

Tutorial de Sendmail

V0.30 - Diego Bravo Estrada

Tabla de contenidos

Nota Introductoria	3
Una Configuración Típica	3
Conceptos	6
Empezando	8
Configuración del sistema.....	11
Sistema de configuración M4.....	13
El Procesamiento de los mensajes	15
Administración de la cola de Sendmail.....	17
Dominios virtuales	20
Enmascaramiento	21
Sitios con más de un servidor.....	22
Performance/Tuning.....	26
Reglas y Rulesets	30
A. Ejemplo: Ambiente de alta seguridad	35
B. Ejemplo de configuración de MUA	37
C. Referencias	40

Este tutorial pretende dar una visión general del MTA Sendmail. Los ejemplos asumen un sistema Linux RedHat 7.X, 8.0 y 9.0, pero pueden extenderse a cualquier otro sistema operativo soportado.

El lector realmente interesado en conocer este MTA debe consultar la referencia [1], a partir de la cual se ha hecho este tutorial.

Nota Introductoria

Si tuviera que buscar un adjetivo para calificar a Sendmail, pensaría en "excesivo". Excesivo puesto que este programa intenta -y puede- satisfacer las necesidades de una audiencia extremadamente amplia... incluso, de una audiencia que hace años ha desaparecido.

En general, cuando un programa es "más y más flexible", los usuarios deben pagar el precio de "más y más complejidad" para asimilar toda aquella flexibilidad. Sendmail permite configurar aspectos que normalmente yacen ocultos en el código compilado de otros programas similares... aspectos que en la práctica diaria ya casi nadie usa.

Es por eso que usar Sendmail suele ser una experiencia desconcertante... desde el inicio y hasta el final. Y en ese sentido tengo que admitir que este pequeño texto también puede serlo, pese a que he procurado que no ocurra así.

Sendmail es calificado de "inseguro", y con justa razón. Tiene una larga historia de "vulnerabilidades" que han conminado a algunos administradores a optar por soluciones supuestamente más seguras como `postfix` y `qmail`. En favor de Sendmail sólo tengo que indicar que sus creadores han venido haciendo grandes esfuerzos para que esto cambie, y ciertamente las últimas versiones han consistido esencialmente en mejoras en esa dirección.

Un último adjetivo para Sendmail podría ser "legendario", en el sentido que es una de las utilidades más antiguas de los sistemas Unix, se proporciona en prácticamente todas sus variantes (y también en Linux), es el servidor de email más utilizado a nivel mundial, y sorprendentemente es a la vez una de las utilidades menos "dominadas" por los administradores debido a lo antes indicado - así como a una muy deficiente documentación!

Cómo se ha confeccionado este texto

En la medida que tengo costumbre de usar `vi`, pero no tengo costumbre de escribir y recordar los extensos y variados tags de Docbook, he usado mi script "QDK" disponible en Sourceforge¹ a fin de generar el SGML respectivo. Recomiendo su uso al lector que está acostumbrado a escribir Docbook directamente.

Cómo contactarme

Cualquier sugerencia para mejorar este texto, o el señalamiento de algún error, agradecería me sea indicado escribiendo a: `diego@most-wanted.com`.

Lima, Noviembre de 2003

Una Configuración Típica

Empezaremos con una descripción general de los pasos que se requieren en una configuración típica de Sendmail.

Los lectores que no disponen de absolutamente ninguna experiencia con Sendmail deberían pasar previamente por la sección denominada "Conceptos".

Escenario

La gran mayoría de sitios pequeños en Internet puede usar la configuración que proporciona RedHat en forma automática. Es el caso típico de una organización que dispone de un único servidor de correo electrónico con conexión directa a Internet, y que posee un dominio tal como "laorganizacion.org".

Asumiremos que el servidor de correo designado se llama "correo.laorganizacion.org" y no consideraremos detalles de seguridad como firewalls y redes DMZ.

Asumiremos también que nuestros clientes son las estaciones de trabajo que se conectan con algún cliente de correo estándar como "Outlook Express", "Mozilla", etc.

Configurar el DNS

Asumiremos que las direcciones de correo de nuestros usuarios son de la forma "usuario@laorganizacion.org". En ese caso, en el archivo de configuración de la zona "laorganizacion.org" deberá inscribirse el siguiente registro MX:

```
@ 1D IN MX 0 correo
```

Esto envía los mensajes con ese formato a nuestro servidor "correo.laorganizacion.org". (He asumido el uso de BIND.)

Configurar las opciones del puerto SMTP

En RedHat 7.x, 8, 9 (y quizá futuras versiones), Sendmail viene por defecto configurado para aceptar sólo conexiones locales; es decir, no recibirá ningún mensaje que llegue desde el exterior.

Esto no sirve de mucho en ambientes típicos de red, por lo que editaremos el archivo /etc/mail/sendmail.mc y modificaremos la siguiente línea:

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')
```

por:

```
DAEMON_OPTIONS(`Port=smtp, Name=MTA')
```

Y regeneraremos la configuración:

```
# cd /etc/mail
# m4 sendmail.mc > sendmail.cf
```

Análogamente, en RedHat 7.x se modificará el archivo redhat.mc.

Configurar el archivo /etc/hosts

Asegurémonos de que existan las siguientes DOS líneas, y que tengan un contenido como a este:

```
127.0.0.1 localhost
1.2.3.4 correo.organizacion.org correo
```

Obviamente, se deberá reemplazar "1.2.3.4" por la dirección IP asociada a la interfaz de la red local.

Configurar el archivo local-host-names

Como nuestras direcciones son "usuario@organizacion.org", nuestro servidor debe asumir como SUYOS todos los mensajes dirigidos a "@organizacion.org". Esto se consigue escribiendo "organizacion.org" en el archivo `/etc/mail/local-host-names`:

```
organizacion.org
```

En muchas versiones antiguas de Sendmail, el archivo equivalente se llama `/etc/sendmail.cw`.

Permitir el relay a nuestros clientes

Asumiremos que las estaciones de nuestra organizacion están contenidas en la subnet "1.2.3.0". En ese caso, añadiremos la siguiente línea al archivo `/etc/mail/access`:

```
1.2.3 RELAY
```

Luego generaremos la versión indexada:

```
# cd /etc/mail
# make
```

Nótese que en muchos sistemas distintos a RedHat habrá que usar (en vez de "make") el comando "makemap" con las opciones correspondientes:

```
bash# cd /etc/mail
bash# makemap hash access < access
```

Esta funcionalidad viene activada normalmente en la configuración proporcionada por RedHat. En otros sistemas probablemente esto se debe configurar explícitamente como se indica más abajo.

Configurar servicios POP / IMAP

Estos dos servicios provienen del paquete "imap*.rpm". Se deberán activar con el comando `ntsysv` o `chkconfig` (basta con uno de ellos.) Por ejemplo, para el último caso:

```
bash# chkconfig --level 345 ipop2 on
bash# chkconfig --level 345 ipop3 on
bash# chkconfig --level 345 imap on
```

Luego se deberá recargar la configuración de `xinetd`:

```
# service xinetd reload
```

Ahora todo está listo por el lado del servidor.

Pruebas con el cliente "mail"

En los sistemas Linux el comando `mail` permite enviar y leer mensajes de correo electrónico mediante Sendmail. Por ejemplo, para enviar un mensaje a una cuenta exterior de test:

```
[diego@edithpiaf diego]$ mail test_user@yahoo.com
Subject: This is a test
```

```
There goes the challenger
.  
Cc:  
[diego@edithpiaf diego]$
```

La lectura de mensajes de correo recibidos se puede hacer también con el comando "mail":

```
[pedrito@correo pedrito]$ mail  
Mail version 8.1 6/6/93. Type ? for help.  
"/var/spool/mail/pedrito": 1 message 1 new  
>N 1 pepe@noskhon.com Sun Jan 27 06:24 "prueba"  
&  
Message 1:  
From root@noskhon.com Sun Jan 27 06:24 2002  
Delivered-To: pedrito@abejas.org  
Date: Mon, 25 Feb 2002 02:09:26 -0500  
From: pepe <pepe@noskhon.com>  
To: pedrito@abejas.org  
Subject: prueba
```

Esta es una prueba

&

Durante estas operaciones conviene verificar los mensajes del log desde otra ventana:

```
# tail -f /var/log/maillog
```

Véase el apéndice intitulado "Ejemplo de configuración de MUA" si requiere una idea acerca de cómo se configura un cliente gráfico de correo electrónico.

Conceptos

Esta sección proporciona una serie de conceptos que se utilizarán en lo que sigue.

Programas Involucrados

MUA/Cliente

El Mail User Agent o Cliente de correo electrónico es un programa que ejecutan los usuarios para leer y escribir sus mensajes. En la mayoría de casos se ejecuta en un computador personal. Este programa normalmente enviará los nuevos mensajes redactados por el usuario al servidor de la organización/proveedor, y descargará los mensajes pendientes para lectura del usuario desde el servidor de la organización/proveedor.

MTA/Servidor

El Mail Transfer Agent se encarga de enviar (y reintentar de ser necesario) los mensajes redactados por los usuarios de la organización. Igualmente, recibe los mensajes dirigidos a usuarios de la organización y los coloca en sus respectivas "casillas de correo" para su posterior lectura.

Protocolos

SMTP

El Simple Message Transfer Protocol se emplea para enviar mensajes de correo electrónico entre servidores. En muchos casos, el programa cliente de correo electrónico remite un nuevo mensaje al servidor usando también SMTP.

POP

El Post Office Protocol permite a los programas clientes de correo electrónico extraer los mensajes pendientes en las casillas de correo del usuario para que éste los pueda visualizar.

IMAP

El Internet Message Access Protocol, al igual que POP, permite a los programas clientes de correo electrónico extraer los mensajes pendientes en las casillas de correo del usuario para que éste los pueda visualizar. Tiene características adicionales a las proporcionadas por POP.

DNS

El Domain Name System permite que los mensajes de correo electrónico sean dirigidos al servidor correspondiente en el Internet. En particular, a partir de una dirección de correo electrónico de destino en un mensaje, se puede encontrar la dirección IP del computador que debe recibir dicho mensaje.

Mensajes

Header (cabecera)

Esta es una sección informativa que contienen todos los mensajes y que contiene datos relacionados a su envío, tales como el nombre y dirección electrónica del creador del mensaje, la lista de destinatarios, la fecha de envío, los servidores intermedios por donde el mensaje ha pasado, etc.

Body

Contiene el texto del mensaje en sí. Está compuesto por caracteres ASCII.

Envelope (sobre)

Contiene información usada para enrutar los mensajes, tal como los destinatarios inmediatos. Esta información normalmente tiene coincidencias con algunos componentes del header.

Attachment

Los archivos que no se componen de texto ASCII pueden ser enviados si primero se codifican como texto ASCII y se añaden ordenadamente a un mensaje normal. Estos añadidos al mensaje se denominan attachments.

Casilla del usuario

Los usuarios de correo electrónico no están conectados a la red durante todo el día, y lo mismo ocurre con sus computadores. Debido a esto, los mensajes que están dirigidos a ellos normalmente se almacenan en un área temporal denominada "casilla de correo" a la espera de que el usuario la extraiga cuando esté listo.

Relay

Corresponde a la facultad del MTA de reenviar los mensajes provenientes de un computador hacia otro computador. Por ejemplo, cuando un usuario de la red local le proporciona un mensaje que en realidad está dirigido hacia un usuario en Internet.

Empezando

Instalación

RedHat Linux

En los sistemas Linux RedHat, el servidor Sendmail se distribuye mediante el paquete "sendmail". Sin embargo, se hará bien en instalar otros paquetes adicionales tales como sendmail-cf y sendmail-doc, que proporcionan herramientas de configuración y documentación adicional, respectivamente. Adicionalmente se requerirá el paquete "m4".

La instalación en RedHat, como de costumbre, se hará mediante el comando RPM (o durante la instalación del sistema operativo, eligiendo "Mail Server" entre las opciones.) Aquí no explicaremos el comando RPM y nos limitaremos a mostrar cómo se puede verificar si los paquetes han sido instalados:

```
bash# rpm -qa|grep sendmail
sendmail-cf-8.11.2-14
sendmail-8.11.2-14
sendmail-doc-8.11.2-14
bash# rpm -q m4
m4-1.4.1-4
```

La versión de los paquetes indicados arriba puede ser distinta dependiendo de la versión de RedHat. El ejemplo anterior se refiere a una instalación RedHat 7.1.

Otros Sistemas Operativos

En casi todos los sistemas operativos Linux y Unix principales, Sendmail se distribuye por el mismo proveedor (posiblemente con algunas alteraciones.) En estos casos el método de instalación de paquetes puede variar, y se deberá consultar la documentación respectiva.

Desde la fuente

En RedHat y cualquier otro sistema operativo, siempre existe la posibilidad de descargar el código fuente de Sendmail a fin de compilarlo e instalarlo manualmente. En este caso deberá descargarse el archivo ".tar" de:

```
ftp://ftp.sendmail.org/ucb/sendmail
```

Al desempacarse este archivo, se encontrará documentos que explican el procedimiento de compilación e instalación (archivo READ_ME.)

Probando Sendmail

Asumiremos que Sendmail ya ha sido instalado. Para verificar la instalación y obtener cierta información básica, usaremos el siguiente comando, cuyo resultado se muestra para mi computador:

```
# sendmail -d0.1 -bt
Version 8.12.5
  Compiled with: DNSMAP HESIOD HES_GETMAILHOST LDAPMAP LOG MAP_REGEX
                MATCHGECOS MILTER MIME7TO8 MIME8TO7 NAMED_BIND NETINET NETINET6
                NETUNIX NEWDB NIS PIPELINING SASL SCANF STARTTLS TCPWRAPPERS
                USERDB USE_LDAP_INIT

===== SYSTEM IDENTITY (after readcf) =====
      (short domain name) $w = edithpiaf
      (canonical domain name) $j = edithpiaf.noskhon.com.pe
      (subdomain name) $m = noskhon.com.pe
      (node name) $k = edithpiaf.noskhon.com.pe
=====

ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
>
#
```

Para salir del prompt ">" se presionó [CTRL]+[D].

Nótese que este comando ha sido ejecutado por el administrador. Los usuarios normales normalmente deberán especificar la ruta completa del ejecutable "sendmail" a fin de obtener algo similar:

```
[diego@edithpiaf diego]$ /usr/sbin/sendmail -d0.1 -bt
Version 8.12.5
  Compiled with: DNSMAP HESIOD HES_GETMAILHOST LDAPMAP LOG MAP_REGEX
                MATCHGECOS MILTER MIME7TO8 MIME8TO7 NAMED_BIND NETINET NETINET6
... siguen más líneas ...
```

La opción "-bt" significa "modo de test", y el "-d0.1" significa "debug de aspectos generales (el cero), en nivel 1". Al modificar el nivel de debug se puede obtener más información. Por ejemplo, el lector podría observar la salida que presenta el comando anterior con "-d0.15".

Inicio automático

Como se aprecia, el servidor "sendmail" puede ser invocado en modo interactivo con diversos propósitos, sin embargo, lo usual es que opere en forma "no interactiva", o

como se suele decir en sistemas Unix, como un "demonio". Por lo general esto es configurado en los scripts de inicio del sistema operativo de modo tal que el "demonio sendmail" se inicie en forma automática cada vez que el computador es iniciado.

En un sistema RedHat Linux (versiones 7 en adelante) se puede emplear el comando "service" para invocar a estos scripts en cualquier momento. Por ejemplo, para consultar acerca del estado actual del servicio Sendmail:

```
[root@edithpiaf root]# service sendmail status
sendmail está parado
```

Para iniciarlo:

```
[root@edithpiaf root]# service sendmail start
Iniciando sendmail:           [ OK ]
Inicio de sm-client:         [ OK ]
```

Para detenerlo:

```
[root@edithpiaf root]# service sendmail stop
Apagando sendmail:           [ OK ]
Desactivación de sm-client:  [ OK ]
```

En otros sistemas habrá que consultar la respectiva documentación. El comando necesario podría ser como sigue:

```
bash# /etc/rc.d/init.d/sendmail start
```

o

```
bash# /etc/init.d/sendmail start
```

o

```
bash# /sbin/init.d/sendmail start
```

Y para detenerlo, reemplazar el "start" por un "stop".

Para que esto se ejecute automáticamente cuando el sistema se inicia, en RedHat se suele emplear los comandos `ntsysv` o `chkconfig`.

El log

Sendmail, como cualquier programa relacionado con el correo electrónico, genera mensajes de eventos (logs) mediante `syslog`. En los sistemas RedHat normalmente `syslog` está configurado para enviar los mensajes hacia el archivo `/var/log/maillog`. Cuando se hacen pruebas con Sendmail es muy conveniente tener una ventana o terminal abierta mostrando el log:

```
# tail -f /var/log/maillog
Oct 26 18:17:10 edithpiaf sendmail[1812]: h9QNHAM0001812:
from=root, size=421, class=0, nrcpts=1, msgid=<200310262317.
h9QNHAM0001812@edithpiaf.noskhon.com.pe>, relay=root@localhost
Oct 26 18:17:10 edithpiaf sendmail[1812]: h9QNHAM0001812:
to=root, ctladdr=root (0/0), delay=00:00:00, xdelay=00:00:00,
mailer=relay, pri=30065, relay=localhost.noskhon.com.pe.
[127.0.0.1], dsn=4.0.0, stat=Deferred: Connection refused
by localhost.noskhon.com.pe.
Oct 27 17:52:14 edithpiaf sendmail[2299]: h9RMqEA5002299:
from=root, size=244, class=0, nrcpts=1, msgid=<200310272252.
h9RMqEA5002299@edithpiaf.noskhon.com.pe>, relay=root@localhost
Oct 27 17:52:14 edithpiaf sendmail[2299]: h9RMqEA5002299:
to=root, ctladdr=root (0/0), delay=00:00:00, xdelay=00:00:00,
mailer=relay, pri=30065, relay=localhost.noskhon.com.pe.
```

```
[127.0.0.1], dsn=4.0.0, stat=Deferred: Connection refused by
localhost.noskhon.com.pe.
```

Configuración del sistema

Dividiremos esto en dos partes: configuración del host y configuración de programa Sendmail.

Configuración del host

A continuación algunos aspectos muy importantes relacionados al sistema operativo donde Sendmail se ejecutará. Esto NO es propiamente la configuración de Sendmail, pero éste requiere interactuar con diversos elementos del sistema operativo.

DNS

A fin de poder enviar mensajes a destinatarios remotos, Sendmail debe ser capaz de obtener la información necesaria de un servidor DNS. Incluso en una red local puede ser conveniente el empleo de un servidor DNS.

Si se desea desactivar completamente el uso del servidor DNS (por ejemplo, si nunca se saldrá a Internet) entonces se debe recompilar Sendmail con las opciones apropiadas (no explicaremos esto aquí.)

En la mayoría de computadores Unix/Linux, la dirección del servidor DNS que usan las aplicaciones se configura en el archivo `/etc/resolv.conf`. Por ejemplo, si su servidor DNS más cercano (el de la organización o el que ha proporcionado su proveedor) tiene dirección ip 100.2.3.4, entonces el archivo "resolv.conf" del computador destinado para ejecutar Sendmail debe lucir así:

```
nameserver 100.2.3.4
```

Por lo general se configuran dos o tres servidores DNS.

Por otro lado, a fin de que nos puedan enviar mensajes desde el exterior a nuestro servidor de correo, requerimos administrar un dominio (el dominio de la organización.) Esto implica que habrá un servidor DNS (posiblemente dentro de nuestra organización o administrado por un proveedor) que contenga la configuración de nuestra zona.

Si el dominio de nuestra organización es "laorganizacion.org", es frecuente definir que las direcciones de los usuarios locales tendrán la forma "usuario@laorganizacion.org". En algunos lugares, prefieren direcciones similares a "usuario@mail.laorganizacion.org" aunque esto es a gusto de la organización.

Esto debe reflejarse en la configuración de nuestro dominio en el servidor DNS. Asumiendo que el servidor DNS que administra nuestra zona usa el software BIND, (puede ser cualquier otro) el archivo de la zona "organizacion.org" debería contener al menos estas líneas para que las direcciones de formato "usuario@laorganizacion.org" llegen al servidor.

```
@ 1D IN MX 0 correo
correo 1D IN A 90.8.7.6
```

Si se desea el segundo formato ("usuario@mail.laorganizacion.org"), las líneas correspondientes serían algo como:

```
mail 1D IN MX 0 correo
correo 1D IN A 90.8.7.6
```

Nosotros no detallaremos más la configuración del servidor DNS por ser un tema fuera del ámbito que nos compete. En [2] se puede encontrar una excelente explicación de todo esto.

Nótese que esto último se hará muy probablemente en un computador distinto al que ejecuta Sendmail.

Archivo hosts

El archivo `/etc/hosts` es complementario al sistema DNS. Para su correcta operación Sendmail normalmente requiere que el computador en que se ejecuta tenga una configuración como la siguiente:

```
127.0.0.1 localhost
90.8.7.6 correo.laorganizacion.org
```

Nótese que esto normalmente NO viene así en RedHat.

La dirección IP suministrada debe coincidir con lo que se configuró en el DNS, y el nombre del host debe ser "full", es decir, debe incluir el nombre del dominio.

Hostname

El nombre del computador donde se ejecuta Sendmail debe corresponder a lo configurado en el DNS y el archivo hosts. Lamentablemente, la configuración de este parámetro varía de sistema en sistema. Por ejemplo, en RedHat, la configuración del hostname se efectúa en el archivo `/etc/sysconfig/network` en la variable `HOSTNAME`.

```
[root@edithpiaf root]# cat /etc/sysconfig/network
NETWORKING=yes
HOSTNAME=edithpiaf.noskhon.com.pe
GATEWAY=172.192.12.1
```

La forma más fácil -pero no la única- de proseguir tras modificar el hostname, consiste en reiniciar el computador.

Configuración del programa Sendmail

Sendmail es extremadamente configurable -aunque no necesariamente de un modo sencillo. Para esto posee un archivo de configuración principal que en RedHat 8 es:

```
/etc/mail/sendmail.cf
```

aunque en otros sistemas Linux/Unix es:

```
/etc/sendmail.cf
```

o incluso:

```
/var/adm/sendmail/sendmail.cf
```

La ruta exacta de este archivo normalmente se define durante la compilación de Sendmail. Es conveniente que el lector lo encuentre antes de proseguir.

Este archivo de configuración de aquí en adelante será llamado el archivo "cf" por la extensión de su nombre.

Como quizá ya haya observado el lector, el archivo "cf" tiene una sintaxis poco intuitiva, y ha sido diseñado principalmente para que el computador lo lea de un modo eficiente (mas no los humanos.)

El archivo "cf" define generalmente la ruta de otros archivos de configuración auxiliares que evitan la modificación directa del primero, simplificando la administración de Sendmail.

En las últimas versiones de sendmail (8.12 o superiores) es usual que se configure el servidor para que se ejecute "dividido" en dos programas complementarios a fin de elevar la seguridad del sistema. La siguiente salida (recortada) de RedHat Linux 8.0 ilustra esta idea:

```
# ps axuw|grep sendmail
root 16216 ? S 15:43 sendmail: accepting connections
smmsp 16226 ? S 15:43 sendmail: Queue runner@01:00:00 for /var/spool/clientmqueue
```

En este caso, el segundo proceso (client queue runner) es controlado mediante otro archivo de configuración:

```
/etc/mail/submit.cf
```

Obviamente, en otros sistemas este archivo se puede encontrar en otros directorios.

Sistema de configuración M4

Motivación

Si el lector tuvo curiosidad de listar el archivo "cf", habrá notado seguramente que éste tiene una sintaxis muy poco intuitiva. Este problema no ha pasado desapercibido para los desarrolladores de Sendmail (aunque la cura quizá haya resultado peor que la enfermedad:)

A fin de facilitar la configuración de Sendmail para los usuarios ocasionales y los administradores en general, existe un mecanismo complementario que evita la escritura y modificación directa del archivo "cf". Este mecanismo consiste en escribir un archivo relativamente sencillo usando la sintaxis del lenguaje "M4", el cual se proporciona en prácticamente todos los sistemas Unix/Linux (a veces como software opcional.)

Mediante este sistema, el usuario creará (o modificará) un archivo relativamente breve, el cual se traducirá en muchas líneas del archivo "cf".

Lo cierto es que es absolutamente impráctico escribir "desde cero" un archivo "cf" medianamente utilizable, así que el método M4 es una opción casi obligatoria.

Regenerando el archivo "cf"

Antes de hacer modificaciones, es recomendable conocer cómo se generó el archivo "cf" proporcionado por el sistema. Normalmente éste proviene de un archivo tipo "M4". Lamentablemente esto no es válido en todos los casos, y las rutas de los archivos involucrados son muy variables.

En RedHat 8 el archivo "cf" distribuido (/etc/mail/sendmail.cf) se puede regenerar en cualquier momento a partir del archivo (/etc/mail/sendmail.mc) que usa la sintaxis "M4". Esto se puede hacer con el siguiente comando:

```
# cd /etc/mail
# m4 sendmail.mc > sendmail.cf
```

* NOTA: Asegúrese de sacar una copia al archivo "cf" antes de hacer esto!

En RedHat 7 la secuencia es parecida, aunque los directorios difieren:

```
bash# cd /usr/share/sendmail-cf/cf
bash# m4 redhat.mc > /etc/sendmail.cf
```

En otros sistemas Unix/Linux (incluso RedHat en versiones anteriores) el archivo `sendmail.mc` puede tener un nombre distinto y una ubicación distinta, y habrá que ver la documentación respectiva. Por ejemplo, en RedHat 7 se llamaba `redhat.mc` y se ubicaba en `/usr/share/sendmail-cf/cf`.

Incluso puede ser que este archivo simplemente no exista y haya que generar uno nuevo. En ese caso Ud. deberá ubicar primero el directorio "cf" de Sendmail y crear un archivo (le llamaremos `prueba.mc`) tal como:

```
include(`../m4/cf.m4')
OSTYPE(hpux10)dnl
MAILER(local)dnl
MAILER(smtp)dnl
```

La directiva `OSTYPE` requiere que Ud. especifique su versión de sistema operativo (en el ejemplo, HP/UX V.10.) Para ver los sistemas disponibles, vea el directorio `sendmail-cf/ostype` o `cf/ostype`. En mi caso:

```
bash# ls ../ostype/
aix2.m4      bsdi2.0.m4  irix4.m4    powerux.m4
aix3.m4      bsdi.m4     irix5.m4    ptx2.m4
aix4.m4      darwin.m4   irix6.m4    qnx.m4
aix5.m4      dgux.m4     isc4.1.m4   riscos4.5.m4
altos.m4     domainos.m4 linux.m4     sco3.2.m4
amdahl-uts.m4 dynix3.2.m4 maxion.m4   sco-uw-2.1.m4
aux.m4       gnu.m4      mklinux.m4  sinix.m4
bsd4.3.m4    hpux10.m4   nextstep.m4 solaris2.m4
bsd4.4.m4    hpux11.m4   openbsd.m4  solaris2.ml.m4
bsdil.0.m4   hpux9.m4    osf1.m4     solaris2.pre5.m4
```

Todo esto requiere que se trabaje en el subdirectorio `sendmail-cf/cf` o `cf/cf`. A continuación, ejecutar `m4`:

```
bash# m4 prueba.mc > /etc/sendmail.cf
```

Volviendo a Linux RedHat, los archivos "M4" usados por Sendmail se proporcionan en el paquete "sendmail-cf". Obviamente requerirá también el paquete "m4" para poder usarlo. En otros sistemas Unix/Linux el software "M4" puede ser opcional o parte de las herramientas de desarrollo.

En Linux RedHat 8.0 y superiores, es también posible regenerar el archivo "submit.cf" a partir de:

```
# cd /etc/mail
# m4 submit.mc > submit.cf
```

Configuración con M4

El sistema M4 de Sendmail permite generar configuraciones para distintos propósitos así como alterar opciones bastante puntuales. A modo de ejemplo, el parámetro que

controla el "tiempo de alerta" de un mensaje en cola (no se preocupe si no entiende esto, es sólo un ejemplo), se configura con M4 mediante una línea como la siguiente:

```
define( `confTO_QUEUEWARN' , `2h' )
```

Lo cual se traduce en exactamente la siguiente línea en el archivo "cf" generado:

```
O Timeout.queuewarn=2h
```

Téngase cuidado dentro del archivo "M4" de emplear las comillas adecuadas para cada caso (obsérve que se han usado ambos tipos:

```
` y '
```

En resumen, mediante la sintaxis (simple) de "M4", se puede regenerar un archivo en la sintaxis (compleja) del "cf". Recuérdese que al final, el programa Sendmail sólo utilizará el archivo "cf".

En lo que sigue, presentaremos la configuración de Sendmail empleando ambos métodos cuando sea posible, pero se preferirá el método M4. Como se indicó, algunas directivas del método "M4" se traducen a una *gran cantidad* de complejas directivas del archivo "cf", el cual resulta impráctico.

El Procesamiento de los mensajes

Los mensajes de correo electrónico tienen principalmente dos tipos de procesamiento:

- Local: Cuando el destinatario es un usuario perteneciente a nuestro servidor de correo electrónico
- Remoto: Cuando el destinatario está ubicado en otro servidor

Estas "rutas" de envío de mensaje se suelen denominar "mailers" y son implementadas por "delivery agents".

Envíos locales

Sendmail normalmente determina si un mensaje es local cuando la dirección de destino contiene sólo un nombre de usuario (por ejemplo, "diego"). Asimismo, cuando la dirección de destino contiene un host que coincide con el "hostname" del servidor. Por ejemplo, en la dirección "diego@correo.caligula.net" el host especificado (lo que sigue a la @) es "correo.caligula.net"; si este coincide con el hostname del computador, entonces el mensaje debe ser considerado local.

Una excepción (extremadamente común) a lo último consiste en forzar que ciertas direcciones sean tratadas como locales. Para esto, el usuario generalmente configurará el archivo `/etc/mail/local-host-names` con los nombres de host considerados "locales" o "sinónimos" del host local. Por ejemplo, para que el mensaje con destino "diego@neron.org" sea también considerado local, habría que incluir en el archivo `/etc/mail/local-host-names`:

```
neron.org
```

Esto es muy importante, puesto que típicamente las organizaciones deciden utilizar una dirección de email terminada en "@dominio", donde "dominio" casi nunca coincide exactamente con el hostname del servidor. De no configurarse el archivo `local-`

host-names, todos los mensajes que lleguen al servidor con esta dirección serían considerados no locales, y por tanto serían rechazados.

En otros sistemas el nombre del archivo de configuración local-host-names puede variar. Para averiguar o verificar esto, debemos consultar la documentación o en último caso el archivo "cf":

```
# grep "^Fw" /etc/mail/sendmail.cf
Fw/etc/mail/local-host-names
```

En versiones anteriores de Sendmail, este archivo se denominaba /etc/sendmail.cw.

Si se dispone del archivo "M4" que dio origen al "cf", deberíamos buscar allí una línea tal como:

```
FEATURE(`access_db', `hash -T<TMPF> -o /etc/mail/access.db')dnl
```

En algunos casos (no en RedHat), esta funcionalidad puede no estar definida y se debe activar.

Sin embargo, esta configuración también puede hacerse en el mismo archivo "cf" con el comando "class-macro-w" (clase w.) Por ejemplo, las siguientes líneas muestran que las direcciones terminadas en "@localhost" y "@localhost.localdomain" también son consideradas locales.

```
# grep "^Cw" /etc/mail/sendmail.cf
Cwlocalhost.localdomain
Cwlocalhost
```

En otras palabras, la "clase w" se puede considerar un "array" que contiene los nombres de las direcciones consideradas "locales". Esta clase se puede configurar en el archivo "cf" mediante el comando "C" (seguido por el nombre de la clase, o sea "Cw") o mediante un archivo externo (definido por el comando "F", en nuestro caso "Fw").

Los envíos locales conllevan a que los mensajes sean almacenados en la "casilla de correo" del usuario destinatario. El usuario deberá extraer estos mensajes con su programa "cliente", posiblemente mediante los protocolos IMAP o POP como se verá después.

La casilla de correo (inbox) es simplemente un archivo cuyo nombre coincide con el del usuario y se ubica en el directorio /var/spool/mail (como siempre, esto puede variar en otros Unix.) Como veremos, Sendmail no escribe directamente en este archivo y por tanto no es parte de su configuración.

Definición del envío local

Como se indicó, el envío local conlleva a que el mensaje sea guardado en el archivo "inbox" o "casilla de correo". Sin embargo, Sendmail no realiza directamente este trabajo, sino que invoca a un programa auxiliar para esta tarea (aparentemente sencilla.)

Esto se configura en el archivo "cf" con la definición "Mlocal" (mailer local.) En el archivo "cf" de RedHat 8.0, esta sección luce así:

```
Mlocal, P=/usr/bin/procmail, F=lsDFMAw5:/|@qSPfhn9,
  S=EnvFromL/HdrFromL,
  R=EnvToL/HdrToL,
  T=DNS/RFC822/X-Unix,
  A=procmail -t -Y -a $h -d $u
```

Las líneas que se inician con espacios o tabuladores se consideran la continuación de la anterior, por tanto, el extracto mostrado arriba debe considerarse una sola línea.

Como se aprecia, Sendmail utiliza un programa auxiliar llamado `/usr/bin/procmail` para el delivery local. En muchos sistemas Unix el programa usado es `/bin/mail`. No profundizaremos más sobre esto por ahora, pero en general "procmail" es mucho más flexible y eficiente que las alternativas tradicionales.

Envíos Remotos

Sendmail tiene la capacidad de hacer envíos remotos usando el protocolo SMTP (que opera usando TCP/IP.) En este caso, Sendmail no emplea un programa auxiliar como en los envíos locales.

Cuando un mensaje no puede ser enviado a uno de los destinatarios, el mensaje es "encolado" para un posterior reintento.

Para efectuar el envío remoto, Sendmail requiere que los destinatarios del mensaje posean una "parte de host" distinta al hostname local (o a sus sinónimos de la clase-w, como se vio en la sección anterior.)

Sendmail normalmente utilizará el DNS a fin de encontrar el servidor remoto que recibirá el mensaje (específicamente, el registro MX y el registro A) y abrirá una conexión SMTP.

El archivo "cf" define este envío con una línea como la que sigue.

```
Msmtp, P=[IPC], F=mDFMuX, S=EnvFromSMTP/HdrFromSMTP, R=EnvToSMTP,
E=\r\n, L=990,
T=DNS/RFC822/SMTP,
A=TCP $h
```

De estos ejemplos se puede apreciar que las definiciones de los "mailers" o "agentes de delivery" siempre empiezan con el comando "M", inmediatamente seguido por el nombre del "mailer".

"mailers" en M4

Los mailers en el archivo M4 deben aparecer al final del archivo fuente M4. Para generar los dos mailers mencionados arriba, simplemente se requiere:

```
MAILER(smtp)dnl
MAILER(procmail)dnl
```

Esto lo puede observar el lector en el archivo "sendmail.mc" proporcionado por Red-Hat.

Administración de la cola de Sendmail

Conceptos

Los mensajes que procesa Sendmail muchas veces no pueden ser enviados a su destino en forma inmediata. Por ejemplo, la línea que conecta nuestra organización a Internet puede estar detenida, o el mail server remoto puede haberse tornado inaccesible.

En estos casos Sendmail "encolará" el mensaje a fin de intentar hacer posteriores reintentos a intervalos predefinidos. Cumplido cierto tiempo, Sendmail desistirá de reintentar y el envío se considera fallido (el usuario que envía recibe una notificación.)

En RedHat 8 la cola de Sendmail se localiza en el directorio /var/spool/mqueue. Cada mensaje se compone de dos archivos (con prefijo "qf" y "df") que corresponden, respectivamente, al "archivo de control" y el "archivo de datos". El archivo de control contiene información respecto al envío del mensaje, particularmente el header. El "archivo de datos" contiene el cuerpo del mensaje.

El directorio de cola se configura en el archivo "cf" del siguiente modo:

```
O QueueDirectory=/var/spool/mqueue
```

En el método M4 se usará (sólo si se desea modificar el default):

```
define( `QUEUE_DIR' , ` /var/spool/mqueue' )
```

Temporización del queue run

El "queue run" corresponde a un intento repetitivo que hace Sendmail para enviar los mensajes que por algún motivo quedaron encolados. El intervalo entre queue run's es configurable en las opciones de línea de comando de Sendmail. Por ejemplo, "sendmail -bd -q3h" significa que Sendmail se ejecutará en background y que el procesamiento de la cola (queue run) se hará cada tres horas (3h.) Este lapso se configura en RedHat en el archivo que se muestra.

```
# cat /etc/sysconfig/sendmail
DAEMON=yes
QUEUE=1h
```

En otros sistemas, el lector deberá indagar por los scripts de inicio del servicio Sendmail para configurar el lapso de queue run.

Forzar el procesamiento de la cola

Si deseamos forzar el procesamiento de la cola completa (por ejemplo, si de pronto nuestra conexión a Internet se ha restablecido tras un corte), en cualquier momento esto se puede hacer con el comando:

```
# sendmail -q
```

Sin embargo, a fin de intentar el envío de sólo un mensaje en particular (o un conjunto de éstos), se puede usar lo siguiente:

```
# sendmail -qIsubstr
```

Donde "substr" es un substring del queue ID del mensaje a enviar (o de los mensajes.) Por ejemplo, si tenemos los siguientes mensajes encolados:

```
# sendmail -bp
/var/spool/mqueue (4 requests)
-----Q-ID----- --Size-- -----Q-Time----- -----Sender/Recipient-----
-----
h8Rkt4UX001163      16 Sat Sep 27 15:55 <root@edithpiaf.noskhon.com.pe>
                    (Deferred: Name server: kika.noskhon.com.pe.: host name looku)
                    <diego@kika.noskhon.com.pe>
h8Rkt4UZ001163      10 Sat Sep 27 15:55 <root@edithpiaf.noskhon.com.pe>
                    (Deferred: Name server: kika.noskhon.com.pe.: host name looku)
                    <diego@kika.noskhon.com.pe>
h8PNiopq002785     298 Thu Sep 25 18:44 root
                    (Warning: could not send message for past 4 hours)
                    diego@kika.noskhon.com.pe
```

```
h8PNh4RM002734      298 Thu Sep 25 18:43 root
                    (Warning: could not send message for past 4 hours)
                    diego@kika.noskhon.com.pe
Total requests: 4
```

Y queremos enviar los dos primeros de la lista (que tienen bastante tiempo en la cola), podemos especificar su Q-ID completo o un substring del mismo. Por ejemplo, puesto que sus Q-ID son "h8RKt4UX001163" y "h8RKt4UZ001163", podremos emplear el substring "1163" así:

```
# sendmail -q1163
[root@edithpiaf root]# sendmail -bp
                        /var/spool/mqueue (2 requests)
-----Q-ID----- --Size-- -----Q-Time----- -----Sender/Recipient-----
-----
h8PNi0pq002785      298 Thu Sep 25 18:44 root
                    (Warning: could not send message for past 4 hours)
                    diego@kika.noskhon.com.pe
h8PNh4RM002734      298 Thu Sep 25 18:43 root
                    (Warning: could not send message for past 4 hours)
                    diego@kika.noskhon.com.pe
Total requests: 2
```

Nota: Como se indicó anteriormente, en las versiones recientes Sendmail viene dividido en "dos partes", una de las cuales se encarga del procesamiento de los mensajes enviados desde la línea de comando del servidor (sm-client) y otra de los mensajes recibidos desde la red. Ambas tienen su propia cola. A fin de ver la cola del "sm-client" es necesario usar:

```
# sendmail -Ac -bp
```

Orden de envío de los mensajes

Normalmente Sendmail ordena los mensajes a ser enviados durante un `queue run` de acuerdo a la "prioridad" del mensaje. La prioridad se calcula mediante:

```
PRI=msgsize-(class*ClassFactor)+(nrcpt*RecipientFactor)+(nrun*RetryFactor)
```

Donde `msgsize` es el tamaño en bytes del mensaje; "class" significa el valor numérico asignado mediante a las opciones "Precedence" en el header (por ejemplo, los mensajes para listas deben tener baja precedencia), y "nrcpt" es el número de destinatarios del mensaje. `ClassFactor` y `RecipientFactor` son factores de ajuste (modificables) para proporcionar un peso relativo (normalmente 1800 y 3000 respectivamente.) Un valor más elevado de este resultado en realidad significa que el mensaje es tratado con menos prioridad.

Por último, en cada intento de envío (cada `queue run`) un mensaje que no pudo ser enviado sufre una alteración en su prioridad en "RetryFactor". El default para este parámetro es 90000.

El ordenamiento de la cola en base a la prioridad es la política por omisión de Sendmail; sin embargo, la opción `QueueSortOrder` permite alterar esto. Por ejemplo,

```
QueueSortOrder=host
```

O en M4:

```
define('confQUEUE_SORT_ORDER', 'host')
```

Encolará los mensajes por el host de destino, estado de conexión con el host (primero se aprovechan las conexiones abiertas con el host) y luego la prioridad.

Esto es recomendado en conexiones de alta velocidad.

Dominios virtuales

En esta sección veremos la forma en que podamos administrar las cuentas de usuarios en diversos dominios. Por simplicidad, supondremos que los dominios son sólo dos: "incacoca.com" y "dbe.org.pe". Las cuentas existentes deben ser:

```
oscar@incacoca.com
ana@incacoca.com
alex@incacoca.com
jose@incacoca.com

diego@dbe.org.pe
ana@dbe.org.pe
DiegoMaradona@dbe.org.pe
```

Cuentas en el sistema

Lo primero que haríamos en el caso estándar de un solo dominio es crear las cuentas de todos los usuarios:

```
# useradd oscar
# useradd ana
...
```

Pero hay dos inconvenientes. En primer lugar, hay dos cuentas distintas (para dos personas distintas) con el mismo usuario "ana". En segundo lugar, la cuenta "Diego-Maradona" no es válida en la medida que Sendmail intentará enviar los mensajes de este destinatario a "diegomaradona" (en minúsculas.)

Una solución a este dilema consiste en asociar nombres de usuario totalmente independientes de la dirección, más o menos del siguiente modo:

```
oscar@incacoca.com -> vdu0001
ana@incacoca.com -> vdu0002
alex@incacoca.com -> vdu0003
jose@incacoca.com -> vdu0004

diego@dbe.org.pe -> vdu0005
ana@dbe.org.pe -> vdu0006
DiegoMaradona@dbe.org.pe -> vdu0007
```

Cualquier otra dirección en estos dominios (o cualquier nuevo dominio) recibirá así un "username" formado por la palabra "vdu" y un número secuencial ("vdu" es un prefijo cualquiera que acabo de imaginar. Para mí significa Virtual-Domain-User.)

Por tanto, crearemos los usuarios del siguiente modo:

```
# useradd vdu0001
# passwd vdu0001
...
# useradd vdu0007
# passwd vdu0007
```

Asociar direcciones electrónicas a las cuentas

Para esto se emplea el archivo "virtusertable" localizado en el directorio "/etc/mail". Allí colocaríamos simplemente:

```
oscar@incacoca.com vdu0001
ana@incacoca.com vdu0002
alex@incacoca.com vdu0003
jose@incacoca.com vdu0004
diego@dbe.org.pe vdu0005
ana@dbe.org.pe vdu0006
DiegoMaradona@dbe.org.pe vdu0007
```

Luego ejecutaríamos "make" (estando todavía en el directorio "/etc/mail") para regenerar la versión compilada.

El archivo "cf" que proporciona RedHat ya incluye la referencia a "virtusertable". Si se partiera de cero, lo más conveniente es usar el método "M4" incluyendo una línea como la que sigue en el archivo "sendmail.mc":

```
FEATURE(`virtusertable', `hash -o /etc/mail/virtusertable.db')dnl
```

Lo cual genera la referencia a este archivo así como los rulesets necesarios para aprovecharlo.

DNS

Como cabría de esperarse, en el DNS debe configurarse los registros necesarios para que el correo de forma user@incacoca.com y user@dbe.org.pe se redirija a nuestro servidor. Esto se hace configurando el registro MX en los archivos de configuración de esas zonas. Si nuestro servidor es "correo.laorganizacion.org", la configuración de la zona "incacoca.com" debería contener:

```
@ 1D IN MX 0 correo.laorganizacion.org.
```

La zona "dbe.org.pe" tendría algo similar.

Local-host-names

Es necesario también indicar a Sendmail que los mensajes de dominios "incacoca.com" y "dbe.org.org" deben ser aceptados. Para esto, incluirlos en el archivo "/etc/mail/local-host-names".

Como siempre, la apertura del relay dependerá de dónde se ubican los clientes, cosa que no se repetirá aquí.

Configuración del MUA

La única diferencia con el caso "mono-dominio" en lo que compete al MUA, corresponde a la configuración para RECIBIR el correo. Ya sea POP o IMAP, hay que indicar que la recepción se debe hacer con los usuarios "vduXXXX".

Por otro lado, el ENVÍO sí se debe hacer con la dirección electrónica completa (por ejemplo alex@incacoca.com)

Enmascaramiento

En una red conformada por un servidor de correo y un conjunto de estaciones (como nuestro dominio "laorganizacion.org") los usuarios de seguro redactan sus mensajes usando MUA's (clientes) que les permiten (y les exigen) especificar la dirección de correo con la que el mensaje aparente haberse originado (campo "From" en el header.) Esto se usa por lo general para que el destinatario pueda respondernos.

En nuestro caso, los usuarios configurarán sus direcciones de origen con algo como "user@laorganizacion.org".

Sin embargo, hay algunos MUA's que no permiten especificar la dirección de origen (por extraño que esto suene.) Por ejemplo, el comando "mail" disponible en la mayoría de sistemas Unix permite enviar mensajes usando Sendmail, pero no especifica dirección de origen (por lo que Sendmail tiene que "imaginarla".)

A fin de no variar el escenario, supongamos que un usuario llamado "oscarin" se la conectado (usando telnet) al servidor de correo - no importa el motivo - y desea enviar un mensaje. Invoca a "mail" del siguiente modo:

```
oscarin$ mail pepebotella@yahoo.com
```

Sendmail tendrá ahora que crear el campo "From" del header, y lo hará usando el hostname (en nuestro caso, correo.laorganizacion.org), por lo que el correo que llega a "pepebotella@yahoo.com" aparece como proveniente de:

```
oscarin@correo.laorganizacion.org
```

No está tan mal, pero no es lo correcto.

A fin de corregir esta situación, simplemente definiremos lo siguiente en nuestro archivo M4 "sendmail.mc":

```
MASQUERADE_AS(`laorganizacion.org')
```

Lo que logrará que el mensaje aparezca (correctamente) como enviado de:

```
oscarin@laorganizacion.org
```

Como siempre, regenerar el archivo "cf" con el comando "m4" y reiniciar Sendmail.

Nótese que esto puede no funcionar adecuadamente con el usuario "root" (para facilitar ciertos diagnósticos) por lo que se deben emplear usuarios comunes.

Recuérdese que esto sólo es necesario cuando se emplean MUA's con "problemas" en el sentido explicado. Más adelante veremos otros casos más interesantes de enmascaramiento.

Sitios con más de un servidor

Redundancia

Si nuestro servidor de email de pronto sufre un desperfecto, nuestros mensajes de email no podrán salir. Esto es grave, pero se puede suplir con llamadas telefónicas u otros medios.

Sin embargo, más grave es el no responder a los mensajes que nos envían. En ciertos casos, los servidores remotos intentarán reenviarnos los mensajes (durante cierto tiempo.) En otros, puede que esto nunca ocurra y que los mensajes "reboten" inmediatamente. En cualquier caso, es muy grave el hecho de perder estos mensajes. Aquí analizaremos algunas medidas destinadas a contrarrestar estos inconvenientes.

Un servidor local adicional

Si nuestro servidor de email deja de operar por un problema (cualquiera) del computador, una forma de mantener el servicio consiste en disponer de un servidor de respaldo ubicado al interior de nuestra organización el cual puede continuar recibiendo los mensajes que nos envían.

Esto es relativamente sencillo de implementar añadiendo una entrada en el DNS:

```
@ MX 10 mailserver1
  MX 20 mailserver2
```

Aquí `mailserver1` es el servidor "normal" que recibe los mensajes, mientras que `mailserver2` es el "backup". Esta configuración ocasionará que los mensajes que llegan del exterior y no pueden ser enviados a `mailserver1` sean enviados a `mailserver2`.

Frecuentemente estos servidores guardarán los mensajes en las casillas de email de los usuarios destinatarios respectivos, a fin de que éstos los recojan vía protocolos POP o IMAP. El problema es que la mayoría de clientes de email POP/IMAP sólo se pueden configurar para acceder a un servidor a la vez.

En otras palabras, si `mailserver1` deja de operar y `mailserver2` toma la posta, entonces todos los usuarios de la red local deberán ser reconfigurados para que apunten a `mailserver2`.

Esto puede ser aceptable en ciertas circunstancias, como en el caso de tener pocos clientes.

A fin de evitar esto, es posible hacer algunos artificios asumiendo que `mailserver1` está inoperativo.

Si nuestros clientes están apuntando a `mailserver1` mediante su dirección IP, entonces se puede asociar temporalmente esta dirección IP a `mailserver2` por medio de un "IP virtual" o "IP alias".

Si nuestros clientes están apuntando a `mailserver1` mediante su nombre (vía DNS) entonces se puede modificar temporalmente la configuración del DNS para que `mailserver1` apunte al IP de `mailserver2` (cambiar el registro A.)

Con esto los clientes accederán a `mailserver2` de modo casi transparente... sin embargo, si usan IMAP y han creado carpetas en el servidor, obviamente estas carpetas no podrán verse (pues se quedaron en `mailserver1`.) Informe a sus usuarios de esta situación.

Otro inconveniente radica en el restablecimiento de `mailserver1`. Ciertos usuarios pueden haber dejado mensajes sin leer en `mailserver2` que deberán ser trasladados manualmente a `mailserver1` para que estén disponibles en su INBOX. Herramientas como `fetchmail` pueden ayudar en estos casos.

En general, si `mailserver1` va a ser interrumpido por sólo unos minutos, es mejor que `mailserver2` NO se haga cargo del email entrante por los inconvenientes señalados. Una solución más adecuada (y costosa) consiste en que ambos servidores compartan un área de almacenamiento externo compartido.

Un servidor remoto

En el caso de que nuestra conexión a Internet se interrumpa, o en caso de un desastre general en nuestra organización, conviene disponer de un servidor ubicado en un lugar geográficamente distante y accesible mediante proveedores distintos (a fin de aumentar la redundancia.) En algunos casos, este servidor puede ser el de otra organización, con la que pactaremos este servicio.

A la configuración anterior del DNS, añadiremos otra línea:

```
@ MX 10 mailserver1
  MX 20 mailserver2
  MX 30 email-friend-store.com.
```

Como se ve, una última opción de envío consiste en que los mensajes lleguen al servidor "email-friend-store.com". Normalmente este servidor rechazaría nuestros mensajes (pues nuestras direcciones terminan en @laorganizacion.org.) Pero, puesto que hemos hecho un acuerdo previo, en "email-friend-store.com" han añadido la siguiente línea en su archivo access:

```
laorganizacion.org RELAY
```

Ahora, en caso de que nuestra conexión se interrumpa, los MTA's remotos intentarán como última opción a email-friend-store.com, el cual recibirá nuestros mensajes y tratará (infructuosamente) de reenviarlos a nuestros servidores mailserver1 y mailserver2. Como no puede, los encolará y los intentará reenviar posteriormente (ver la sección dedicada al encolamiento para más información.)

Organización con divisiones administrativas

En organizaciones muy grandes es frecuente que se establezcan áreas relativamente interdependientes pero separadas. Por ejemplo, geográficamente.

A fin de optimizar el rendimiento, el diseño de la configuración de email debe aprovechar estas características.

Solución trivial: Diversos dominios

Supondremos que nuestra organización se llama "inkacoca" y que tiene sucursales en las ciudades de Lima, Trujillo, Cuzco, Iquitos y Puerto Maldonado. Asumimos que la organización ha adquirido la autoridad del dominio "inkacoca.org".

En ese sentido, una manera de operar sería crear los siguientes subdominios, que corresponderían a las direcciones email respectivas:

Tabla 1.

Ciudad	Dominio	Dirección email
Lima	lima.inkacoca.org	usuario@lima.inkacoca.org
Trujillo	trujillo.inkacoca.org	usuario@trujillo.inkacoca.org
Cuzco	cuzco.inkacoca.org	usuario@cuzco.inkacoca.org
Iquitos	iquitos.inkacoca.org	usuario@iquitos.inkacoca.org
Puerto Maldonado	pmaldonado.inkacoca.org	usuarios@pmaldonado.inkacoca.org

Luego se configurarían servidores de email independientes para cada ciudad. Desde

el punto de vista del email, cada subdominio viene a ser una organización independiente. En cada servidor (en cada ciudad) el administrador crea sus propias cuentas independientes.

El DNS deberá contener entradas independientes para cada uno de estos servidores (aquí llamados lima-mail, tru-mail, cuzco-mail, iqui-mail, pto-mail.)

```
; archivo de zona de inkacoca.org
lima    IN MX 10 lima-mail
lima-mail IN A 18.1.3.40
trujillo IN MX 10 tru-mail
tru-mail IN A 18.1.4.40
cuzco   IN MX 10 cuzco-mail
cuzco-mail IN A 18.1.5.40
iquitos IN MX 10 iqui-mail
iqui-mail IN A 18.1.6.40
pmaldonado IN MX 10 pto-mail
pto-mail IN A 18.1.7.40
```

Los problemas de este esquema son:

1. No se está reflejando el dominio único de la organización (nadie tiene cuenta de la forma usuario@inkacoca.org)
2. Las direcciones de email son muy largas

Un sólo dominio

Tratemos de mejorar el esquema anterior. Intentemos que todas las direcciones sean de la forma usuario@inkacoca.org, manteniendo los cinco servidores en cada ciudad. El primer inconveniente de este esquema es que si tenemos dos usuarios llamados "diego" en Lima y Cuzco, y ambos originalmente tenían direcciones:

```
diego@lima.inkacoca.org
diego@cuzco.inkacoca.org
```

ahora habrá que buscar una forma de diferenciarlos. Por ejemplo, podríamos usar "diego1" y "diego2". Esto, si bien no es estéticamente agradable, puede ser mejorado con el uso de alias. Pero dejaremos esto para más adelante.

Ahora bien, si un mensaje llega desde el exterior a "usuario@inkacoca.org", ¿qué servidor lo recibe?

Una posibilidad es designar un servidor adicional (ubicado en cualquier ciudad) para que sirva como "switch", aunque podría ser cualquiera de los otros, como veremos.

Este servidor deberá ser capaz de redirigir el mensaje al servidor adecuado. Es decir, deberá disponer de una tabla similar a:

Tabla 2.

Usuario	Destino
diego1	lima.inkacoca.org
diego2	cuzco.inkacoca.org

Esto es precisamente lo que hace el archivo virtusertable que se discutió anteriormente en la sección "dominios virtuales".

Para esto, el archivo `/etc/mail/virtusertable` contendrá algo como:

```
diego1@incakoca.com diego1@lima.incakoca.org
diego2@incakoca.com diego2@cuzco.incakoca.org
```

Recuérdese que se debe generar la versión "compilada" como se vio anteriormente.

Adicionalmente se requieren registros en el DNS para el "mail switch":

```
; archivo de zona de incakoca.org
@ IN MX 10 switch
switch IN A 18.1.3.41
lima IN MX 10 lima-mail
lima-mail IN A 18.1.3.40
trujillo IN MX 10 tru-mail
tru-mail IN A 18.1.4.40
cuzco IN MX 10 cuzco-mail
cuzco-mail IN A 18.1.5.40
iQUITOS IN MX 10 iqui-mail
iqui-mail IN A 18.1.6.40
pmaldonado IN MX 10 pto-mail
pto-mail IN A 18.1.7.40
```

Todo esto permite que los mensajes destinados con `@incakoca.org` lleguen al servidor de correo local que les corresponde.

Los problemas pendientes son:

Pérdida de independencia

Cada administrador local debe notificar al administrador del switch de que se ha creado un usuario local a fin de que se le registre en `virtusertable`. Se pierde independencia y no hay una forma sencilla de evitar esto.

Ineficiencia en escritura de direcciones

Cuando un mensaje se envía desde el interior de la organización a otro usuario de la organización, es necesario especificar la dirección de email completa (`usuario@incakoca.org`). Sería deseable usar sólo el nombre del usuario y que el sistema asuma que se trata de la organización.

Ineficiencia en los envíos locales

Si se envía un mensaje desde Iquitos a otro usuario *en Iquitos* usando su dirección email "`usuario@incakoca.org`", el mensaje (de acuerdo al DNS) irá primero al switch de la organización y luego ¡retornará! a Iquitos. Esto es correcto pero ineficiente.

Sería deseable que si el usuario destinatario es local al remitente, entonces el mensaje no tenga que viajar hasta el switch.

* PENDIENTE.

Performance/Tuning

Balance con varios servidores

Normalmente los sitios que reciben una gran cantidad de email tienen una configuración tipo "switch" como la que se comentó anteriormente en la sección "Organización con divisiones administrativas". En ese caso puede ser conveniente configurar varios computadores para que reciban la carga de mensajes entrantes y la redirijan a los computadores departamentales.

A fin de lograr esto, se suele aplicar dos técnicas, a saber, vía DNS o mediante NAT. En el DNS es posible especificar varios servidores de switch para la organización con la misma preferencia. Siguiendo el ejemplo de la sección señalada, habría que modificar esto:

```
; archivo de zona de inkacoca.org
@ IN MX 10 switch
switch IN A 18.1.3.41
lima IN MX 10 lima-mail
lima-mail IN A 18.1.3.40
...
```

Por lo siguiente:

```
; archivo de zona de inkacoca.org
@ IN MX 10 switch
switch IN A 18.1.3.41
switch IN A 18.1.3.42
switch IN A 18.1.3.43
lima IN MX 10 lima-mail
lima-mail IN A 18.1.3.40
...
```

Esto supone disponer de tres mailservers llamados "switch" asociados a las direcciones IP 18.1.3.4[123]. Bajo condiciones normales, esto balanceará la carga entre los tres servidores. Para más información, revise la referencia [2] en lo referente a "Round Robin Load Distribution".

El método NAT consistiría en nuestro ejemplo en redirigir las conexiones destinadas a la dirección 18.1.3.41 (no habría ningún cambio en el DNS) hacia un grupo de otras direcciones (DNAT) que realmente están asociadas a los servidores de email. Los ruteadores normalmente disponen de estas facilidades (y por supuesto Linux cuando actúa como router, usando el comando `iptables`.) No lo explicaremos aquí.

Intervalo de Queue Run

En sitios pequeños el intervalo debe ser corto a fin de asegurar el rápido envío (por ejemplo, 15 minutos.) En la mayoría de sitios, sin embargo, una hora es un tiempo aceptable.

Como se indicó, el intervalo de Queue Run se especifica mediante opciones al ejecutarse sendmail (opción -q).

Procesos de ejecución en Queue Run

Cuando el intervalo de Queue Run es corto, cuando los mensajes encolados son muchos, y cuando los servidores remotos son lentos, Sendmail puede lanzar muchos procesos "queue runners" concurrentes que intentan en paralelo limpiar la cola. Esto normalmente debe autocontrolarse, pero en situaciones extremas el número de "queue runners" puede ser muy elevado al punto de crear un peligro para el sistema.

La opción `MaxQueueChildren` permite especificar el máximo número de procesos "hijos" de Sendmail encargados de procesar la cola simultáneamente. Normalmente no hay límite.

El valor más adecuado se debe obtener observando el comportamiento "usual" de los queue runners de Sendmail, por ejemplo, mediante un comando como el que se muestra:

```
[root@edithpiaf root]# ps ax | grep sendmail | grep queue | wc -l
7
```

Esto se debe observar a distintas horas. Un valor razonable puede ser el máximo observado durante la semana multiplicado por dos.

Por otro lado, si el sistema constantemente tiene problemas de falta de memoria, y se sospecha que el problema está asociado a un número elevado de "queue runners", puede ser conveniente reducir esta opción gradualmente (quizá reduciendo 10% al valor observado) a fin de aliviar este problema. Esto ocasionará que algunos mensajes encolados tarden más en procesarse.

Limpiar más aprisa los mensajes en cola

La opción "Timeout.queuwarn" determina el lapso que permanece un mensaje en cola para generar un mensaje de alerta al redactor del mismo. Esta alerta le permite saber que el mensaje no pudo ser enviado (default=4 horas.)

La opción "Timeout.queuereturn" determina el tiempo máximo que el sistema mantendrá el mensaje en la cola (intentando enviarlo.) Cuando este tiempo se cumple, el mensaje es eliminado de la cola y se envía una alerta al redactor (default=5 días.)

Por consistencia, Timeout.queuwarn < Timeout.queuereturn.

A fin de que los mensajes "reboten" más aprisa (y la cola se libere más aprisa), es conveniente en ciertos casos reducir estos valores, por ejemplo:

En el archivo "cf":

```
O Timeout.queuwarn=2h
O Timeout.queuereturn=1d
```

En M4:

```
define(`confTO_QUEUEWARN',`2h')
define(`confTO_QUEUERETURN',`1d')
```

Cola con varios directorios

El acceso a disco generado por las colas puede ser considerable en sistemas grandes. Una forma de balancear esta carga y mejorar la performance consiste en utilizar discos físicos distintos, los cuales pueden montarse y usarse simultáneamente para la cola.

Esto se puede conseguir creando varios subdirectorios a partir de un directorio principal (por ejemplo bajo /var/spool/mqueue) y configurándolos en el archivo "cf" (o vía M4.)

```
[root@edithpiaf mqueue]# ls -ld /var/spool/mqueue/
drwx----- 2 root mail 4096 oct 5 13:39 /var/spool/mqueue/
[root@edithpiaf mqueue]# cd /var/spool/mqueue
[root@edithpiaf mqueue]# mkdir cola1
[root@edithpiaf mqueue]# mkdir cola2
[root@edithpiaf mqueue]# mkdir cola3
[root@edithpiaf mqueue]# chmod 700 cola*
[root@edithpiaf mqueue]# mount /dev/sdX cola1
[root@edithpiaf mqueue]# mount /dev/sdY cola2
```

```
[root@edithpiaf mqueue]# mount /dev/sdZ cola3
[root@edithpiaf mqueue]# chown root.mail cola*
[root@edithpiaf mqueue]# ls -l
total 12
drwx----- 2 root mail 4096 oct 11 13:01 cola1
drwx----- 2 root mail 4096 oct 11 13:01 cola2
drwx----- 2 root mail 4096 oct 11 13:01 cola3
```

En el archivo "cf" se colocaría:

```
QueueDirectory=/var/spool/mqueue/cola*
```

Por el método M4:

```
define(`QUEUE_DIR',`/var/spool/mqueue/cola*')
```

Tenga cuidado de no especificar `"/var/spool/mqueue/*"` pues esto haría que Sendmail incluya (incorrectamente) los directorios `"/var/spool/mqueue/."` y `"/var/spool/mqueue/.."`.

Encolar si hay mucha carga

Si la carga del sistema se hace muy elevada, es conveniente limitar la operación de Sendmail, al menos para los mensajes de menor prioridad.

Cuando un nuevo mensaje es recibido, Sendmail le calcula su prioridad inicial "PRI" con la fórmula señalada anteriormente.

Si la carga es muy elevada, Sendmail tiene la opción de no intentar enviar el mensaje, sino, simplemente encolarlo para un intento posterior. Esto ocurrirá si se verifican las siguientes condiciones:

1) La carga promedio del sistema (load average) es mayor que el indicado en la opción configurable "QueueLA" (cuyo default es ocho.) La carga promedio es un valor proporcionado por el sistema operativo que se puede obtener con el comando "uptime".

2) Se verifica la ecuación:

$$PRI > QueueFactor / (LA - QueueLA + 1)$$

Recordar que un valor "PRI" más elevado es menor prioridad y por tanto hace más probable que la ecuación anterior se verifique.

QueueFactor es una opción configurable cuyo valor por omisión es 600000.

En sistemas grandes puede ser necesario elevar el valor de QueueLA:

```
O QueueLA=15
```

En M4:

```
define(`confQUEUE_LA',`15')
```

Rechazar si hay mucha carga

La opción "RefuseLA" permite especificar un límite para la carga promedio del sistema (load average) a partir del cual Sendmail simplemente rechaza los mensajes que se le envían. Esto, evidentemente asume que quien se está intentando conectar lo intentará nuevamente más adelante.

El valor por omisión de esta opción es doce. En sistemas grandes puede ser necesario elevar este valor.

```
O RefuseLA=18
```

En M4:

```
define('confREFUSE_LA', '18')
```

Procesar lentamente las conexiones

La opción ConnectionRateThrottle permite "relentizar" las conexiones que llegan a un ritmo muy veloz. Por ejemplo, si este parámetro es igual a cinco, en caso de llegar más de cinco conexiones en menos de un segundo, sólo las cinco son atendidas inmediatamente. Otras cinco serán atendidas luego de un segundo, y así sucesivamente.

Esta opción es adecuada cuando se usan las opciones QueueLA y RefuseLA.

```
O ConnectionRateThrottle=5
```

En M4:

```
define('confCONNECTION_RATE_THROTTLE', '5')
```

Reglas y Rulesets

En una primera lectura de este tutorial sugiero evitar todo este material puesto que no es esencial para la gran mayoría de administradores de Sendmail.

Las reglas son especificaciones de configuración que sirven para modificar las direcciones electrónicas y detectar errores en las mismas. Hay diversos motivos por los que una dirección electrónica debe ser modificada; por ejemplo, si se desea que todos los mensajes luzcan como si hubieran sido enviados desde cierto computador aunque en realidad se han enviados desde diversos computadores de distinto nombre.

Los conjuntos de reglas se agrupan en los llamados "Rulesets" que funcionan a modo de "subrutina" de cualquier lenguaje de programación. Los "Rulesets" se identifican con un número, aunque en las últimas versiones de Sendmail es posible identificarlos con una palabra (que internamente es traducida a un número por Sendmail.)

Algunos rulesets son definidos internamente (como los rulesets 0, 1, 2, 3, 4, y 5) mientras que otros se definen manualmente en el archivo "cf".

Los rulesets se definen mediante el comando "S" y las reglas mediante el comando "R". A continuación un extracto del archivo "cf" que viene con RedHat 8.0 en el que se ilustra el ruleset "0" o también denominado "parse". Obsérvese que el ruleset termina donde empieza uno nuevo:

```
#####  
### Ruleset 0 -- Parse Address ###  
#####  
  
Sparse=0  
  
R$*                $: $>Parse0 $1                initial parsing  
R<@>              $#local $: <@>                special case error msgs  
R$*                $: $>ParseLocal $1           handle local hacks
```

```
R$*                $: $>Parse1 $1                final parsing
```

Esta salida puede ser muy distinta en otros sistemas Unix/Linux, pero por el momento eso no importa.

El ruleset 0 se pudo definir mediante "S0" en lugar de "Sparse", pero como "parse" es más "comprensible" que "0", entonces se prefirió esta última forma (Sparse=0.)

Para verificar cómo ha cargado Sendmail las reglas, se puede usar el modo de test de Sendmail con el comando "=S" y el número del ruleset:

```
# sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> =S 0
R$*                $: $> Parse0 $1
R< @ >            $# local $: < @ >
R$*                $: $> ParseLocal $1
R$*                $: $> Parse1 $1
```

Nótese que esto coincide con la definición del archivo de configuración. De igual modo puede consultarse con el nombre del ruleset:

```
# sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> =S parse
R$*                $: $> Parse0 $1
R< @ >            $# local $: < @ >
R$*                $: $> ParseLocal $1
R$*                $: $> Parse1 $1
```

Como es fácil de apreciar, cada ruleset consiste de varias "reglas" definidas con el comando "R".

Cada regla consiste de dos partes separadas por un tabulador, posiblemente seguidas de un comentario (separado también por un tabulador.)

Ejemplo de ruleset con una regla

Para esta sección haremos algunas modificaciones al archivo de configuración "cf". Sin embargo, a fin de evitar alterar la configuración actual, trabajaremos sobre un archivo "cf" alternativo. Empezamos haciendo una copia del archivo "cf":

```
# cp /etc/mail/sendmail.cf prueba.cf
```

Ahora podremos hacer nuestros cambios en "prueba.cf". Añadiremos a éste último las siguientes líneas en la parte final:

```
D{MAILHUB}mail.peru.com.pe
Sprueba
R$+@$+ $1*${MAILHUB}          convierte user@host
```

Como se ve, hemos definido la macro "MAILHUB", el ruleset "prueba" y una única regla. Para que la regla esté adecuadamente definida, es imprescindible que sus tres partes estén separadas por al menos un tabulador (no espacios):

```
R$+@$+ <TAB> $1*${MAILHUB} <TAB> convierte user@host
```

Para probar nuestra regla, ejecutaremos Sendmail en modo de prueba pero especificando el nuevo archivo "cf":

```
# sendmail -Cprueba.cf -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> prueba diego@hotmail.com
prueba          input: diego @ hotmail . com
prueba          returns: diego *mail . peru . com . pe
#
```

Como se aprecia, el modo de prueba permite especificar el nombre de cualquier rule-set seguido por una dirección electrónica a procesar.

Partes de una regla

Cada regla tiene hasta tres partes (el comentario es opcional) separadas por tabuladores (puede haber espacios dentro de cada parte.)

Recuérdese que las reglas alteran las direcciones de correo electrónico.

La primera parte (que sigue inmediatamente al comando "R") es la "left hand side" (lado izquierdo o LHS) mientras que la segunda es la "right hand side" (lado derecho o RHS.) La LHS especifica un patrón de búsqueda. De haber coincidencia con el patrón de búsqueda, la dirección de correo electrónico que se está procesando es convertida en el RHS.

Tokens

A fin de comprender las reglas, es necesario conocer el concepto de "token".

Las direcciones electrónicas procesadas en los rulesets por las reglas son internamente fraccionadas en varias unidades independientes denominadas "token". Así, la dirección proporcionada en el ejemplo de arriba: "diego@hotmail.com" es internamente dividida en los siguientes cinco tokens:

```
diego
@
hotmail
.
com
```

Los siguientes caracteres normalmente delimitan los tokens (y actúan ellos mismos como tokens adicionales):

```
.:@[ ]
()<>,;\\"x\n
```

Expresiones de búsqueda

En el ejemplo de arriba, la LHS está conformada por dos "expresión de búsqueda" (comodines o wildcards) de tipo "\$+". Estos sirven para buscar coincidencias de uno o más "tokens". Más adelante veremos más de estos comodines.

En nuestro ejemplo, la dirección proporcionada hizo coincidir el primer "\$+" con el token "diego" y el segundo "\$+" con los tokens "hotmail", ".", "com".

Por el lado de la RHS, el contenido de cada "expresión de búsqueda" es accesible mediante los operadores \$1, \$2, ... respectivamente. Para nuestro caso, tendremos:

```
$1 = diego
$2 = hotmail . com
```

En nuestro ejemplo, el RHS es "\$1*\${MAILHUB}", lo que equivale a "diego*mail.peru.com.pe", cosa que se aprecia en la respuesta.

El modificador de debug 21.12 permite obtener más información acerca del procesamiento de cada regla.

```
# sendmail -d21.12 -bt -Cprueba.cf
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> prueba diego@hotmail.com
prueba          input: diego @ hotmail . com
-----trying rule: $+ @ $+
-----rule matches: $1 *mail . peru . com . pe
rewritten as: diego *mail . peru . com . pe
-----trying rule: $+ @ $+
----- rule fails
prueba          returns: diego *mail . peru . com . pe
```

Rulesets Internos

Los rulesets 0, 1, 2, 3, 4, 5 están reservados para usos específicos de Sendmail. A futuro Sendmail puede definir los rulesets 6, 7, 8, 9. En general, es conveniente que el usuario defina - si lo requiere - rulesets con textos identificatorios (para que Sendmail automáticamente los numere) con lo que se evitan conflictos innecesarios.

Es conveniente conocer el uso que da Sendmail a los rulesets internos. La siguiente figura intenta ilustrar este punto.

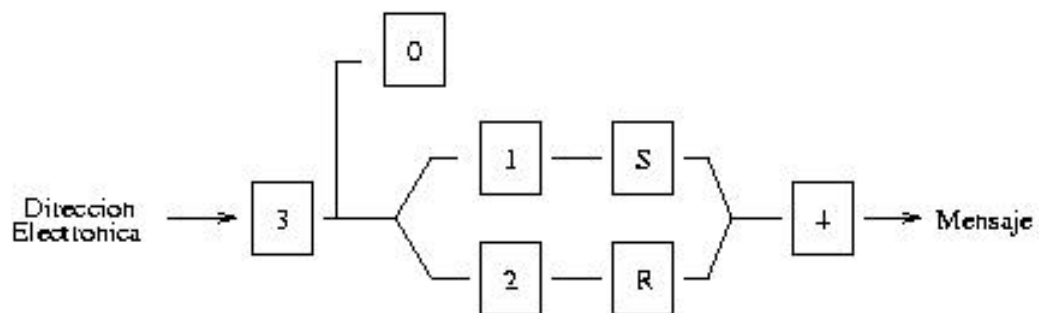


Figura 1.

En general, todas las direcciones electrónicas pasan por el ruleset 3 apenas se inicia el procesamiento de las mismas. Entre otras cosas, el ruleset 3 extrae la dirección electrónica "apta para procesamiento" a partir de la dirección electrónica entregada por los clientes.

Por ejemplo, es usual que las direcciones electrónicas luzcan así (tal como las genera el programa cliente):

```
Pedro Escobedo <pescobedo@noskhon.com.pe>
```

Obviamente, para efectos de la transferencia del mensaje, el nombre real no es importante. Probemos esto:

```
# sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> 3 Pedro Escobedo <pescobedo@noskhon.com.pe>
canonify          input: Pedro Escobedo < pescobedo @ noskhon . com . pe >
Canonify2         input: pescobedo < @ noskhon . com . pe >
Canonify2         returns: pescobedo < @ noskhon . com . pe . >
canonify          returns: pescobedo < @ noskhon . com . pe . >
```

Lo que hemos hecho es transformar la dirección electrónica mediante el ruleset 3, el cual se encarga de "preprocesar" todas las direcciones electrónicas. Nótese que la respuesta consiste de la dirección electrónica en un formato especial usado internamente.

Es sencillo imprimir el ruleset 3 (o cualquier otro) desde Sendmail. Para esto, simplemente entrar en modo debug (con -bt) y tipear el comando "=S":

```
> =S 3
R$@          $@ < @ >
R$*          $: $1 < @ >
R$* < $* > $* < @ >          $: $1 < $2 > $3
...
```

Finalmente, todavía en modo test, pruébese el comando "/parse" que permite simular el procesamiento de una dirección de correo electrónico (por ejemplo, pruebe "/parse user@localhost").

Macros en el archivo "cf"

Para explicar esto, copiaré una parte del archivo "cf" que se vió anteriormente (en la sección del delivery local.)

```
Mlocal, P=/usr/bin/procmail, F=lsDFMAw5:/|@qSPfhn9,
S=EnvFromL/HdrFromL,
R=EnvToL/HdrToL,
T=DNS/RFC822/X-Unix,
A=procmail -t -Y -a $h -d $u
```

Como se aprecia, aquí se emplearon las construcciones "\$h" y "\$n". Como ya sospechará el lector, esto corresponde a la expansión de dos "macros".

En particular, la macro "h" (cuyo valor se obtiene con "\$h") corresponde al host destinatario del mensaje, mientras que la macro "n" corresponde al usuario destinatario del mensaje. En este caso el valor de la macro es ajustado por Sendmail automáticamente para cada mensaje.

Las macros se pueden definir con el comando "D" en cualquier parte del archivo "cf". Por ejemplo, esto redefine la macro "w" al valor "jibaros":

```
Dwjibaros
```

Las macros cuyos nombres tienen más de un carácter deben usar llaves en su definición y su expansión:

```
D{PRUEBA}amazonas.com.pe
... ${PRUEBA} ...
```

La opción de debug -d35.9 permite obtener un extenso listado de macros definidas por Sendmail y vía el archivo "cf".

Ciertas macros son asignadas internamente por el programa Sendmail (como la macro "w" que es inicializada al nombre del host "sin dominio") y otras son creadas explícitamente en el archivo "cf" con diversos propósitos.

Clases

Las "clases" son una suerte de variables tipo "array", es decir, que pueden contener un conjunto de valores.

Las clases no se solapan con las macros. Por ejemplo, existe la clase "w" que no tiene relación alguna con la macro "w". Los elementos de la clase se añaden con el comando "C". Por ejemplo, los siguientes dos comandos añaden los elementos "localhost" y "localhost.localdomain" a la clase "w" del archivo "cf":

```
[root@edithpiaf tmp]# grep '^Cw' /etc/mail/sendmail.cf
Cwlocalhost.localdomain
Cwlocalhost
```

Tal como se mencionó anteriormente, la clase "w" corresponde a los "nombres de host" que Sendmail considera "locales" y que permiten que un mensaje sea dirigido al mailbox en lugar de ser redirigido hacia otro destino.

Como se recordará, esa configuración la hacíamos mediante el archivo `/etc/mail/local-host-names`. En otras palabras, las líneas de este archivo pasan a conformar la clase "w". Esto se define en el archivo "cf" mediante el comando "F":

```
[root@edithpiaf tmp]# grep '/etc/mail/local-host-names' /etc/mail/sendmail.cf
Fw/etc/mail/local-host-names
```

Evidentemente, esto también se pudo configurar añadiendo directamente directivas "Cw..." al archivo "cf", pero es un método menos flexible.

La clase w desde M4

Como se ve, esta clase es especial. A tal efecto si usamos el método M4 y se desea generar un archivo "cf" que incluya la configuración via el archivo `local-host-names` deberá usarse:

```
FEATURE(use_cw_file)dnl
```

A. Ejemplo: Ambiente de alta seguridad

Descripción

Supondremos que nuestra organización se desempeña tras un Firewall. Podríamos hacer que el esquema anterior siga sin variaciones simplemente habilitando los puertos respectivos en el filtro de paquetes. Sin embargo, aquí consideraremos una situación más complicada en la que se requiere de dos servidores de email. Asumimos que el lector conoce el concepto de una red DMZ (zona desmilitarizada) usada como primera línea de defensa para los servicios que deben dar cara al exterior.

Uno de estos servidores de email se encontrará en la red DMZ (`correo_dmz`) y será accesible desde Internet (aunque pasando por el Firewall), y el otro se encuentra dentro de nuestra red LAN (`correo_lan`). Las estaciones seguirán configuradas

para conectarse a este último (`correo_lan`) por lo que no haremos cambios al respecto. Sin embargo, `correo_lan` NO enviará mensajes al exterior directamente, sino a través de `correo_dmz`, y viceversa, no recibirá mensajes desde el exterior, salvo desde `correo_dmz`. La siguiente figura esquematiza esta configuración:

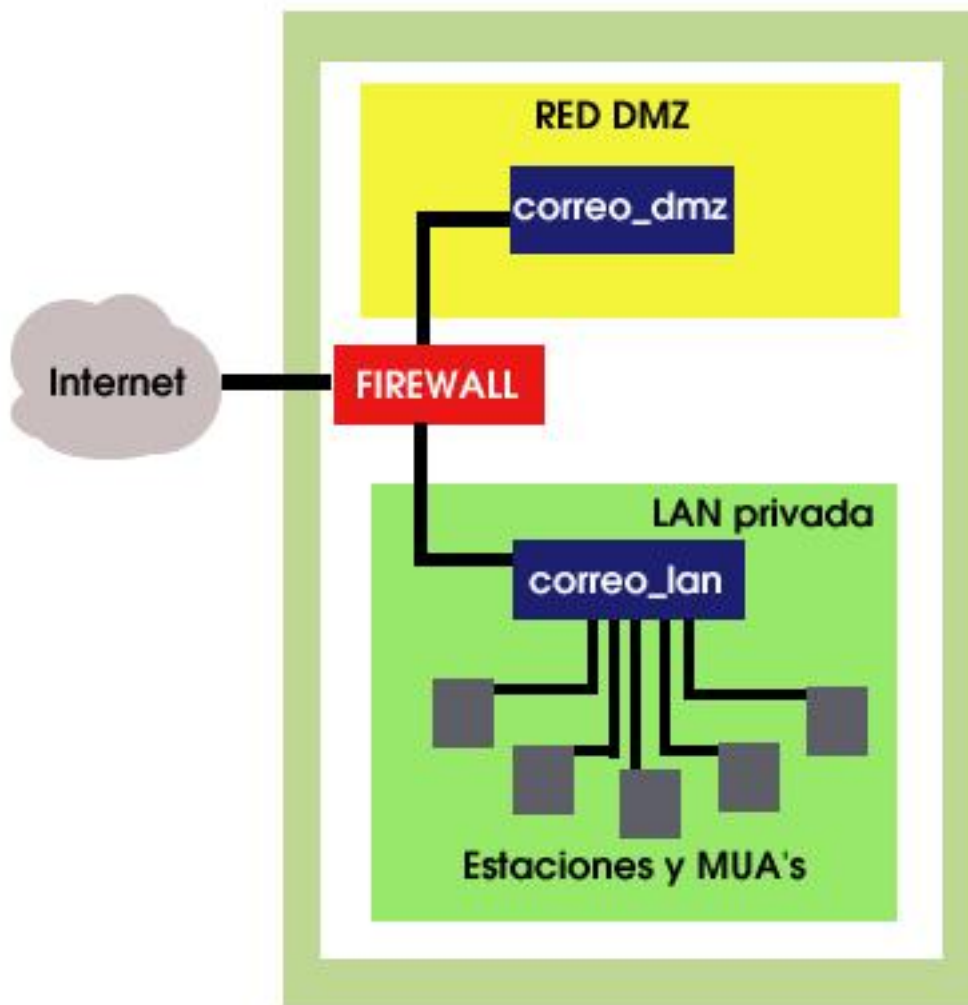


Figura A-1. Solución con red DMZ

Nótese que nosotros no indicaremos absolutamente nada acerca de la configuración del firewall pues esto está fuera de los objetivos de esta guía. Si se implementa el firewall con Linux, en Internet ya hay abundantes documentos acerca de cómo configurarlo.

Configuración de `correo_dmz`

Este servidor no almacenará los mensajes pendientes en casillas, por lo que no requiere que se inscriban allí nuestros usuarios. A fin de que los mensajes

que llegan al dominio "abejas.org" se redirijan a correo_lan, en el archivo /etc/mail/mailertable debemos inscribir la siguiente línea:

```
abejas.org      smtp:correo_lan.abejas.org
```

Ud. ahora debe generar la versión compilada con:

```
bash# cd /etc/mail
bash# make
```

Los detalles que vimos para el archivo access son aplicables a este archivo. En particular, para activar esta funcionalidad desde el archivo "mc", se debe incluir:

```
FEATURE(`mailertable', `hash -o /etc/mail/mailertable')dnl
```

Por otro lado, "correo_dmz" sirve de relay a "correo_lan", por lo que debemos inscribir a este último en /etc/mail/access como se vió en una sección anterior:

```
correo_lan.abejas.org      RELAY
```

Configuración del DNS

Desde Internet, los mensajes deben llegar a "correo_dmz", por lo cual éste servidor deberá ser el destino especificado en el registro MX para nuestro dominio:

```
$TTL 86400
@      IN      SOA      @  root.localhost (
                        4 ; serial
                        28800 ; refresh
                        7200 ; retry
                        604800 ; expire
                        86400 ; ttl
                        )
      IN      NS       ns1
      IN      MX       0  correo_dmz

correo_dmz IN      A       100.4.244.191
correo_lan  IN      A       100.4.244.195
ns1         IN      A       100.4.244.193
```

Configuración de correo_lan

Tal como vimos en una sección anterior, se debe modificar el archivo /etc/mail/local-host-names (o su equivalente) para que se acepten los mensajes con el formato deseado.

Ahora haremos que se envíe todo el correo no local hacia el computador "correo_dmz" en lugar del Internet. Para esto, debemos regenerar el archivo "cf" a partir del "mc", añadiendo previamente la siguiente directiva a éste último:

```
define(`SMART_HOST', `smtp:correo_dmz.abejas.org')dnl
```

Eso debería bastar. Revise los logs, verifique que el firewall no bloquee más de lo necesario.

B. Ejemplo de configuración de MUA

Los clientes de nuestra red local accederán al servidor mediante programas especializados para la redacción y visualización de mensajes. Estos programas se conocen como MUA's (Mail User Agent.) Este es un asunto que estrictamente no tiene que ver con nuestra guía, pero creemos conveniente proporcionar algún ejemplo ilustrativo. Supondremos que los usuarios están utilizando Netscape Messenger, que es parte del Netscape Communicator. A continuación mostramos la pantalla principal de Netscape Messenger:



Figura B-1. Netscape Messenger

Vamos a configurar el MUA para que acceda a nuestro servidor de correo. Asumimos que ya están activos los servicios POP o IMAP. Cuál de estos se usará para traer los mensajes, y de qué usuario, se selecciona en el menú Edit -> Preferences -> Mail Servers -> Incoming Mail Servers -> Edit o Add -> Server Name y Username , como se aprecia en la figura:

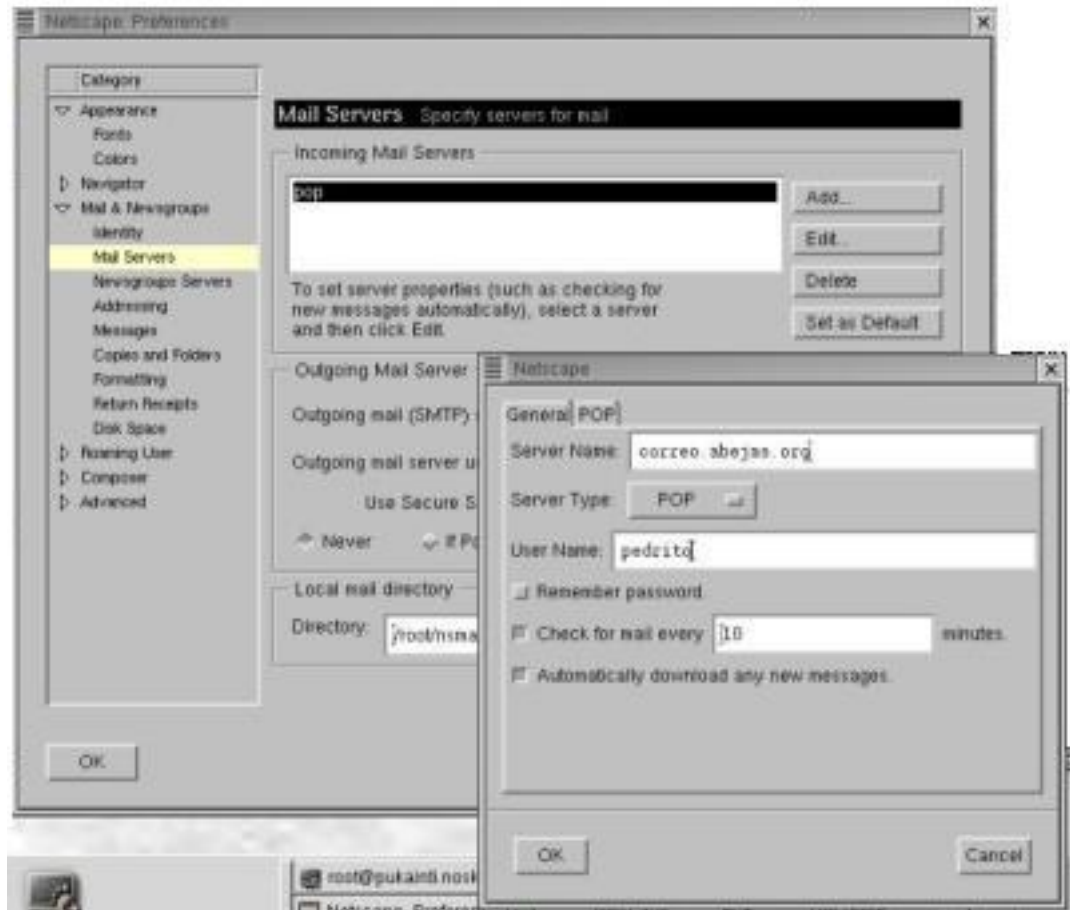


Figura B-2. Configuración de servidor POP

Asimismo, en Edit -> Preferences -> Mail Servers -> Outgoing (SMTP) server, se debe especificar nuevamente nuestro servidor de correo, puesto que hacia allí se enviarán los mensajes que redactemos.

Finalmente, en Edit -> Preferences -> Identity, se debe especificar nuestra dirección de email y nuestro nombre real, tal como lo verán los destinatarios:

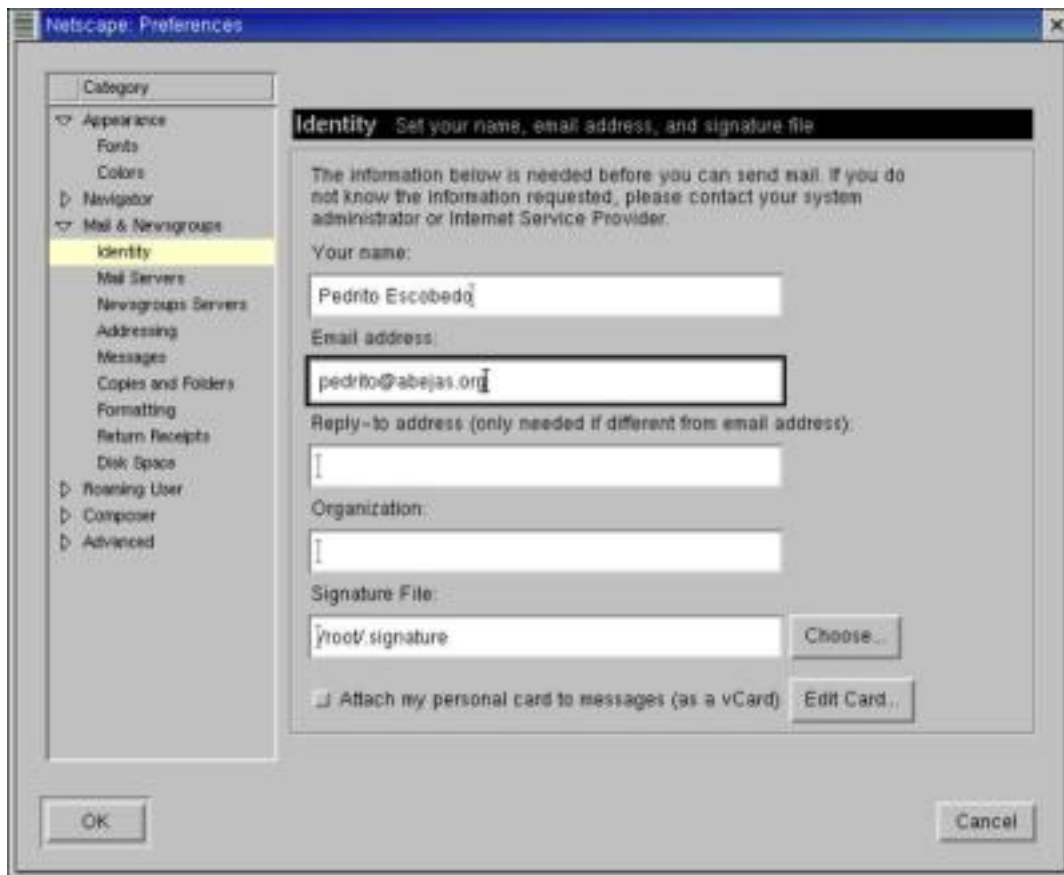


Figura B-3. Configuración de servidor POP

Ahora Ud. debería poder enviar mensajes a usuarios locales (de la forma user@abejas.org), y recibir mensajes desde cualquier lugar de Internet. Pruebe a crear algunos usuarios en el servidor y a hacer que se intercambien mensajes desde sus estaciones.

C. Referencias

[1] Bryan Costales & Erick Allman: Sendmail. 2th Edition. O'Reilly. USA. ISBN 1-56592-222-0

[2] Paul Albitz & Cricket Liu: DNS and BIND. 4th Edition. O'Reilly. USA. ISBN 0-596-00158-4. 2001

Notas

1. <http://qdk.sourceforge.net/>