usr/local/include
# Path to OpenSSL libraries
#TLSLIBS=-L/usr/local/lib -lssl -lcrypto
# Path to OpenSSL binary
#OPENSSLBIN=/usr/local/bin/openssl
```

Al comentar las líneas se dice que no se quiere cifrado SSL para el SMTP, y sólo en caso de que se requieran se eliminará el comentario.

4.2.4 Creación de los usuarios de QMAIL

Se crean los usuarios de qmail, que es el último paso antes de la compilación.

 # groupadd nofiles
 # useradd -g nofiles -d /var/qmail/alias alias
#useradd -g nofiles -d /var/qmail qmaild
 # useradd –g nofiles -d /var/qmail qmaill
 # useradd -g nofiles –d /var/qmail qmailp
 # groupadd qmail
 # useradd -g qmail –d /var/qmail qmailq

 # useradd -g qmail -d /var/qmail qmailr
 # useradd -g qmail -d /var/qmail qmails

Ahora sólo queda compilar y observar si hay posibles errores en la compilación.

 # make setup check

Una vez terminado, deja por defecto todo instalado en el directorio /var/qmail.

4.2.5 Configuración básica de QMAIL

Se crean los alias necesarios, para ello ejecutamos los siguientes comandos en el directorio /var/qmail/alias:

 # touch .qmail-postmaster .qmail-mailer-daemon .qmail-root
 # chmod 644 ~alias/.qmail*

Después se crea un directorio llamado /rc

 # mkdir /var/qmail/rc

Se copia el archivo *home* que se encuentra en /var/qmail/boot/home a ~/rc

 # cp /var/qmail/boot/home /var/qmail/rc/

Se edita el archivo /var/qmail/rc/home y se cambia la palabra Mailbox por ./Maildir/, como se muestra a continuación:

qmail-start ./Maildir/ splogger qmail

4.2.6 Preparación de los archivos de control

Para el correcto funcionamiento y configuración del qmail es necesario crear en */var/qmail/control* los siguientes archivos con su contenido.

me

tu.dominio.com

rcpthosts

tu.dominio.com

locals

tu.dominio.com

ldapbasedn

Nota: Esta información se encuentra en el archivo slapd.conf
"dc=tu,dc=dominio,dc=com"

ldapserver

Por default *"localhost"*

ldaplogin

"cn=Admin,dc=tu,dc=dominio,dc=com"

ldappassword

password de autentificación para ldap

El password debe estar en texto plano por lo que se tiene que cambiar los permisos con el siguiente comando:

#chmod 600 ldappassword

ldapdefaultdomain

tu.dominio.com

plusdomain

tu.dominio.com

defaultdelivery

./Maildir/

ldapmessagestore

/var/qmail/maildirs

ldapgid

2110

ldapuid

11184

ldaprebind

ldapobjectclasss 1

ldapdefaultquota

1000000S, 1000C (1000000S=1Mb, 1000C=1000 mail)

ldapwarning

Texto a elegir

dirmaker

Se crea el archivo *dirmaker*

 # vi /var/qmail/dirmaker

El archivo dirmaker contendrá lo siguiente:

```
# !/bin/sh
mkdir -m 700 -p $1
```

Una vez editado se guarda el archivo y se le proporciona permisos de ejecución:

 # chmod a+x /var/qmail/dirmaker

4.2.7 Creación de los guiones de arranque con qmail-conf

Es necesario de los siguientes paquetes para la configuración de qmail-conf:

qmail-conf-0.60.tar.gz
djbdns-1.05.tar.gz

Se sitúa en el directorio donde se tiene los dos tar.gz y se realiza los siguientes pasos:

```
# gzip -dc qmail-conf-0.60.tar.gz | tar xf -
# cd qmail-conf-0.60
# gzip -dc ../djbdns-1.05.tar.gz | tar xf -
# make -f Makefile.ini djbdns=djbdns-1.05
# make
# make setup check
```

Por lo regular los archivos binarios se quedan /var/qmail/bin. Para su ejecución se debe introducir en el directorio donde se encuentra el binario y anteponer los caracteres ./, por ejemplo:

 # ./qmail-delivery-conf

Una vez compilado se crean los guiones de arranque con las herramientas de qmail-conf.
Se crea el directorio donde se localizarán los servicios

 # mkdir /var/qmail/service/

Se accesa al subdirectorio donde se encuentran los binarios y se ejecuta el siguiente comando:

#./qmail-delivery-conf qmaill /var/qmail/service/qmail

Esto creará el directorio de servicio.

Se edita el archivo ./run que se encuentra en /var/qmail/service/qmail y se añade lo siguiente:

```
#!/bin/sh
exec /var/qmail/rc/home
```

Finalmente se hace el enlace al directorio de servicio */service*.

 # ln -s /var/qmail/service/qmail /service

4.2.8 Configuración para iniciar el servicio de POP3

Ejecute los siguientes comandos:

```
#./qmail-pop3d-conf/var/qmail/bin/auth_pop qmaill
/var/qmail/service/pop3d
# cd /var/qmail/service/pop3d
# ./add-client 127.0.0.1 192.168.1
```

Dónde *127.0.0.1* es la IP local del equipo y *192.168.1* es la IP o subred del servidor.

Posteriormente ejecute los siguientes comandos:

```
# make
# ln -s /var/qmail/service/pop3d /service
```

Nota: en caso de que se necesite acceder al pop3 desde cualquier IP, edite el tcp y déjelo sólo con la siguiente línea:

```
:allow
```

4.2.9 Configuración para iniciar el servicio de SMTP

Ejecute los siguientes comandos:

```
# ./qmail-smtpd-conf qmaild qmaill /var/qmail/service/smtpd
# cd /var/qmail/service/smtpd
# ./add-relayclient 127.0.0.1 192 .168.1
```

Nota: Como ejemplo, le decimos al tcpserver que sólo deje hacer RELAY a la subred 192.168.1. y a loopback.

```
# make
# ln -s /var/qmail/service/smtpd /service
```

4.2.10 Configuración para iniciar el servicio de QMQPD.

Ejecute los siguientes comandos:

```
# ./qmail-qmqpd-conf qmaild qmaill /var/qmail/service/qmqpd
# cd /var/qmail/service/qmqpd
# ./add-client 127.0.0.1 192.168.1
# make
# ln -s /var/qmail/service/smtpd /service
```

4.2.11 Configuración para iniciar el servicio de IMAPD.

Para configurar el servicio de imap es necesario que éste se encuentre debidamente instalado.

Para configurarlo ejecute los siguientes comandos:

```
#cd /var/qmail/service
#cp -r pop3d /imapd
#cd imapd
```

Se debe editar el archivo y cambiarlo como a continuación se muestra:

```
#vi run

!/bin/sh
exec 2>&1 \
envdir ./env \
sh -c '
   case "$REMOTENAME" in h) H=;; p) H=p;; *) H=H;; esac
   case "$REMOTEINFO" in r) R=;; [0-9]*) R="t$REMOTEINFO";; *) R=R;; esac
   exec \
   softlimit ${DATALIMIT+"-d$DATALIMIT"} \
   /usr/local/bin/tcpserver \
      -vD"$H$R" \
      ${LOCALNAME+"-l$LOCALNAME"} \
      ${BACKLOG+"-b$BACKLOG"} \
      ${CONCURRENCY+"-c$CONCURRENCY"} \
      -xtcp.cdb \
      -- "${IP-0}" "${PORT-143}" \
   /usr/local/courier/sbin/imaplogin \
   /var/qmail/bin/auth_imap \
   /usr/local/courier/bin/imapd "${MAILDIRNAME-Maildir}"
'
```

Continuamos con los siguientes comandos:

```
#cd env
#vi PORT

PORT:143
```

```
# cd /var/qmail/service/imapd
# ./add-relayclient 127.0.0.1 192 .168.1
# make
# ln -s /var/qmail/service/imapd /service
```

4.2.12 Inicio, terminación y reinicio de servicios.

svc tiene las siguientes opciones:

-u levanta el servicio.
-d termina el servicio.
-t reinicia al servicio.

La forma de usar el comando *svc* es la siguiente:

Para levantar el servicio con la bitácora correspondiente de qmail, se utiliza el siguiente comando

```
#svc -u /service/qmail /service/qmail/log
```

El comando svstat servirá para comprobar el estado del servicio:

```
#svstat /service/qmail
```

Si es necesario levantar todos los servicios configurados anteriormente se utiliza el siguiente comando:

```
#svc -u /service/* /service/*/log
```

Debido a que todos los servicios quedan ligados al directorio */service,* mediante la instrucción *ln -s /var/qmail/service/tipo_de_servicio /servicio,* el comando *svc* con la opción *-u* inicia todos los procesos que se encuentran ligados a este directorio.

Si no se desea arrancar algún servicio en particular, sólo basta con **no** ligarlo al directorio para que éste no inicie sus procesos.

4.2.13 Funcionamiento de las Herramientas de LDAP y QMAIL.

Los capítulos anteriores describieron el funcionamiento y configuración de Ldap y Qmail. A continuación se presenta en la figura F4.2.1 la interacción entre ambas herramientas.

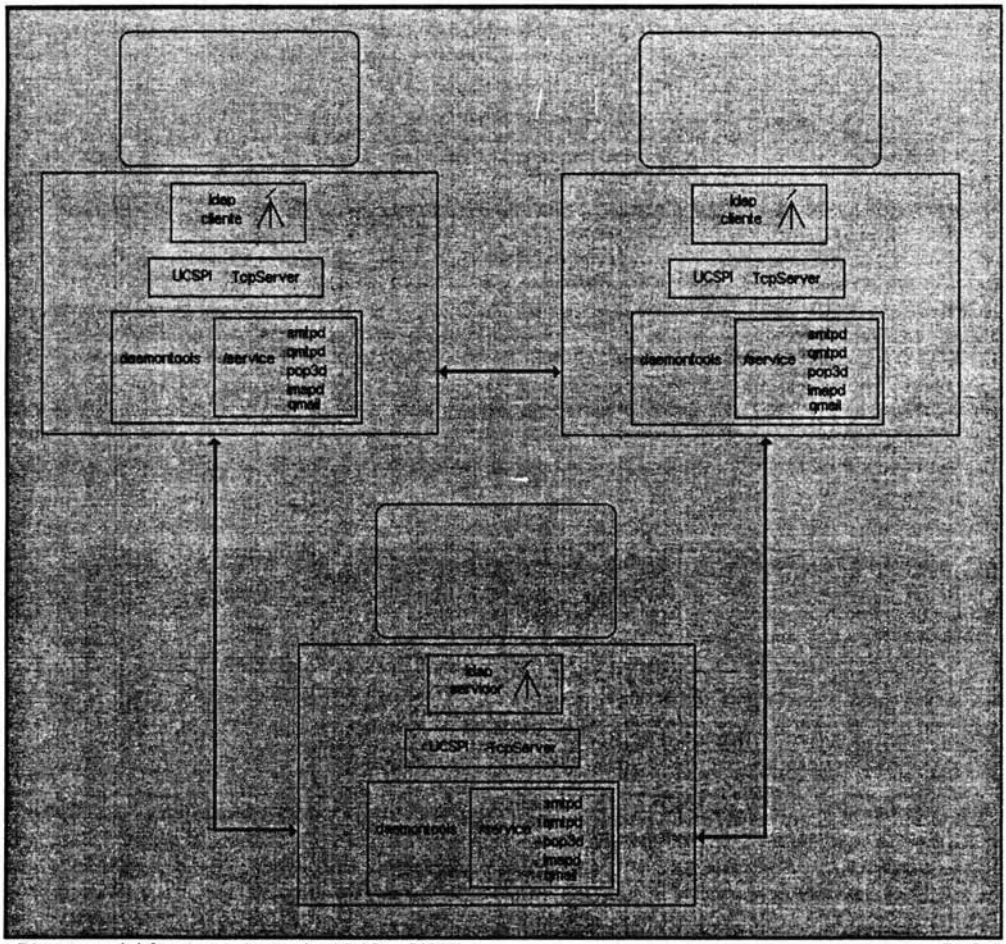


Diagrama del funcionamiento de LDAP y QMAIL F4.2.1

CONCLUSIONES

Como se trató en esta investigación, Qmail y LDAP son herramientas libres perfectamente escalables que permiten el manejo de servidores de correo con grandes capacidades de almacenamiento, sin problemas de seguridad.

Qmail se muestra como un rápido y poderoso reemplazo de Sendmail, su programación modular sin el uso constante de root, elimina excesivos ataques al servidor, así como el uso de relay para evitar correos spam.

La unificación de ambas tecnologías hoy está almacenando más de 20000 usuarios con un promedio de 15000 transacciones diarias (llamamos transacciones al envío y recepción de correo). Las máquinas que almacenan los correos de los usuarios tienen porcentajes mínimos de procesamiento, esto gracias a la distribución de usuarios que realiza qmail y al uso de clusters de ldap.

La limitante a la creación de nuevos usuarios, uno de los grandes problemas que hoy tienen muchos administradores, ha quedado en el olvido. Los excesivos tiempos de respaldo, que podían ser en algunas ocasiones más de 12 horas, hoy se han reducido en un 30%, una cifra considerable, lo que nos habla de la gran utilidad que tiene el formato maildir frente a mbox.

Otro de los puntos importantes de esta herramienta es el control del usuario sobre su correo. El administrador no tendrá la necesidad de modificar fechas en archivos del sistema por cuentas saturadas.

Qmail y LDAP presentan una debilidad frente al uso de otras herramientas, debilidad que se traduce en desconfianza por ser un software libre que carece de soporte. Esto ha causado que la mayoría de las empresas no opten por obtener este tipo de herramientas. Aunque la mayor parte del soporte se encuentra en Internet (donde se especifican las arquitecturas de ambas tecnologías y su funcionamiento), los administradores que laboran en empresas tienen miedo a enfrentarse a estos nuevos desarrollos prefiriendo comprar otros productos que harán algo similar pero con una diferencia notable en costos.

Grandes proyectos navegan alrededor de estas tecnologías, no obstante, que LDAP es medianamente antigüo, está alcanzando un auge en todo tipo de sistemas, donde la optimización y organización de los datos, que aportan los directorios, se ajusta perfectamente a los proyectos, siendo OpenLDAP una solución libre y segura para estos nuevos desarrollos.

Debido a lo anterior y todas sus demás características, sin olvidar el fácil manejo y configuración de las herramientas, Qmail y LDAP han completado todas las expectativas de aquellos administradores que tienen la tarea de mantener el servicio estable y que hasta ahora siguen sufriendo con otros agentes de transferencia de correo.

Esta tesis fue diseñada gracias a los administradores curiosos que creen en el software libre, y que han contribuido con sus documentaciones a que estas herramientas estén presentes en varias instituciones sin costo alguno.

APÉNDICE A

A.1 Tcp Wrappers

Cualquier sistema operativo debe de configurarse para que sea seguro antes de conectarlo a una red, especialmente cuando nos conectamos a Internet.

La mayoría de las veces se necesita que una misma computadora brinde varios servicios de red, si cada servicio tuviera un demonio ejecutándose esto causaría una sobrecarga en la computadora, por lo que un mismo demonio es el que se encarga de atender todos los servicios de red.

Entre los servicios más comunes tenemos ftp, smtp, telnet, http, etc. Estos servicios están definidos en /etc/inetd.conf y controlados por un demonio, que no es más que un programa que se ejecuta dentro del sistema operativo.

Tcp Wrappers es una herramienta de seguridad que trabaja a través de un demonio permitiendo el acceso a los servicios que se definan en los archivos hosts.allow o denegándolos en el archivo hosts.deny. Cuando un cliente quiere utilizar un servicio Tcp Wrappers ejecuta el servicio correspondiente y le da el control a la conexión, de esta manera sólo se tiene un demonio ejecutándose reduciendo la carga del sistema.

Tcpd anota todos los intentos de conexión que le llegan en /var/log/secure para que tenga la posibilidad de saber quién intenta conectarse a su máquina y si lo consigue. Si tcpd autoriza la conexión, ejecuta ipop3d que es el programa que realmente atiende la conexión, solicitando una autentificación para el acceso. Observe que ya se tiene tres niveles de seguridad: prestar un servicio, autorizar una conexión y validar un usuario.

También hay que asegurarse de que el programa ipop3d no tenga ninguna vulnerabilidad, es decir que esté actualizado. Existen numerosos medios para actualizar los programas y eliminar las posibles vulnerabilidades que se presentan.

Tcp wrappers no es una utilidad de cortafuegos pero funciona para algo parecido, además ofrece detección básica de impostores.

A continuación se describe el porque Tcp Wrappers no se considera como un verdadero cortafuegos.

- Se tiene que estar directamente conectado a Internet, es decir, se tiene que tener una dirección IP.
- Sólo controla la máquina en donde se encuentra instalado
- Un cortafuegos puede proteger todas las máquinas cualquiera que sea su arquitectura, Tcp Wrappers no funciona en Macintosh ni en MS DOS.

Tcp Wrappers es un software libre y lo puedes obtener en:

ftp://coast.cs.purdue.edu/pub/tools/unix/netutils/tcp_wrappers

A.2 Requerimientos e Instalación de Tcp Wrappers

Se debe tener instalado GCC de GNU, un compilador de lenguaje C, y seguir con los siguientes pasos:

1. Se descomprime y desempaqueta el software.

   $gzip –d < tcp_wrappers_vversion.tar.gz | tar xvf –

2. Se cambio al directorio de tcp-wrappers.

   $cd tcp_wrappers_versión

3. Se edita el archivo Makefile para ajustar parámetros de compilación según el sistema operativo que se maneje. (Para este apartado se utilizó Solaris)

   $vi Makefile

   Se quita el símbolo # para habilitar algunas líneas de código.

   ```
   #SysV.4 Solaris (versión) OSF AIX
   #REAL_DAEMON_DIR=/usr/sbin
   SysV.4 Solaris (versión) OSF AIX
   REAL_DAEMON_DIR=/usr/sbin

   #LIBS=-lsocket-Insl # SysV.4 Solaris(version)
   LIBS=-lsocket-Insl # SysV.4 Solaris(version)
   ```

   Se coloca el símbolo # y se quita otro # en la siguiente línea.

   ```
   BUGS = -DGETPEERNAME_BUG –DBROKEN_FGETS
   -DLIBC_CALLS_STRTOK
   #BUGS = -DGETPEERNAME_BUG –DBROKEN_FGETS
   -DSOLARIS_24_GETHOSTBYNAME_BUG

   #BUGS = -DGETPEERNAME_BUG –DBROKEN_FGETS
   -DLIBC_CALLS_STRTOK
   BUGS = -DGETPEERNAME_BUG –DBROKEN_FGETS
   -DSOLARIS_24_GETHOSTBYNAME_BUG
   ```

   Se debe cambiar LOG_MAIL por LOG_LOCAL0

   ```
   FACILITY=LOG_MAIL # LOG_MAIL is what most sendmail daemons use
   FACILITY=LOG_LOCAL0 # LOG_MAIL is what most sendmail daemons use
   ```

   Se elimina el simbolo #

   ```
   #AUTH =-DALWAYS_RFC931
   AUTH =-DALWAYS_RFC931
   ```

4. Se edita el archivo /etc/syslog.conf

$vi /etc/syslog.conf

Agrega en la última línea: *Local0.info /var/adm/tcpdlog*

NOTA:
El espacio entre *.info* y */var* es un tabulador y no un espacio.
La ruta y el nombre del archivo son establecidos de acuerdo al criterio del administrador. En este archivo se guardan los accesos al equipo a través de los servicios protegidos por tcp wrappers.

5. Se crea la ruta y el archivo especificado en /etc/syslog.conf

$cd /var/adm
$touch tcpdlog

6. Ejecute el comando make con un usuario normal (que no sea root)

#make sysv4 CC=gcc

7. Debe de continuar con root

#su – Password

8. Se debe cambiar al directorio de tcp wrappers y copiar los archivos indicados al directorio determinado por el administrador.

$cd ~user/tcp_wrappers_versión
$cp tcpd tcpdmatch tcpdchk try-from safe_finger /user/etc

9. Edite el archivo inetd.conf

$vi /etc/inetd.conf

Modificar las líneas que no concuerden con el archivo original de inet.conf

```
#ftp      stream  tcp   nowait  root  /usr/sbin/in.ftpd       in.ftpd
ftp       stream  tcp   nowait  root  /usr/etc/tcpd           in.ftpd
#telnet   stream  tcp   nowait  root  /usr/sbin/in.telnetd    in.telnetd
telnet    stream  tcp   nowait  root  /usr/etc/tcpd           in.telnetd
```

```
# Shell, login, exec, comsat and talk are BSD protocols

#shell    stream  tcp   nowait        root     /usr/sbin/in.rshd       in.rshd
#login    stream  tcp   nowait        root     /usr/sbin/in.rlogind    in.rlogin
#exec     stream  tcp   nowait        root     /usr/sbin/in.rexecd     in.rexecd
shell     stream  tcp   nowait        root     /usr/etc/tcpd           in.rshd
login     stream  tcp   nowait        root     /usr/etc/tcpd           in.rlogin
exec      stream  tcp   wait          root     /usr/etc/tcpd           in.rexecd
comsat    dgram   udp   wait          root     /usr/sbin/in.comsat     in.comsat
#talk     dgram   udp   wait          root     /usr/sbin/in.talkd      in.talkd
talk      dgram   udp   wait          root     /usr/etc/tcpd           in.talkd
finger    stream  tcp   nowait nobody /usr/etc/tcpd                    in.rexecd
```

10. Cree archivos de acceso

$vi /etc/hosts.allow

ALL:LOCAL .dominio.com, ip.que.entra.aqui

11. Cree archivos para negar el acceso

$vi /etc/hosts.allow

ALL:ALL

12. Levante el demonio de inetd.

$ps –fea | grep inetd

```
root     139      1        0    14:58:14?   0:00  /usr/sbin/inetd -s
root     227323   272881   0    12:45:14 pts/4   0:00   grep inetd
```

$kill –HUP 139

13. Levante el demonio de syslog.

$ps –fea | grep syslog

```
root     12459    54547    0    18:05:32 pts/4 0:00      grep syslogd
root     418      1        0    17:21:08 ?   0:03   /usr/sbin/syslogd
```

$kill –HUP 418

APÉNDICE B

B.1 Opciones para los comandos ldapadd, ldapdelete, ldapmodify, ldapsearch

-C Automáticamente (graba Y busca) las referencias

-n Muestra el resultado visualmente sin haber realizado ninguna modificación. Se utiliza en conjunción con -v en modo de ejecución.

-v Se ejecuta el comando en tiempo de diseño. Imprime algunos diagnósticos escritos en la salida estándar. (verbose)

-k Utiliza Kerberos para la autentificación. Se deberá compilar con Kerberos para que tenga efecto.

-K Utiliza Kerberos para la autentificación. Éste se utiliza cuando se conecta a un slapd donde X500dsa.hostname no ha sido registrado por el servidor de Kerberos.

-M[M] Habilita el manejador que controla el DSAIT -MM para los controles críticos.

-d Permite que LDAP encuentre los errores por niveles. Para ldapmodify debe compilarse la variable LDAP_DEBUG.

-D Utiliza el "Distinguished Name" o "dn" para enlazarse con el LDAP.

-W Se utiliza para especificar el passwd a través de la línea de comandos.

-w Se utiliza para la autentificación simple a través del password.

-H Especifica URI() para el servidor LDAP.

-h Especifica un host alterno en el que se encuentra un directorio LDAP.

-p Especifica un puerto TCP alterno donde se conectará LDAP.

-P Especifica la versión del protocolo que esté utilizando el LDAP.

-O Especifica la seguridad con SASL

-I Habilita el modo SASL interactivo. Com prompt

-Q Habilita el modo SASL no interactivo.

-U Con está opción se especifica el nombre del Usuario SASL correspondiente. La sintaxis del nombre del usuario depende del mecanismo que utilice SASL

-X Especifica la petición autorizada por SASL. Debe tener uno de los siguientes formatos:
dn:<distinguished name> ó <user name>

-x Especifica una simple autentificación por SASL.

-Y Especifica el mecanismo que utilizará SASL para la autentificación. En caso de que no sea especificada, el servidor tomará el mecanismo que mejor le convenga.

-Z[Z] Utiliza StartTLS (Transporte Seguro entre Capas). Este comando requiere que ciertas operaciones sean satisfactorias para que pueda funcionar.

-f Lee una serie de líneas desde un archivo. Se utiliza está opción para especificar operaciones como agregar, eliminar, modificar y buscar a través de un archivo.

-c En caso de reportarse algún tipo de error en la ejecución de algún comando éste continua con la operación asignada.

B.2 Schemas

B.2.1 Corba.schema

```
attributetype ( 1.3.6.1.4.1.42.2.27.4.1.14
        NAME 'corbaIor'
        DESC 'Stringified interoperable object reference of a CORBA object'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.42.2.27.4.1.15
        NAME 'corbaRepositoryId'
        DESC 'Repository ids of interfaces implemented by a CORBA object'
        EQUALITY caseExactMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )


objectclass ( 1.3.6.1.4.1.42.2.27.4.2.10
        NAME 'corbaContainer'
        DESC 'Container for a CORBA object'
        SUP top
        STRUCTURAL
        MUST cn )

objectclass ( 1.3.6.1.4.1.42.2.27.4.2.9
        NAME 'corbaObject'
        DESC 'CORBA object representation'
        SUP top
        ABSTRACT
        MAY ( corbaRepositoryId $ description ) )

objectclass ( 1.3.6.1.4.1.42.2.27.4.2.11
        NAME 'corbaObjectReference'
        DESC 'CORBA interoperable object reference'
        SUP corbaObject
        AUXILIARY
        MUST corbaIor )
```

B2.2 Core.schema

```
# Standard X.501(93) Operational Attribute Types from RFC2252

attributetype ( 2.5.18.1 NAME 'createTimestamp'
        EQUALITY generalizedTimeMatch
        ORDERING generalizedTimeOrderingMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
        SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

attributetype ( 2.5.18.2 NAME 'modifyTimestamp'
        EQUALITY generalizedTimeMatch
        ORDERING generalizedTimeOrderingMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
        SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

attributetype ( 2.5.18.3 NAME 'creatorsName'
        EQUALITY distinguishedNameMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
        SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

attributetype ( 2.5.18.4 NAME 'modifiersName'
        EQUALITY distinguishedNameMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
        SINGLE-VALUE NO-USER-MODIFICATION USAGE directoryOperation )

attributetype ( 2.5.18.10 NAME 'subschemaSubentry'
        EQUALITY distinguishedNameMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 NO-USER-MODIFICATION
        SINGLE-VALUE USAGE directoryOperation )

attributetype ( 2.5.21.1 NAME 'dITStructureRules'
        EQUALITY integerFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.17 USAGE directoryOperation )

attributetype ( 2.5.21.2 NAME 'dITContentRules'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.16 USAGE directoryOperation )

attributetype ( 2.5.21.4 NAME 'matchingRules'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.30 USAGE directoryOperation )

attributetype ( 2.5.21.5 NAME 'attributeTypes'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.3 USAGE directoryOperation )

attributetype ( 2.5.21.6 NAME 'objectClasses'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.37 USAGE directoryOperation )

attributetype ( 2.5.21.7 NAME 'nameForms'
        EQUALITY objectIdentifierFirstComponentMatch
```

```
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.35 USAGE directoryOperation )

attributetype ( 2.5.21.8 NAME 'matchingRuleUse'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.31 USAGE directoryOperation )

# From X.500(93)
attributetype ( 2.5.21.9 NAME 'structuralObjectClass'
        DESC 'X.500(93) structural object class'
        EQUALITY objectIdentifierMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
        NO-USER-MODIFICATION SINGLE-VALUE USAGE directoryOperation )

# LDAP Operational Attributes from RFC2252
attributetype ( 1.3.6.1.4.1.1466.101.120.5 NAME 'namingContexts'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.6 NAME 'altServer'
        SYNTAX 1.3.6.1.4 1.1466.115.121.1.26 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.7 NAME 'supportedExtension'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.13 NAME 'supportedControl'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.14 NAME 'supportedSASLMechanisms'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.15 NAME 'supportedLDAPVersion'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.120.16 NAME 'ldapSyntaxes'
        EQUALITY objectIdentifierFirstComponentMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.54 USAGE directoryOperation )

# Object Classes from RFC2252
objectclass ( 2.5.20.1 NAME 'subschema' AUXILIARY
        DESC 'RFC2252: controlling subschema'
        MAY ( dITStructureRules $ nameForms $ ditContentRules $
                objectClasses $ attributeTypes $ matchingRules $
                matchingRuleUse ) )

# Standard attribute types used for subtyping from RFC2256

attributetype ( 2.5.4.41 NAME 'name'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

attributetype ( 2.5.4.49 NAME 'distinguishedName'
        EQUALITY distinguishedNameMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
```

```
# Standard attribute types from RFC2256

attributetype ( 2.5.4.0 NAME 'objectClass'
        EQUALITY objectIdentifierMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )


attributetype ( 2.5.4.1 NAME 'aliasedObjectName'
        EQUALITY distinguishedNameMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 SINGLE-VALUE )

# obsolete
attributetype ( 2.5.4.2 NAME 'knowledgeInformation'
        EQUALITY caseIgnoreMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

attributetype ( 2.5.4.3 NAME ( 'cn' 'commonName' ) SUP name )

attributetype ( 2.5.4.4 NAME ( 'sn' 'surname' ) SUP name )

attributetype ( 2.5.4.5 NAME 'serialNumber' EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{64} )

# (2-letter code from ISO 3166)
attributetype ( 2.5.4.6 NAME ( 'c' 'countryName' ) SUP name SINGLE-VALUE )

attributetype ( 2.5.4.7 NAME ( 'l' 'localityName' ) SUP name )

attributetype ( 2.5.4.8 NAME ( 'st' 'stateOrProvinceName' ) SUP name )

attributetype ( 2.5.4.9 NAME ( 'street' 'streetAddress' )
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )

attributetype ( 2.5.4.10 NAME ( 'o' 'organizationName' ) SUP name )

attributetype ( 2.5.4.11 NAME ( 'ou' 'organizationalUnitName' ) SUP name )

attributetype ( 2.5.4.12 NAME 'title' SUP name )

attributetype ( 2.5.4.13 NAME 'description'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024} )

# Obsoleted by enhancedSearchGuide
attributetype ( 2.5.4.14 NAME 'searchGuide'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.25 )

attributetype ( 2.5.4.15 NAME 'businessCategory'
```

```
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )

attributetype ( 2.5.4.16 NAME 'postalAddress'
        EQUALITY caseIgnoreListMatch
        SUBSTR caseIgnoreListSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )

attributetype ( 2.5.4.17 NAME 'postalCode'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{40} )

attributetype ( 2.5.4.18 NAME 'postOfficeBox'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{40} )

attributetype ( 2.5.4.19 NAME 'physicalDeliveryOfficeName'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{128} )

attributetype ( 2.5.4.20 NAME 'telephoneNumber'
        EQUALITY telephoneNumberMatch
        SUBSTR telephoneNumberSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.50{32} )

attributetype ( 2.5.4.21 NAME 'telexNumber'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.52 )

attributetype ( 2.5.4.22 NAME 'teletexTerminalIdentifier'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.51 )

attributetype ( 2.5.4.23 NAME ( 'facsimileTelephoneNumber' 'fax' )
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.22 )

attributetype ( 2.5.4.24 NAME 'x121Address'
        EQUALITY numericStringMatch
        SUBSTR numericStringSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.36{15} )

attributetype ( 2.5.4.25 NAME 'internationaliSDNNumber'
        EQUALITY numericStringMatch
        SUBSTR numericStringSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.36{16} )

attributetype ( 2.5.4.26 NAME 'registeredAddress' SUP postalAddress
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.41 )

attributetype ( 2.5.4.27 NAME 'destinationIndicator'
        EQUALITY caseIgnoreMatch
```

```
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{128} )

attributetype ( 2.5.4.28 NAME 'preferredDeliveryMethod'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.14
        SINGLE-VALUE )

attributetype ( 2.5.4.29 NAME 'presentationAddress'
        EQUALITY presentationAddressMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.43
        SINGLE-VALUE )

attributetype ( 2.5.4.30 NAME 'supportedApplicationContext'
        EQUALITY objectIdentifierMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )

attributetype ( 2.5.4.31 NAME 'member' SUP distinguishedName )

attributetype ( 2.5 4.32 NAME 'owner' SUP distinguishedName )

attributetype ( 2.5.4.33 NAME 'roleOccupant' SUP distinguishedName )

attributetype ( 2.5.4.34 NAME 'seeAlso' SUP distinguishedName )

attributetype ( 2.5.4.35 NAME 'userPassword'
        EQUALITY octetStringMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.40{128} )

# Must be transferred using ;binary
attributetype ( 2.5.4.36 NAME 'userCertificate'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )

# Must be transferred using ;binary
attributetype ( 2.5.4.37 NAME 'cACertificate'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.8 )

# Must be transferred using ;binary
attributetype ( 2.5.4.38 NAME 'authorityRevocationList'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

# Must be transferred using ;binary
attributetype ( 2.5.4.39 NAME 'certificateRevocationList'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

# Must be stored and requested in the binary form
attributetype ( 2.5.4.40 NAME 'crossCertificatePair'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.10 )

# 2.5.4.41 is defined above as it's used for subtyping
#attributetype ( 2.5.4.41 NAME 'name'
#       EQUALITY caseIgnoreMatch
#       SUBSTR caseIgnoreSubstringsMatch
#       SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )
```

```
attributetype ( 2.5.4.42 NAME ( 'givenName' 'gn' ) SUP name )

attributetype ( 2.5.4.43 NAME 'initials' SUP name
        DESC 'The initials attribute type contains the initials of some
                or all of an individuals names, but not the surname(s).' )

attributetype ( 2.5.4.44 NAME 'generationQualifier'
        DESC 'e.g. Jr or II.'
        SUP name )

attributetype ( 2.5.4.45 NAME 'x500UniqueIdentifier'
        EQUALITY bitStringMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.6 )

attributetype ( 2.5.4.46 NAME 'dnQualifier'
        EQUALITY caseIgnoreMatch
        ORDERING caseIgnoreOrderingMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )

attributetype ( 2.5.4.47 NAME 'enhancedSearchGuide'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.21 )

attributetype ( 2.5.4.48 NAME 'protocolInformation'
        EQUALITY protocolInformationMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.42 )

# 2.5.4.49 is defined above as it's used for subtyping
#attributetype ( 2.5.4.49 NAME 'distinguishedName'
#       EQUALITY distinguishedNameMatch
#       SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )

attributetype ( 2.5.4.50 NAME 'uniqueMember'
        EQUALITY uniqueMemberMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.34 )

attributetype ( 2.5.4.51 NAME 'houseIdentifier'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )

# Must be transferred using ;binary
attributetype ( 2.5.4.52 NAME 'supportedAlgorithms'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.49 )

# Must be transferred using ;binary
attributetype ( 2.5.4.53 NAME 'deltaRevocationList'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.9 )

attributetype ( 2.5.4.54 NAME 'dmdName' SUP name )

# Standard object classes from RFC2256
```

```
objectclass ( 2.5.6.0 NAME 'top' ABSTRACT
        MUST objectClass )

objectclass ( 2.5.6.1 NAME 'alias' SUP top STRUCTURAL
        MUST aliasedObjectName )

objectclass ( 2.5.6.2 NAME 'country' SUP top STRUCTURAL
        MUST c
        MAY ( searchGuide $ description ) )

objectclass ( 2.5.6.3 NAME 'locality' SUP top STRUCTURAL
        MAY ( street $ seeAlso $ searchGuide $ st $ l $ description ) )

objectclass ( 2.5.6.4 NAME 'organization' SUP top STRUCTURAL
        MUST o
        MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
                x121Address $ registeredAddress $ destinationIndicator $
                preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $
                facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
                postalAddress $ physicalDeliveryOfficeName $ st $ l $ description ) )

objectclass ( 2.5.6.5 NAME 'organizationalUnit' SUP top STRUCTURAL
        MUST ou
        MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
                x121Address $ registeredAddress $ destinationIndicator $
                preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $
                facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
                postalAddress $ physicalDeliveryOfficeName $ st $ l $ description ) )

objectclass ( 2.5.6.6 NAME 'person' SUP top STRUCTURAL
        MUST ( sn $ cn )
        MAY ( userPassword $ telephoneNumber $ seeAlso $ description ) )

objectclass ( 2.5.6.7 NAME 'organizationalPerson' SUP person STRUCTURAL
        MAY ( title $ x121Address $ registeredAddress $ destinationIndicator $
                preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $
                facsimileTelephoneNumber $ street $ postOfficeBox $ postalCode $
                postalAddress $ physicalDeliveryOfficeName $ ou $ st $ l ) )

objectclass ( 2.5.6.8 NAME 'organizationalRole' SUP top STRUCTURAL
        MUST cn
        MAY ( x121Address $ registeredAddress $ destinationIndicator $
                preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $ facsimileTelephoneNumber $
                seeAlso $ roleOccupant $ preferredDeliveryMethod $ street $
                postOfficeBox $ postalCode $ postalAddress $
                physicalDeliveryOfficeName $ ou $ st $ l $ description ) )

objectclass ( 2.5.6.9 NAME 'groupOfNames' SUP top STRUCTURAL
```

```
        MUST ( member $ cn )
        MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )

objectclass ( 2.5.6.10 NAME 'residentialPerson' SUP person STRUCTURAL
        MUST l
        MAY ( businessCategory $ x121Address $ registeredAddress $
                destinationIndicator $ preferredDeliveryMethod $ telexNumber $
                teletexTerminalIdentifier $ telephoneNumber $ internationaliSDNNumber $
                facsimileTelephoneNumber $ preferredDeliveryMethod $ street $
                postOfficeBox $ postalCode $ postalAddress $
                physicalDeliveryOfficeName $ st $ l ) )

objectclass ( 2.5.6.11 NAME 'applicationProcess' SUP top STRUCTURAL
        MUST cn
        MAY ( seeAlso $ ou $ l $ description ) )

objectclass ( 2.5.6.12 NAME 'applicationEntity' SUP top STRUCTURAL
        MUST ( presentationAddress $ cn )
        MAY ( supportedApplicationContext $ seeAlso $ ou $ o $ l $
        description ) )

objectclass ( 2.5.6.13 NAME 'dSA' SUP applicationEntity STRUCTURAL
        MAY knowledgeInformation )

objectclass ( 2.5.6.14 NAME 'device' SUP top STRUCTURAL
        MUST cn
        MAY ( serialNumber $ seeAlso $ owner $ ou $ o $ l $ description ) )

objectclass ( 2.5.6.15 NAME 'strongAuthenticationUser' SUP top AUXILIARY
        MUST userCertificate )

objectclass ( 2.5.6.16 NAME 'certificationAuthority' SUP top AUXILIARY
        MUST ( authorityRevocationList $ certificateRevocationList $
                cACertificate ) MAY crossCertificatePair )

objectclass ( 2.5.6.17 NAME 'groupOfUniqueNames' SUP top STRUCTURAL
        MUST ( uniqueMember $ cn )
        MAY ( businessCategory $ seeAlso $ owner $ ou $ o $ description ) )

objectclass ( 2.5.6.18 NAME 'userSecurityInformation' SUP top AUXILIARY
        MAY ( supportedAlgorithms ) )

objectclass ( 2.5.6.16.2 NAME 'certificationAuthority-V2' SUP
        certificationAuthority
        AUXILIARY MAY ( deltaRevocationList ) )

objectclass ( 2.5.6.19 NAME 'cRLDistributionPoint' SUP top STRUCTURAL
        MUST ( cn )
        MAY ( certificateRevocationList $ authorityRevocationList $
                deltaRevocationList ) )

objectclass ( 2.5.6.20 NAME 'dmd' SUP top STRUCTURAL
        MUST ( dmdName )
```

```
        MAY ( userPassword $ searchGuide $ seeAlso $ businessCategory $
                x121Address $ registeredAddress $ destinationIndicator $
                preferredDeliveryMethod $ telexNumber $ teletexTerminalIdentifier $
                telephoneNumber $ internationaliSDNNumber $ facsimileTelephoneNumber $
                street $ postOfficeBox $ postalCode $ postalAddress $
                physicalDeliveryOfficeName $ st $ l $ description ) )

objectclass ( 1.3.6.1.4.1.1466.101.120.111 NAME 'extensibleObject'
        DESC 'RFC2252: extensible object'
        SUP top AUXILIARY )

#
# Standard Track URI label schema from RFC2079
#
attributetype ( 1.3.6.1.4.1.250.1.57 NAME 'labeledURI'
        DESC 'RFC2079: Uniform Resource Identifier with optional label'
        EQUALITY caseExactMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15 )

objectclass ( 1.3.6.1.4.1.250.3.15 NAME 'labeledURIObject'
        DESC 'RFC2079: object that contains the URI attribute type'
        MAY ( labeledURI )
        SUP top AUXILIARY )

#
# Standard Track Dynamic Directory Services from RFC2589
#
objectclass ( 1.3.6.1.4.1.1466.101.119.2 NAME 'dynamicObject'
        DESC 'RFC2589: Dynamic Object'
        SUP top AUXILIARY )

attributetype ( 1.3.6.1.4.1.1466.101.119.3 NAME 'entryTtl'
        DESC 'RFC2589: entry time-to-live'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE
        NO-USER-MODIFICATION USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.1466.101.119.4 NAME 'dynamicSubtrees'
        DESC 'RFC2589: dynamic subtrees'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 NO-USER-MODIFICATION
        USAGE dSAOperation )

#
# Derived from RFC1274, but with new "short names"
#
attributetype ( 0.9.2342.19200300.100.1.1
        NAME ( 'uid' 'userid' )
        DESC 'RFC1274: user identifier'
        EQUALITY caseIgnoreMatch
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{256} )

attributetype ( 0.9.2342.19200300.100.1.3
        NAME ( 'mail' 'rfc822Mailbox' )
```

```
        DESC 'RFC1274: RFC822 Mailbox'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

objectclass ( 0.9.2342.19200300.100.4.19 NAME 'simpleSecurityObject'
        DESC 'RFC1274: simple security object'
        SUP top AUXILIARY
        MUST userPassword )

# RFC1274 + RFC2247
attributetype ( 0.9.2342.19200300.100.1.25
        NAME ( 'dc' 'domainComponent' )
        DESC 'RFC1274/2247: domain component'
        EQUALITY caseIgnoreIA5Match
        SUBSTR caseIgnoreIA5SubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

# RFC2247
objectclass ( 1.3.6.1.4.1.1466.344 NAME 'dcObject'
        DESC 'RFC2247: domain component object'
        SUP top AUXILIARY MUST dc )


# From RFC2377
objectclass ( 1.3.6.1.1.3.1 NAME 'uidObject'
        DESC 'RFC2377: uid object'
        SUP top AUXILIARY MUST uid )

#
# From draft-zeilenga-ldap-nameref-00.txt
#       used to represent referrals in the directory
#
attributetype ( 2.16.840.1.113730.3.1.34 NAME 'ref'
        DESC 'Named referral'
        EQUALITY caseExactMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
        USAGE distributedOperation )

objectclass ( 2.16.840.1.113730.3.2.6 NAME 'referral'
        DESC 'Named referral object'
        SUP top STRUCTURAL MUST ref )

#
# LDAPsubEntry
#       likely to change!
objectclass ( 2.16.840.1.113719.2.142.6.1.1 NAME 'LDAPsubEntry'
        DESC 'LDAP Subentry'
        SUP top STRUCTURAL MAY cn )

#
# OpenLDAProotDSE
#       likely to change!
```

```
objectclass ( 1.3.6.1.4.1.4203.1.4.1
        NAME ( 'OpenLDAProotDSE' 'LDAProotDSE' )
        DESC 'OpenLDAP Root DSE object'
        SUP top STRUCTURAL MAY cn )

#
# From Cosine Pilot
#
attributetype ( 0.9.2342.19200300.100.1.37 NAME 'associatedDomain'
        EQUALITY caseIgnoreIA5Match
        SUBSTR caseIgnoreIA5SubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

#
# From U-Mich
#
attributetype ( 1.3.6.1.4.1.250.1.32
        NAME ( 'krbName' 'kerberosName' )
        DESC 'Kerberos Name'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
        SINGLE-VALUE )

#
# draft-zeilenga-ldap-features-xx.txt (supportedFeatures)
#
attributetype ( 1.3.6.1.4.1.4203.1.3.5
    NAME 'supportedFeatures'
    DESC 'features supported by the server'
    EQUALITY objectIdentifierMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
    USAGE dSAOperation )
#
# OpenLDAP specific schema items
#
attributetype ( 1.3.6.1.4.1.4203.1.3.1
        NAME 'entry'
        DESC 'OpenLDAP ACL entry pseudo-attribute'
        SYNTAX 1.3.6.1.4.1.4203.1.1.1
        SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.4203.1.3.2
        NAME 'children'
        DESC 'OpenLDAP ACL children pseudo-attribute'
        SYNTAX 1.3.6.1.4.1.4203.1.1.1
        SINGLE-VALUE NO-USER-MODIFICATION USAGE dSAOperation )

attributetype ( 1.3.6.1.4.1.4203.666.1.5
        NAME 'OpenLDAPaci'
        DESC 'OpenLDAP access control information'
        EQUALITY OpenLDAPaciMatch
        SYNTAX 1.3.6.1.4.1.4203.666.2.1
        USAGE directoryOperation )
```

B.2.3 qmail.schema

```
# Attribute Type Definitions

attributetype ( 1.3.6.1.4.1.7914.1.2.1.1 NAME 'qmailUID'
        DESC 'UID of the user on the mailsystem'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.36
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.2 NAME 'qmailGID'
        DESC 'GID of the user on the mailsystem'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.36
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.3 NAME 'mailMessageStore'
        DESC 'Path to the maildir/mbox on the mail system'
        EQUALITY caseExactIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.4 NAME 'mailAlternateAddress'
        SUBSTR caseIgnoreSubstringsMatch
        DESC 'Secondary (alias) mailaddresses for the same user'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.5 NAME 'mailQuota'
        DESC 'The amount of space the user can use until all further messages get bounced.'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.6 NAME 'mailHost'
        DESC 'On which qmail server the messagestore of this user is located.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.7 NAME 'mailForwardingAddress'
        DESC 'Address(es) to forward all incoming messages to.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.8 NAME 'deliveryProgramPath'
        DESC 'Program to execute for all incoming mails.'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.9 NAME 'qmailDotMode'
        DESC 'Interpretation of .qmail files: both, dotonly, ldaponly, ldapwithprog, none'
        EQUALITY caseIgnoreIA5Match
```

```
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.10 NAME 'deliveryMode'
        DESC 'multi field entries of: normal, forwardonly, nombox, localdelivery, reply, echo'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44 )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.11 NAME 'mailReplyText'
        DESC 'A reply text for every incoming message'
        SUBSTR caseIgnoreSubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44{4096}
        SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7914.1.2.1.12 NAME 'accountStatus'
        DESC 'The status of a user account: active, nopop, disabled'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.44
        SINGLE-VALUE )

# Object Class Definitions

objectclass ( 1.3.6.1.4.1.7914.1.2.2.1 NAME 'qmailUser'
        DESC 'QMail-LDAP User' SUP top AUXILIARY
        MUST ( mail $ uid )
        MAY ( mailMessageStore $ homeDirectory $ userPassword $
            mailAlternateAddress $ qmailUID $ qmailGID $ mailQuota $
            mailHost $ mailForwardingAddress $ deliveryProgramPath $
            qmailDotMode $ deliveryMode $ mailReplyText $
            accountStatus ) )
```

B.2.4 qmailControl.schema

```
# Attributes that are qmail specific

attributetype ( 1.3.6.1.4.1.7006.1.2.1.13 NAME 'badMailFrom'
        DESC 'Unacceptable envelope sender addresses.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.14 NAME 'bounceFrom'
        DESC 'Bounce username'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.15 NAME 'bounceHost'
        DESC 'Bounce host'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.16 NAME 'concurrencyLocal'
        DESC 'Maximum number of simultaneous local delivery attempts.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.17 NAME 'concurrencyRemote'
        DESC 'Maximum number of simultaneous remote delivery attempts.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.18 NAME 'defaultDomain'
        DESC 'Default domain name.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.19 NAME 'defaultHost'
        DESC 'Default host name.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.20 NAME 'dataBytes'
        DESC 'Maximum number of bytes allowed in a message, or 0 for no limit.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.21 NAME 'doubleBounceHost'
        DESC 'Double-bounce host.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.22 NAME 'doubleBounceTo'
        DESC 'User to receive double-bounces.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

```
attributetype ( 1.3.6.1.4.1.7006.1.2.1.23 NAME 'envNoAtHost'
        DESC 'Presumed domain name for addresses without @ signs.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.24 NAME 'heloHost'
        DESC 'Host name used to say hello to the remote SMTP server.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE  )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.25 NAME 'idHost'
        DESC 'Host name for Message-IDs.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.26 NAME 'localIpHost'
        DESC 'Replacement host name for local IP addresses.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.27 NAME 'locals'
        DESC 'List of domain names that the current host receives mail for.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.28 NAME 'moreRcptHosts'
        DESC 'Extra allowed RCPT domains.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.29 NAME 'percentHack'
        DESC 'List of domain names where the percent hack is applied.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.30 NAME 'plusDomain'
        DESC 'Plus domain name.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.31 NAME 'qmqpServers'
        DESC 'IP addresses of QMQP servers.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{128} )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.32 NAME 'queueLifetime'
        DESC 'Number of seconds a message can stay in the queue.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.33 NAME 'rcptHosts'
        DESC 'Allowed RCPT domains.'
```

```
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.34 NAME 'smtpGreeting'
        DESC 'SMTP greeting message.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.35 NAME 'smtpRoutes'
        DESC 'Artificial SMTP routes.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.36 NAME 'timeoutConnect'
        DESC 'Number of seconds qmail-remote will wait for the remote SMTP server to accept
a connection.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.37 NAME 'timeoutRemote'
        DESC 'Number of seconds qmail-remote will wait for each response from the remote
SMTP server.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.38 NAME 'timeoutSmtpd'
        DESC 'Timeout for each new buffer of data from the remote SMTP client.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.39 NAME 'virtualDomains'
        DESC 'List of virtual users or domains.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

# Attributes from qmail-ldap

attributetype ( 1.3.6.1.4.1.7006.1.2.1.40 NAME 'ldapBaseDN'
        DESC 'The base DN from where the search in the LDAP tree begins.'
        EQUALITY caseIgnoreIA5Match
        SUP distinguishedName SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.41 NAME 'ldapLogin'
        DESC 'Username for the LDAP server connection.'
        EQUALITY caseIgnoreIA5Match
        SUP distinguishedName SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.42 NAME 'ldapPassword'
        DESC 'Password for the LDAP server connection.'
        EQUALITY caseExactMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.43 NAME 'ldapLocalDelivery'
```

```
        DESC 'If on do a lookup on the local passwd file.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.7 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.44 NAME 'ldapRebind'
        DESC 'Use the possibility of rebinding to the ldap-server to compare pop3 and imap
passwords.'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.45 NAME 'ldapCluster'
        DESC 'Turn clustering on and off.'
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.46 NAME 'ldapDefaultQuota'
        DESC 'The default amount of disk space the user can use.'
        EQUALITY caseExactIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.47 NAME 'ldapDefaultDotMode'
        DESC 'The default interpretation of .qmail files'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.48 NAME 'ldapMessageStore'
        DESC 'The default prefix for pathes in mailMessageStore.'
        EQUALITY caseExactIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.49 NAME 'ldapUid'
        DESC 'The default UID used in virtual users environments.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.50 NAME 'ldapGid'
        DESC 'The default GID used in virtual users environments.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.51 NAME 'customBounceText'
        DESC 'Additional custom text in bounce messages.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.52 NAME 'quotaWarning'
        DESC 'Custom text in quota warning message.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{1024} SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.53 NAME 'tarpitCount'
        DESC 'Number of RCPT TOs you accept before you start tarpitting.'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
```

```
attributetype ( 1.3.6.1.4.1.7006.1.2.1.54 NAME 'tarpitDelay'
        DESC 'Number of seconds of delay to introduce after each subsequent RCPT TO'
        EQUALITY integerMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.55 NAME 'badRcptTo'
        DESC 'List of recipient addresses that should be rejected.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.56 NAME 'dirMaker'
        DESC 'Absolute path to your program/script that creates missing homedirs.'
        EQUALITY caseExactIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

attributetype ( 1.3.6.1.4.1.7006.1.2.1.57 NAME 'ldapServer'
        DESC 'LDAP Server address.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )

# Attributes from TLS

attributetype ( 1.3.6.1.4.1.7006.1.2.1.58 NAME 'tlsClients'
        DESC 'This email-address is logged in the headers.'
        EQUALITY caseIgnoreIA5Match
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

B.2.5 openldap.schema

```
# OpenLDAP Project's directory schema items
#
# depends upon:
#       core.schema
#       cosine.schema
#       inetorgperson.schema
#
# These are provided for informational purposes only.


objectClass ( 1.3.6.1.4.1.4203.1.4.3 NAME 'OpenLDAPorg'
        DESC 'OpenLDAP Organizational Object'
        SUP organization
        MAY ( buildingName $ displayName $ labeledURI ) )

objectClass ( 1.3.6.1.4.1.4203.1.4.4 NAME 'OpenLDAPou'
        DESC 'OpenLDAP Organizational Unit Object'
        SUP organizationalUnit
        MAY ( buildingName $ displayName $ labeledURI $ o ) )

objectClass ( 1.3.6.1.4.1.4203.1.4.5 NAME 'OpenLDAPperson'
        DESC 'OpenLDAP Person'
        SUP ( pilotPerson $ inetOrgPerson )
        MUST ( uid $ cn )
        MAY ( givenName $ labeledURI $ o ) )

objectClass ( 1.3.6.1.4.1.4203.1.4.6 NAME 'OpenLDAPdisplayableObject'
        DESC 'OpenLDAP Displayable Object'
        MAY displayName AUXILIARY )
```

BIBLIOGRAFÍA Y VÍNCULOS

QMAIL, capítulo 3, David Gunter, Steve Burnett, Lola Gunter,. Guia de Integración de Windows NT y UNIX, Editorial McGrawHill, 1998.

Introducción, página 6, Cricket Liu, Jerry Peek, Russ jones, Bryan Buus, Adrián Nye, Administración de Servicios de Información en Internet, Editorial Mc Graw Hill, 1997.

Vinculos (URLs):

- Arquitectura de Qmail, página 43, http://www.nrg4u.com/
- Introducción a qmail, página 42, http://www.lifewithqmail.org/lwq.html#whatitis :
- Creación de los guiones de arranque con qmail-conf, página 68, http://www.din.or.jp/~ushijima/qmail-conf.html
- ¿Qué es LDAP?, página 16, http://www.linuxfocus.org/Castellano/July2000/article159.shtml
- OpenLdap y su Instalación, página 19, www.openldap.org
- ¿Qué es LDAP?, página 16, www.kingsmountain.com/ldapRoadmap.shtml
- ¿Qué es LDAP?, página 16, www.gracion.com/server/whatldap.html
- ¿Qué es LDAP?, página 16, http://www.pcwebopaedia.com/TERM/L/LDAP.htm
- Inserción de datos en LDAP, página 31, http://www.georgetown.edu/giia/internet2/ldap-recipe/
- OpenLdap y su Instalación, página 19, http://quark.humbug.org.au/publications/ldap_tut.html
- ¿Qué es LDAP?, página 16, http://www-sop.inria.fr/semir/personnel/Laurent.Mirtain/ldap-livre.html
- LDAP, capítulo 2, http://www.tldp.org/HOWTO/LDAP-HOWTO/
- LDAP, capítulo 2, http://networking.webopedia.com/TERM/L/LDAP.html
- LDAP,Terminología, http://java.sun.com/products/jndi/tutorial/getStarted/concepts/glossary.htm#REFERRAL
- QMAIL, capítulo 3, http://www.flounder.net/qmail/qmail-howto.html
- QMAIL, capítulo 3, http://members.elysium.pl/brush/qmail-smtpd-auth/
- INSTALACIÓN Y FUNCIONAMIENTO DE QMAIL CON LDAP, capítulo 4, http://www.din.or.jp/~ushijima/qmail-conf.html
- INSTALACIÓN Y FUNCIONAMIENTO DE QMAIL CON LDAP, capítulo 4, http://untroubled.org/qmail+patches/
- INSTALACIÓN Y FUNCIONAMIENTO DE QMAIL CON LDAP, capítulo 4,
- http://freebsd.ntu.edu.tw/djb/qmail/var-qmail.html
- Proceso de compilación, página 64, http://www.jms1.net/qmail/
- INSTALACIÓN Y FUNCIONAMIENTO DE QMAIL CON LDAP, capítulo 4,
- http://tertius.org/software/authpop/

Software

http://www.mrzesty.net/index.html?frame=http://braindump.MrZesty.net/qmail-ldap.html