Servicios avanzados V: Servidor de correo (SMTP) y protocolos de entrega final.

Autor: Enrique V. Bonet Esteban

Introducción.

Los primeros sistemas de correo electrónico simplemente consistían en protocolos de transferencia de archivos, generalmente protocolos como UUCP o FTP[1], con la convención de que la primera línea de cada mensaje, es decir del archivo, contenía la dirección del destinatario.

A medida que se acumuló experiencia, se observó las carencias de este método de funcionamiento (dificultad en enviar el mensaje a más de un destinatario, etc.), y se propusieron sistemas de correo electrónico más elaborados. De esta forma, en 1982 se publicaron las propuestas de correo electrónico del ARPANET como RFC 821 (protocolo de transmisión) y RFC 822 (formato de mensaje), consistiendo, en esta época, un sistema de correo electrónico en dos subsistemas distintos:

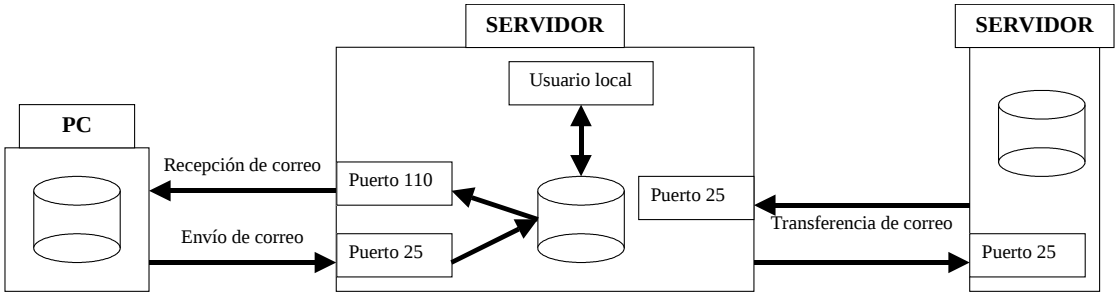
- Un agente de transferencia de mensaje, encargado de transmitir los mensajes del remitente al destinatario. En Internet, el correo electrónico se transfiere de la computadora origen a la computadora destino, mediante el establecimiento de una conexión TCP desde la computadora origen al puerto 25 de la computadora destino. Para ello se utiliza el protocolo SMTP (Simple Mail Transfer Protocol).

- Un agente de usuario, que permite leer y enviar el correo electrónico. Los agentes de usuario son programas locales que proporcionan un método basado en comandos, basado en menús o basado en la interacción gráfica, para comunicarse con el sistema de correo electrónico.

En la actualidad, y con la popularización de los ordenadores personales, los usuarios suelen enviar y/o recibir el correo en su propio ordenador personal, de forma que se han desarrollado protocolos de entrega final de correo, los cuales permiten que los usuarios descarguen el correo a su ordenador personal sin necesidad de utilizar un agente de usuario en el propio servidor de correo. De los protocolos de entrega final existentes los más conocidos son POP3 e IMAP.

Un sencillo esquema con el funcionamiento actual del correo electrónico puede verse en la figura siguiente.

[1] Tanto UUCP como FTP son protocolos de transferencia de archivos entre ordenadores cuya misión es transferir archivos de cualquier tipo entre ordenadores en Internet. Actualmente el más conocido y utilizado de los dos es FTP.



Configuración del servidor de SMTP.

El servidor de Simple Mail Transfer Protocol (protocolo sencillo de transferencia de correo) es el encargado de transferir el correo entre los ordenadores de Internet desde el ordenador origen al ordenador destino. SMTP es un protocolo cliente/servidor en formato ASCII que, como hemos dicho, utiliza una conexión TCP al puerto 25 del servidor para el envío del correo.

Los comandos de SMTP fueron definidos en el RFC 821, y modificados en el RFC 1425, donde se definió el ESMTP (Extended Simple Mail Transfer Protocol)[2].

Por su parte, el formato de los mensajes se definió en el RFC 822, y fue extendido en el RFC 1521, en el cual se introdujo la definición de los formatos MIME (Multipurpose Internet Mail Protocol)[3].

Actualmente se encuentran disponibles distintos servidores de SMTP, pero nosotros nos centraremos en la configuración del servidor *sendmail*[4], el cual se encuentra en el directorio */usr/sbin*, pudiendo funcionar en modo de "demonio", esto es, puede permanecer en estado de espera escuchando el puerto TCP 25 o bien puede ser ejecutado por *xinetd*[5]. El funcionamiento por defecto es que permanezca escuchando el puerto TCP 25.

Los ficheros de configuración de *sendmail* se encuentran dentro del directorio */etc/mail*. En dicho directorio pueden encontrarse, básicamente, dos tipos de ficheros en función de su extensión, ficheros de extensión *.cf* y ficheros de extensión *.db*. Ambos tipos de ficheros contienen las instrucciones básicas que configuran el funcionamiento de *sendmail*. Sin embargo, y dada la dificultad de sintaxis de esos ficheros, existen otros ficheros, de sintaxis más sencilla, que son los que realmente se configuran, siendo convertidos a los ficheros anteriores mediante utilidades del sistema[6].

En concreto, los ficheros de extensión *.cf* contienen las instrucciones básicas de funcionamiento de *sendmail*, y son creados a partir de los ficheros de extensión *.mc* que se encuentran en el mismo directorio. La conversión se realiza mediante el programa *m4*, que se encuentra en */usr/bin*. La sintaxis de ejecución de dicho comando es:

[2] Una explicación de los comandos de SMTP y ESMTP pueden verse en el apéndice A.
[3] Una explicación de los formatos de mensajes RFC 822 y RFC 1521 se encuentra en el apéndice B.
[4] El servidor sendmail es el más extendido de todos y se encuentra disponible en la mayoría de sistemas operativos.
[5] Veremos el servidor xinetd en temas posteriores.
[6] Por ejemplo, el fichero sendmail.mc que instala inicialmente el sistema tiene una longitud de 178 líneas, mientras que su conversión en sendmail.cf, tiene una longitud de 1840 líneas.

```
/usr/bin/m4 /etc/mail/fichero.mc > /etc/mail/fichero.cf
```

Donde puede observarse que la salida del programa debe ser redirigida a un fichero pues, por defecto, la salida del programa se dirige a la salida estándar, generalmente la pantalla.

Por otra parte, los ficheros de extensión *.db* especifican permisos de acceso, etc., al servidor de *sendmail*, y son creados a partir de ficheros sin extensión, pero que poseen el mismo nombre que el fichero de extensión *.db*. La creación se realiza mediante el comando *makemap*, que se encuentra en */usr/sbin*. El comando *makemap* convierte un fichero de texto en un fichero de base de datos[7] que pueda ser leído por *sendmail*. La conversión se realiza con el comando:

```
/usr/bin/makemap hash /etc/mail/fichero </etc/mail/fichero
```

Donde puede observarse que la entrada al programa debe ser especificada desde el fichero de texto, pues por defecto, *makemap* lee de la entrada estándar, generalmente el teclado.

Configuración de los ficheros .mc.

En el directorio */etc/mail* existen dos ficheros con extensión *.mc*, los ficheros *submit.mc* y *sendmail.mc*. La existencia de dos ficheros de configuración surge a partir de la versión 8.12 de *sendmail* y es introducida por motivos de seguridad. A partir de esa versión de *sendmail*, el programa se ejecuta como dos agentes distintos, el MTA (Mail Transport Agent) encargado de transferir el correo entre diferentes ordenadores, y el MSA (Mail Submission Agent) encargado de recibir los correos de los usuarios y escribirlos en la cola de correos a enviar.

La separación en dos agentes distintos es consecuencia de que el MTA se ejecuta como un usuario adecuado a su labor, generalmente el usuario root, mientras que su uso como MSA requiere que cualquier usuario pueda escribir en ciertos directorios del sistema, como puede ser el directorio con la cola de correos a enviar, lo que requiere dar permisos para que todo el mundo pueda escribir en dicho directorio, o bien, activar como mínimo el bit de GID para que un usuario, al ejecutar *sendmail* en modo de MSA, pueda escribir en ese directorio.

Los ficheros *.mc* poseen las siguientes palabras clave[8]:

Palabra	Descripción
dnl	Comienzo de comentario. Todo lo que siga en la línea se considera un comentario y no será tenido en cuenta.

[7] Existen tres formatos de la base de datos, dbm, btree y hash, siendo por defecto el formato dbm el utilizado.

[8] Una explicación más detallada de estas y otras palabras claves existentes, así como los valores que pueden tomar, puede encontrarse en el fichero /usr/share/sendmail-cf/README o en la URL http://www.sendmail.org.

Palabra	Descripción
divert(n)	Las líneas que siguen deben ser ignoradas en la salida (n=-1) o incluidas en la salida (n>=0). De forma general para incluir las líneas n será 0, aunque puede tomar valores hasta 9 para indicar su inclusión según ciertas condiciones y/o orden.
include	Incluye el contenido del fichero indicado.
sinclude	Incluye el contenido del fichero indicado.
define	Permite definir el valor de una macro de configuración con el valor indicado.
VERSIONID	Incluye como información el mensaje escrito.
OSTYPE	Define el sistema operativo sobre el que ejecuta sendmail.
MAILER	Tipos de correo que son aceptados.
DOMAIN	Define los ordenadores que aceptarán cada tipo de correo.
FEATURE	Especifica opciones particulares de configuración de sendmail.

El fichero submit.mc.

El fichero *submit.mc* que se encuentra instalado por defecto es el siguiente:

```
divert(-1)
#
# Copyright (c) 2001-2003 Sendmail, Inc. and its suppliers.
#       All rights reserved.
#
# By using this file, you agree to the terms and conditions set
# forth in the LICENSE file which can be found at the top level of
# the sendmail distribution.
#
#


#
#  This is the prototype file for a set-group-ID sm-msp sendmail that
#  acts as a initial mail submission program.
#

divert(0)dnl
sinclude(`/usr/share/sendmail-cf/m4/cf.m4')
VERSIONID(`linux setup')dnl
define(`confCF_VERSION', `Submit')dnl
define(`__OSTYPE__',`')dnl dirty hack to keep proto.m4 from complaining
define(`_USE_DECNET_SYNTAX_', `1')dnl support DECnet
define(`confTIME_ZONE', `USE_TZ')dnl
define(`confDONT_INIT_GROUPS', `True')dnl
dnl # If you're operating in a DSCP/RFC-4594 environment with QoS
dnl define(`confINET_QOS', `AF11')dnl
define(`confPID_FILE', `/run/sm-client.pid')dnl
dnl define(`confDIRECT_SUBMISSION_MODIFIERS',`C')dnl
FEATURE(`use_ct_file')dnl
dnl
dnl If you use IPv6 only, change [127.0.0.1] to [IPv6:::1]
FEATURE(`msp', `[127.0.0.1]')dnl
```

Por defecto, este fichero se encuentra configurado de forma que cualquier usuario local puede enviar correo electrónico a través del ordenador, por lo que no suele ser necesario modificarlo. Sus principales líneas son:

```
VERSIONID(`linux setup')dnl
```

Incluye esta información sobre la versión en el fichero de salida.

```
define(`confCF_VERSION', `Submit')
```

Define el valor de la macro *confCF_VERSION*, cuyo valor será añadido a la identificación de la versión de configuración.

```
define(`__OSTYPE__',`')
```

Es una línea añadida para cumplir la exigencia del programa de conversión *m4*.

```
define(`_USE_DECNET_SYNTAX_', `1')
```

Indica que se utilice la sintaxis DECnet, que permite la conexión de diferentes ordenadores, redes punto a punto, etc., de manera tal que los usuarios puedan compartir programas, archivos de datos y dispositivos de terminal remotos.

```
define(`confTIME_ZONE', `USE_TZ')
```

Indica de donde debe obtenerse la zona horaria del sistema. Algunos valores posibles son *USE_SYSTEM* para obtenerla del sistema, *USE_TZ* para obtenerla de la variable de ambiente *TZ*, o cualquier valor para forzar a usar esa zona horaria.

```
define(`confDONT_INIT_GROUPS', `True')
```

Indica que no se llame a la función *initgroups*, de forma que los usuarios tan solo pertenecerán al grupo que tengan asignado en el fichero */etc/passwd* y no a cualquier otro grupo adicional al que se hayan asignado en */etc/groups*.

```
define(`confPID_FILE', `/var/run/sm-client.pid')
```

Indica la localización del identificador del proceso.

```
dnl define(`confDIRECT_SUBMISSION_MODIFIERS',`C')
```

Define banderas que modifican el funcionamiento por defecto del MSP. En nuestro caso la línea esta comentada por lo que el funcionamiento es el por defecto.

```
FEATURE(`use_ct_file')
```

Indica que se lea el fichero */etc/mail/trusted-users*, el cual indica que usuarios están autorizados a enviar un correo en nombre de otros (opción *-f* de envío) sin generar un mensaje de aviso[9].

```
FEATURE(`msp', `[127.0.0.1]')
```

[9] Un ejemplo de la necesidad de este fichero es cuando un gestor de correo como *squirremail*, que es un gestor de correo con interfaz Web ejecutado a través del servidor Web apache como usuario apache, envía el correo que un usuario cualquiera del sistema ha escrito a través de su interfaz como enviado por el usuario que lo ha escrito y no como el usuario apache.

Indica el ordenador (o dirección IP) como el que se envía el correo local. El valor por defecto es [localhost], pero puede cambiarse por [127.0.0.1] para que ponga la IP en vez del nombre o [IPv6:::1] para la versión 6 del protocolo IP.

El fichero sendmail.mc.

El fichero *sendmail.mc* contiene, por defecto, las siguientes líneas:

```
divert(-1)dnl
dnl #
dnl # This is the sendmail macro config file for m4. If you make changes to
dnl # /etc/mail/sendmail.mc, you will need to regenerate the
dnl # /etc/mail/sendmail.cf file by confirming that the sendmail-cf package is
dnl # installed and then performing a
dnl #
dnl #     /etc/mail/make
dnl #
include(`/usr/share/sendmail-cf/m4/cf.m4')dnl
VERSIONID(`setup for linux')dnl
OSTYPE(`linux')dnl
dnl #
dnl # Do not advertize sendmail version.
dnl #
dnl define(`confSMTP_LOGIN_MSG', `$j Sendmail; $b')dnl
dnl #
dnl # default logging level is 9, you might want to set it higher to
dnl # debug the configuration
dnl #
dnl define(`confLOG_LEVEL', `9')dnl
dnl #
dnl # Uncomment and edit the following line if your outgoing mail needs to
dnl # be sent out through an external mail server:
dnl #
dnl define(`SMART_HOST', `smtp.your.provider')dnl
dnl #
define(`confDEF_USER_ID', ``8:12'')dnl
dnl define(`confAUTO_REBUILD')dnl
define(`confTO_CONNECT', `1m')dnl
define(`confTRY_NULL_MX_LIST', `True')dnl
define(`confDONT_PROBE_INTERFACES', `True')dnl
define(`PROCMAIL_MAILER_PATH', `/usr/bin/procmail')dnl
define(`ALIAS_FILE', `/etc/aliases')dnl
define(`STATUS_FILE', `/var/log/mail/statistics')dnl
define(`UUCP_MAILER_MAX', `2000000')dnl
define(`confUSERDB_SPEC', `/etc/mail/userdb.db')dnl
define(`confPRIVACY_FLAGS', `authwarnings,novrfy,noexpn,restrictqrun')dnl
define(`confAUTH_OPTIONS', `A')dnl
dnl #
dnl # The following allows relaying if the user authenticates, and disallows
dnl # plaintext authentication (PLAIN/LOGIN) on non-TLS links
dnl #
dnl define(`confAUTH_OPTIONS', `A p')dnl
dnl #
dnl # PLAIN is the preferred plaintext authentication method and used by
dnl # Mozilla Mail and Evolution, though Outlook Express and other MUAs do
dnl # use LOGIN. Other mechanisms should be used if the connection is not
dnl # guaranteed secure.
dnl # Please remember that saslauthd needs to be running for AUTH.
dnl #
dnl TRUST_AUTH_MECH(`EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define(`confAUTH_MECHANISMS', `EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5 LOGIN
PLAIN')dnl
dnl #
dnl # Rudimentary information on creating certificates for sendmail TLS:
dnl #     cd /etc/pki/tls/certs; make sendmail.pem
```

```
dnl # Complete usage:
dnl #     make -C /etc/pki/tls/certs usage
dnl #
dnl define(`confCACERT_PATH', `/etc/pki/tls/certs')dnl
dnl define(`confCACERT', `/etc/pki/tls/certs/ca-bundle.crt')dnl
dnl define(`confSERVER_CERT', `/etc/pki/tls/certs/sendmail.pem')dnl
dnl define(`confSERVER_KEY', `/etc/pki/tls/certs/sendmail.pem')dnl
dnl #
dnl # This allows sendmail to use a keyfile that is shared with OpenLDAP's
dnl # slapd, which requires the file to be readble by group ldap
dnl #
dnl define(`confDONT_BLAME_SENDMAIL', `groupreadablekeyfile')dnl
dnl #
dnl define(`confTO_QUEUEWARN', `4h')dnl
dnl define(`confTO_QUEUERETURN', `5d')dnl
dnl define(`confQUEUE_LA', `12')dnl
dnl define(`confREFUSE_LA', `18')dnl
define(`confTO_IDENT', `0')dnl
dnl # If you're operating in a DSCP/RFC-4594 environment with QoS
dnl define(`confINET_QOS', `AF11')dnl
dnl FEATURE(delay_checks)dnl
FEATURE(`no_default_msa', `dnl')dnl
FEATURE(`smrsh', `/usr/sbin/smrsh')dnl
FEATURE(`mailertable', `hash -o /etc/mail/mailertable.db')dnl
FEATURE(`virtusertable', `hash -o /etc/mail/virtusertable.db')dnl
FEATURE(redirect)dnl
FEATURE(always_add_domain)dnl
FEATURE(use_cw_file)dnl
FEATURE(use_ct_file)dnl
dnl #
dnl # The following limits the number of processes sendmail can fork to accept
dnl # incoming messages or process its message queues to 20.) sendmail refuses
dnl # to accept connections once it has reached its quota of child processes.
dnl #
dnl define(`confMAX_DAEMON_CHILDREN', `20')dnl
dnl #
dnl # Limits the number of new connections per second. This caps the overhead
dnl # incurred due to forking new sendmail processes. May be useful against
dnl # DoS attacks or barrages of spam. (As mentioned below, a per-IP address
dnl # limit would be useful but is not available as an option at this
writing.)
dnl #
dnl define(`confCONNECTION_RATE_THROTTLE', `3')dnl
dnl #
dnl # The -t option will retry delivery if e.g. the user runs over his quota.
dnl #
FEATURE(local_procmail, `', `procmail -t -Y -a $h -d $u')dnl
FEATURE(`access_db', `hash -T<TMPF> -o /etc/mail/access.db')dnl
FEATURE(`blacklist_recipients')dnl
EXPOSED_USER(`root')dnl
dnl #
dnl # For using Cyrus-IMAPd as POP3/IMAP server through LMTP delivery
uncomment
dnl # the following 2 definitions and activate below in the MAILER section the
dnl # cyrusv2 mailer.
dnl #
dnl define(`confLOCAL_MAILER', `cyrusv2')dnl
dnl define(`CYRUSV2_MAILER_ARGS', `FILE /var/lib/imap/socket/lmtp')dnl
dnl #
dnl # The following causes sendmail to only listen on the IPv4 loopback
address
dnl # 127.0.0.1 and not on any other network devices. Remove the loopback
dnl # address restriction to accept email from the internet or intranet.
dnl #
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
dnl #
dnl # The following causes sendmail to additionally listen to port 587 for
dnl # mail from MUAs that authenticate. Roaming users who can't reach their
```

```
dnl # preferred sendmail daemon due to port 25 being blocked or redirected
find
dnl # this useful.
dnl #
dnl DAEMON_OPTIONS(`Port=submission, Name=MSA, M=Ea')dnl
dnl #
dnl # The following causes sendmail to additionally listen to port 465, but
dnl # starting immediately in TLS mode upon connecting. Port 25 or 587
followed
dnl # by STARTTLS is preferred, but roaming clients using Outlook Express
can't
dnl # do STARTTLS on ports other than 25. Mozilla Mail can ONLY use STARTTLS
dnl # and doesn't support the deprecated smtps; Evolution <1.1.1 uses smtps
dnl # when SSL is enabled-- STARTTLS support is available in version 1.1.1.
dnl #
dnl # For this to work your OpenSSL certificates must be configured.
dnl #
dnl DAEMON_OPTIONS(`Port=smtps, Name=TLSMTA, M=s')dnl
dnl #
dnl # The following causes sendmail to additionally listen on the IPv6
loopback
dnl # device. Remove the loopback address restriction listen to the network.
dnl #
dnl DAEMON_OPTIONS(`port=smtp,Addr=::1, Name=MTA-v6, Family=inet6')dnl
dnl #
dnl # enable both ipv6 and ipv4 in sendmail:
dnl #
dnl DAEMON_OPTIONS(`Name=MTA-v4, Family=inet, Name=MTA-v6, Family=inet6')
dnl #
dnl # We strongly recommend not accepting unresolvable domains if you want to
dnl # protect yourself from spam. However, the laptop and users on computers
dnl # that do not have 24x7 DNS do need this.
dnl #
FEATURE(`accept_unresolvable_domains')dnl
dnl #
dnl FEATURE(`relay_based_on_MX')dnl
dnl #
dnl # Also accept email sent to "localhost.localdomain" as local email.
dnl #
LOCAL_DOMAIN(`localhost.localdomain')dnl
dnl #
dnl # The following example makes mail from this host and any additional
dnl # specified domains appear to be sent from mydomain.com
dnl #
dnl MASQUERADE_AS(`mydomain.com')dnl
dnl #
dnl # masquerade not just the headers, but the envelope as well
dnl #
dnl FEATURE(masquerade_envelope)dnl
dnl #
dnl # masquerade not just @mydomainalias.com, but @*.mydomainalias.com as well
dnl #
dnl FEATURE(masquerade_entire_domain)dnl
dnl #
dnl MASQUERADE_DOMAIN(localhost)dnl
dnl MASQUERADE_DOMAIN(localhost.localdomain)dnl
dnl MASQUERADE_DOMAIN(mydomainalias.com)dnl
dnl MASQUERADE_DOMAIN(mydomain.lan)dnl
MAILER(smtp)dnl
MAILER(procmail)dnl
dnl MAILER(cyrusv2)dnl
```

Lo primero comentar que aunque no aparezca explícitamente una línea `divert(0)`, el fichero que se incluye al principio (`/usr/share/sendmail-cf/m4/cf.m4`) tiene la citada línea, por lo que habilita que las líneas siguientes sean incluidas en la configuración.

Las principales líneas del fichero *sendmail.mc* son las siguientes:

```
VERSIONID(`setup for linux')dnl
```

Incluye esta información sobre la versión en el fichero de salida.

```
OSTYPE(`linux’)
```

Especifica el sistema operativo sobre el que se ejecuta el demonio de *sendmail*. Sirve para que el sistema pueda configurar el camino de ficheros como los de ayuda y de estado, las opciones necesarias para el correcto funcionamiento de *sendmail*, etc. Si se omite se produce un error en la configuración.

```
dnl define(`SMART_HOST',`smtp.your.provider')
```

Especifica el ordenador a través del cual se enviará el correo electrónico saliente, en lugar de enviarlo directamente al ordenador especificado en el mensaje. Así, por ejemplo, si se activa la línea anterior (eliminando la indicación de comentario) y se deja como:

```
define(`SMART_HOST',`postin.uv.es')
```

Se indica que todo el correo electrónico saliente se envíe a través del ordenador *postin.uv.es* en lugar de enviarse directamente al ordenador de destino[10].

Es necesario resaltar que la resolución del nombre puesto en la línea SMART_HOST en su dirección IP se realiza utilizando el valor MX de los servidores de nombres. Por ello, si para un determinado ordenador los valores de su registro MX y de su registro A son distintos, y se desea utilizar el registro de tipo A (la dirección del ordenador), se debe poner el nombre entre corchetes cuadrados. Por ejemplo, si se desea utilizar la dirección IP de glup.uv.es (147.156.222.65) y no su registro MX (147.156.1.90), se debería poner:

```
define(`SMART_HOST',`[glup.uv.es]')
```

Continuando con las líneas del fichero, la línea:

```
define(`confDEF_USER_ID`,``8:12’’)
```

Define el identificador de usuario (UID) y de grupo (GID) con los que se ejecutara el servidor de SMTP. En este caso son el usuario 8 y grupo 12 que corresponden a un usuario de nombre *mail* y grupo *mail*.

```
define(`confTO_CONNECT’, `1m’)
```

Definición del tiempo inicial de espera hasta que la conexión se realice (1 minuto).

[10] Esta opción es necesaria, por ejemplo, si tan solo un ordenador esta autorizado a enviar correo a través del cortafuegos perimetral de una red, pero existen varios ordenadores con capacidad para aceptar y enviar correo.

```
define(`confTRY_NULL_MX_LIST’,true)
```

Indica que este ordenador es el mejor intercambiador de correo (Mail Exchanger) para recibir el correo con este destino.

```
define(`confDONT_PROBE_INTERFACES’,true)
```

Indica que el ordenador no inserte el nombre ni las direcciones de cualquier interface local en la lista de direcciones “equivalentes”. Su valor por defecto es false.

```
define(`PROCMAIL_MAILER_PATH',`/usr/bin/procmail')
```

Indica donde se encuentra el programa *procmail*, que es llamado cuando un usuario reenvía el correo que le llega a este ordenador a otro ordenador mediante un archivo *.forward* en su directorio raíz.

```
define(`ALIAS_FILE', `/etc/aliases')
```

Define el camino donde se encuentra el fichero de alias de correo del ordenador[11].

```
define(`STATUS_FILE', `/var/log/mail/statistics')
```

Indica donde se deben guardar la información sobre el estado del programa.

```
define(`UUCP_MAILER_MAX', `2000000')
```

Define el tamaño máximo del correo que puede recibirse mediante UUCP, su valor por defecto es de 1000000.

```
define(`confUSERDB_SPEC', `/etc/mail/userdb.db')
```

Especificación de la localización de la base de datos de los usuarios autorizados a enviar correo. Si no existe todos los usuarios pueden enviar correo.

```
define(`confPRIVACY_FLAGS', `authwarnings,novrfy,noexpn,restrictqrun')
```

Especifica las banderas de privacidad, por ejemplo, no permitir utilizar la verificación de usuarios (novrfy), no permitir la expansión de los alias (noexpn), limitar el procesamiento de la cola de mensajes al usuario root (restrictqrun), etc. Por defecto solo esta activada authwarnings[12].

```
define(`confAUTH_OPTIONS', `A')
```

[11] Siempre que se modifique el fichero de alias debe ejecutarse el comando newaliases para que la modificación se ejecute y tenga efecto.

[12] Por seguridad, se deberían siempre activar las opciones novrfy y noexpn para evitar que un usuario malintencionado obtenga información sobre los nombres de los usuarios existentes en el ordenador mediante el uso de esas dos opciones del protocolo SMTP.

Especifica que todos los usuarios pueden utilizar *sendmail* sin necesidad de autenticarse. Si se desea que los usuarios deban autenticarse para utilizar este servidor de correo electrónico la línea anterior debe comentarse, descomentando la línea:

```
define(`confAUTH_OPTIONS', `A p')
```

La cual indica que los usuarios deben autenticarse para poder enviar correo electrónico a través del servidor. Esta autenticación debe realizarse utilizando uno de los mecanismos indicados en las líneas:

```
dnl TRUST_AUTH_MECH(`EXTERNAL DIGEST-MD5 CRAM-MD5 LOGIN PLAIN')dnl
dnl define(`confAUTH_MECHANISMS', `EXTERNAL GSSAPI DIGEST-MD5 CRAM-MD5
LOGIN PLAIN')dnl
```

Donde en caso de ser utilizado el mecanismo de usuario/contraseña, este deberá siempre ir sobre SSL. En este caso, la localización de los ficheros con los certificados del servidor, etc., se especifican en las entradas:

```
dnl define(`confCACERT_PATH',`/etc/pki/tls/certs')
dnl define(`confCACERT',`/etc/pki/tls/certs/ca-bundle.crt')
dnl define(`confSERVER_CERT',`/etc/pki/tls/certs/sendmail.pem')
dnl define(`confSERVER_KEY',`/etc/pki/tls/certs/sendmail.pem')
```

De forma general, en un servidor de *sendmail* puede ser suficiente utilizar la limitación de acceso a ordenadores/subredes que veremos más adelante.

```
dnl define(`confTO_QUEUEWARN', `4h')
```

Define el tiempo por defecto que esperará el servidor antes de enviar un mensaje al usuario que envía un correo indicando que su correo aún no ha podido ser entregado. El valor por defecto es de 4 horas.

```
dnl define(`confTO_QUEUERETURN', `5d')
```

Define el tiempo por defecto que esperará el servidor antes de desistir de enviar un correo e informar al usuario que envía un correo de que su mensaje ya no será enviado. El valor por defecto es de 4 horas.

```
dnl define(`confQUEUE_LA', `12')
dnl define(`confREFUSE_LA', `18')
```

Indican el promedio de carga del procesador[13] a partir del cual los mensajes recibidos para su envío (*confQUEUE_LA*) o entrantes (*confREFUSE_LA*) serán rechazados. Sus valores por defecto son 8 y 12, respectivamente.

```
define(`confTO_IDENT', `0')
```

Especifica el tiempo de espera antes de aceptar una conexión TCP. El valor por defecto es de 5 segundos.

[13] El promedio es por procesador, por lo que debe multiplicarse por el número de procesadores existentes para obtener la carga total del sistema antes de que suceda el rechazo.

```
dnl FEATURE(delay_checks)
```

Indica que no se ejecuten la comprobación del cliente que se conecta o ejecuta un comando de correo.

```
FEATURE(`no_default_msa',`dnl')
```

Bloquea la definición de *no_default_msa* mediante el valor *dnl*, permitiendo la definición por defecto de un MSA para la recepción de correo.

```
FEATURE(`smrsh',`/usr/sbin/smrsh')
```

Especifica que el programa *sendmail* utilice, cuando sea necesario, la SendMail Restricted SHell en lugar de una shell estandar. Además, se especifica la ubicación de la misma.

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

Incluye una “tabla de mail” que será usada para sobrescribir la ruta para algunos dominios particulares[14].

Continuando con la explicación del fichero, la línea:

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

Especifica la forma del alias que permite múltiples dominios virtuales para el servidor de correo[15].

```
FEATURE(redirect)
```

Rechazar el correo enviado a usuarios cuya cuenta ha sido trasladada a otro servidor mediante el código de error 551.

```
FEATURE(always_add_domain)
```

Incluye el dominio del ordenador local en el mail recibido desde el mismo. Por defecto no es incluido.

```
FEATURE(use_cw_file)
```

Lee el fichero */etc/mail/local-host-names* para seleccionar otros nombres posibles para este ordenador[16].

```
FEATURE(use_ct_file)
```

[14] El contenido del fichero *mailertable* se verá con posterioridad.
[15] Los dominios virtuales de correo deben entenderse, salvando su uso, de forma similar a los dominios virtuales de HTTP que hemos visto con anterioridad. Un ejemplo del mismo se verá con posterioridad.
[16] Este fichero será explicado con posterioridad.

Lee el fichero */etc/mail/trusted-user* para seleccionar los nombres de usuarios que son de confianza y pueden enviar un correo en nombre de otro usuario sin que ello genere un mensaje de aviso.

```
dnl define(`confMAX_DAEMON_CHILDREN', 12)
```

Especifica el número máximo de procesos hijos que puede lanzar el servidor para atender las peticiones. Un valor <= 0 o su no especificación implican un número ilimitado de procesos hijos.

```
dnl define(`confCONNECTION_RATE_THROTTLE', 3)dnl
```

Especifica el número máximo de conexiones por segundo que son permitidas, siendo retrasadas las conexiones que sobrepasen este valor. Un valor <=0 o su no especificación implican un número ilimitado de conexiones por segundo.

```
FEATURE(local_procmail,`',`procmail -t -Y -a $h -d $u')
```

Especifica la forma en que será llamado el programa *procmail* en el reenvío de un correo.

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

Habilita la base de datos que permite aceptar el correo, por razones administrativas (anti-SPAM), de determinados dominios u ordenadores para su reenvío fuera de este ordenador El resto del correo de otros dominios es rechazado por defecto.

```
FEATURE(`blacklist_recipients')
```

Habilita la posibilidad de bloquear los correos de ciertos usuarios, nombres de ordenador o direcciones IP.

```
EXPOSED_USER(`root')
```

Indica los usuarios deben ser siempre mostrados en lugar de otros nombres que puedan tener.

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

Esta línea, que por defecto esta sin comentar, indica al demonio de SMTP que tan solo permanezca a la escucha de conexiones que se realicen a la dirección de loopback (IP 127.0.0.1). Para permitir que el demonio de SMTP escuche y acepte las conexiones en cualquiera de sus direcciones IP debe comentarse la opción:

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

Un uso interesante de esta opción es la posibilidad de limitar a una serie de direcciones IP del ordenador la escucha de los servicios de SMTP. Así, si incluimos las líneas:

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

```
DAEMON_OPTIONS(`Port=smtp,Addr=147.156.222.34, Name=MTA')
```

Permitimos que el servidor de SMTP tan solo permanezca a la escucha en el interfaz de loopback y en la dirección IP 147.156.222.34 del ordenador, no permaneciendo a la escucha en cualquier otra dirección de red que posea el ordenador.

```
FEATURE(`accept_unresolvable_domains')
```

Habilita aceptar el correo que se recibe de dominios no resolubles, esto es, aquellos en los que en MAIL FROM: *usuario@dominio*, el dominio no puede ser resuelto por el DNS. Esta opción permite bloquear el envío de correos por ordenadores no registrados en Internet, aunque puede producir, si se produce un fallo en la resolución del nombre por un problema en algún DNS, que se rechace temporalmente correo valido.

```
dnl FEATURE(`relay_based_on_MX')
```

Habilita el permitir el reenvío de correo basado en registros DNS de tipo MX (Mail eXchanger).

```
MAILER(smtp)
```

Especifica los servidores de correo que son utilizados en este sistema. Por defecto, el servidor “local” siempre es incluido, aunque no se ponga. El resto hay que especificarlos. Pueden ser “smtp”, “uucp”, “procmail”, etc.

Además de las líneas anteriores, suele ser útil introducir algunas líneas adicionales que permitan nuevas funcionalidades del correo. Estas pueden ser muy variadas, pero una de las más utilizadas consiste en la introducción de un filtro de correo que permite al servidor analizar el correo en busca de virus. Un ejemplo de esto son las siguientes líneas:

```
dnl # Introduccion del ClamAV como antivirus
INPUT_MAIL_FILTER(`clamav-
milter',`S=local:/var/run/clamav/clmilter.socket,F=,
T=S:4m;R:4m')
define(`confINPUT_MAIL_FILTERS',`clamav-milter')
```

Las cuales configuran como filtro de entrada del correo el antivirus ClamAV, de forma que todo correo entrante o saliente es chequeado por el antivirus[17].

Configuración de los ficheros de base de datos (.db).

Los ficheros que especifican bases de datos con información para el funcionamiento de *sendmail* son *access*, *domaintable*, *local-host-names*, *mailertable*, *trusted-users* y *virtusertable*.

El fichero *access* contiene los ordenadores que están autorizados a enviar correo a otros ordenadores utilizando este ordenador como SMTP, así como otras propiedades

[17] El código fuente e información más detallada sobre este antivirus de distribución gratuita puede encontrarse en la URL http://www.clamav.net

como puede ser el limite de conexiones que puede realizar un cliente. Un ejemplo es el siguiente:

```
Connect:localhost.localdomain          RELAY
Connect:localhost                      RELAY
Connect:127.0.0.1                      RELAY
Connect:147.156.222                    RELAY
Connect:147.156.223                    RELAY
Connect:irobot.uv.es                   RELAY
ClientRate:127.0.0.1                   0
ClientRate:147.156.1.90                0
ClientRate:147.156.0.253               0
ClientRate:147.156.222.65              0
ClientRate:                            21
```

Donde las tres primeras líneas indican que el propio ordenador (*localhost.localdomain*, *localhost* y *127.0.0.1*) esta autorizado a enviar correo fuera de él, las líneas *147.156.222* y *147.156.223* indican que todos los ordenadores cuya dirección IP empiece por esos valores están también autorizados a enviar correo. Por último, la línea siguiente indica que todos los ordenadores del dominio *irobot.uv.es* están también autorizados

Con esta configuración la dirección de conexión de un ordenador es comprobada por el DNS y si en la resolución inversa se obtiene que su dominio es *irobot.uv.es* el correo es aceptado para su envío a cualquier ordenador de Internet. Sin embargo, otros ordenadores que no pertenezcan al dominio *irobot.uv.es*, y cuyo correo no tenga como destino final el servidor donde se ejecuta *sendmail* será rechazado. Esto puede apreciarse en el siguiente ejemplo:

```
S 220 glup.uv.es ESMTP Sendmail 8.12.8/8.12.5; Sun, 18 Feb 2007 17:24:30 +0100
C HELO slabii.informat.uv.es
S 250 glup.uv.es Hello slabii.informat.uv.es [147.156.16.31], pleased to meet you
C MAIL FROM: ebonet@slabii.informat.uv.es
S 250 2.1.0 ebonet@slabii.informat.uv.es... Sender ok
C RCPT TO: Enrique.Bonet@uv.es
S 550 5.7.1 Enrique.Bonet@uv.es... Relaying denied
```

Pues el ordenador *slabii.informat.uv.es*, no esta autorizado a utilizar nuestro servidor de SMTP para enviar correo a otros ordenadores, pudiendo tan solo enviar correo con destino final un usuario de nuestro ordenador.

Por otra parte, las líneas ClientRate indican el número máximo de conexiones que puede establecer un ordenador en el intervalo de tiempo indicado por la variable confCONNECTION_RATE_WINDOW_SIZE, cuyo valor por defecto es de 60 segundos, pero que puede modificarse en el fichero de configuración *sendmail.mc*. Así, en nuestro caso, los ordenadores de dirección IP 127.0.0.1, 147.156.1.90, 147.156.0.253 y 147.156.222.65 poseen un número ilimitado de conexiones, mientras que cualquier otro ordenador puede establecer un máximo de 20 conexiones en el intervalo de tiempo.

El fichero *domaintable* sirve para especificar otros dominios por los que puede ser conocido nuestro ordenador. Generalmente solo se utiliza cuando se cambia el dominio de un servidor de correo, por lo que usualmente esta vacío.

El fichero *local-host-names* contiene todos los nombres por los que se puede hacer referencia al ordenador donde se ejecuta el servidor de SMTP. Dicho fichero, que no es convertido en base de datos, contiene, por ejemplo, las siguientes líneas:

```
# local-host-names - include all aliases for your machine here.
localhost
localhost.localdomain
glup
robotica
irtic
glup.uv.es
robotica.uv.es
irtic.uv.es
glup.irobot.uv.es
robotica.irobot.uv.es
irtic.irobot.uv.es
```

Donde, como puede verse, se encuentran tanto los nombres del ordenador (*glup* y su alias *robotica*) como las entradas que se refieren siempre al propio ordenador (*localhost*).

El fichero *mailertable*, que vimos con anterioridad, contiene información sobre determinados ordenadores para los que se modifica el funcionamiento del servidor de correo. Por ejemplo, si el fichero *mailertable* contiene las siguientes líneas:

```
uv.es           esmtp:[post.uv.es]
valencia.edu    esmtp:[post.uv.es]
alumni.uv.es    esmtp:[post.uv.es]
.uv.es          esmtp:[%1.uv.es]
```

Especifica que todo el correo cuyo dominio sea *uv.es*, *valencia.edu* o *alumni.uv.es* sea enviado directamente a la dirección IP del ordenador *post.uv.es*, mientras que otros correos como *usuario@ordenador.uv.es* sean enviados directamente al ordenador indicado. Esto permite, por ejemplo, que los correos internos de un dominio no sean enviados al SMART_HOST, pues con estas reglas se le indica al servidor que a esos dominios se puede acceder directamente y no es necesario utilizar el SMART_HOST.

El fichero *virtusertable* especifica una tabla de direcciones de correo virtuales, de forma que el correo enviado a uno de las direcciones de correo virtuales es reenviado a la dirección de correo indicada. Por ejemplo:

```
root@robotica.uv.es         quique
quique@robotica.uv.es       ebonet@amparo.uv.es
@correo.cdlibre.org         barto
```

Indica que el correo a *root@robotica.uv.es* será enviado al usuario local *quique*, mientras que el correo enviado al usuario *quique@robotica.uv.es* será enviado al usuario *ebonet@amparo.uv.es*, que como podemos ver es un ordenador externo (e incluso en el ejemplo tiene nombre de usuario distinto), mientras que todo el correo destinado a *correo.cdlibre.org* será enviado al usuario *barto.*

Por último, el fichero trusted-users contiene una lista de usuarios que pueden enviar correo en nombre de otros usuarios sin que ello genere un aviso al administrador del sistema. Un ejemplo de este fichero es el siguiente:

```
# trusted-users - users that can send mail as others without a
# warning apache, mailman, majordomo, uucp, are good candidates
apache
```

Donde podemos ver que se ha definido el usuario *apache*[18] como un usuario que puede enviar correo en nombre de otros usuarios sin generar un aviso del sistema. Esto es necesario, por ejemplo, en sistemas que ejecutan un interfaz web para el envío y recepción de correo, tal y como sucede en la Universidad de Valencia.

El fichero de alias.

En el directorio */etc* existe un fichero, de nombre *aliases*, que contiene la definición de "alias" de correo. Dicho fichero es transformado en la base de datos */etc/aliases.db* mediante el comando */usr/bin/newaliases*.

El fichero de "alias" contiene una relación de otros nombres que tienen los usuarios, así como una relación de nombres que hacen referencia a varios usuarios, siendo enviado a cada uno de ellos un correo que tenga como destino el nombre original. Un ejemplo sencillo de fichero */etc/aliases* es:

```
# Basic system aliases -- these MUST be present.
mailer-daemon:    postmaster
postmaster:       root

# General redirections for pseudo accounts.
bin:              root
daemon:           root
adm:              root
lp:               root
...

# trap decode to catch security attacks
decode:           root

# Alias creados

# Alias del administrador de las paginas WEB.

webmaster:        root

# Lista de administradores

administradores:  quique@glup.uv.es, susana@glup.uv.es

# Lista de alias de nombres de usuarios

Enrique.Bonet:    quique@glup.uv.es
Susana.Pons:      susana@glup.uv.es
```

[18] Suponemos que el usuario apache es el usuario como el que se ejecuta el servidor de HTTP,

Además de las líneas anteriores, el fichero *aliases* puede contener líneas que supongan la ejecución de algún comando, como pueden ser las siguientes líneas:

```
lista1:          "|/usr/lib/mailman/mail/mailman post lista1"
lista1-admin:    "|/usr/lib/mailman/mail/mailman admin lista1"
...
```

Donde se esta indicando que para enviar el correo a un usuario o usuarios llamado *lista1* se ejecute el comando indicado con posterioridad, que en este caso no es más que una llamada al programa *mailman*[19].

Protocolos de entrega final.

Como comentamos en la introducción, actualmente los correos suelen ser leídos y escritos en los ordenadores personales, debiendo por tanto poder ser enviados y recibidos estos ordenadores personales.

En el envío de correo desde un ordenador personal a un servidor de correo, el ordenador personal utiliza el protocolo SMTP, actuando el ordenador personal como cliente de SMTP y el servidor de correo como servidor de SMTP.

Sin embargo, la recepción del correo existente en el servidor de correo por parte del ordenador personal se realiza mediante los protocolos POP3 o IMAP[20], siendo el ordenador personal un cliente de POP3 ó IMAP y, por tanto, el servidor de correo debe actuar como servidor de POP3 y/o IMAP[21], por lo que debemos configurarlo para ello.

El programa servidor de POP3 e IMAP es */usr/sbin/dovecot* y utiliza, para su ejecución, los programas *imap*, *imap-login*, *pop3* y *pop3-login* que se encuentran en el directorio */usr/libexec/dovecot*. El funcionamiento de *dovecot* es sencillo, pues se encarga de controlar el acceso a los servicios de POP3 e IMAP, lanzando servidores de estos servicios a medida que son requeridos por los clientes.

La configuración de *dovecot* se realiza en el fichero */etc/dovecot/dovecot.conf* y en los ficheros que se encuentran dentro del directorio */etc/dovecot/conf.d*.

El fichero */etc/dovecot/dovecot.conf* contiene las líneas generales de configuración del servidor, de las cuales podemos destacar:

```
protocols = imap pop3
```

Que indica los protocolos que atiende dovecot en el arranque, en este caso los protocolos IMAP y POP3.

```
listen = *
```

[19] Mailman es un paquete que permite administrar, manejar y enviar correo a listas de correo.
[20] Existen más protocolos de entrega del correo desde el servidor al cliente, pero POP3 e IMAP son los más utilizados.
[21] El protocolo POP3 utiliza el puerto 110/TCP ó el 995/TCP si es sobre SSL, mientras que el protocolo IMAP utiliza el puerto 143/TCP ó el 993/TCP si es sobre SSL.

Que configura los interfaces de red en los que *dovecot* permanece a la escucha. Si se especifica * se indican todos los interfaces de red del servidor, mientras que si se especifica una IP o una lista de IPs separadas por comas *dovecot* solo se pondrá a la escucha en esos interfaces de red.

```
base_dir = /var/run/dovecot/
```

Directorio donde guarda el programa información sobre su ejecución.

```
instance_name = dovecot
```

Nombre de la instancia. Su uso es necesario solo si se lanzan múltiples instancias de *dovecot*, pues permiten a programas externos poder referirse a cada una de las instancias.

De los ficheros que se encuentran dentro del directorio /etc/dovecot/conf.d es necesario modificar, para un correcto funcionamiento del servicio, los ficheros *10-auth.conf*, *10-mail.conf*, *10-ssl.conf*, *20-imap.conf* y *20-pop3.conf*.

El fichero *10-auth.conf* contiene los mecanismos de autenticación que pueden utilizarse y sus posibilidades de uso. La línea:

```
auth_mechanisms = plain
```

Indica los mecanismos de autenticación que se permiten, en este caso usuario/contraseña (*plain*), aunque puede tomar otros valores como login, CRAM-MD5, DIGEST-MD5, etc., e incluso la línea puede contener varios de estos valores, separados por espacios, para indicar posibles valores de autenticación aceptados. Si el mecanismo de autenticación es usuario/contraseña, la línea:

```
disable_plaintext_auth = yes
```

Indica si el mecanismo se puede utilizar siempre (valor no) o solamente se puede utilizar si la conexión es segura pues está cifrada utilizando SSL/TLS (valor yes). Es necesario resaltar que si la dirección IP del cliente es igual a una dirección IP local del servidor, la conexión se considera segura y puede realizarse sin utilizar SSL/TLS.

El fichero *10-mail.conf* configura la localización de los buzones de correo de los usuarios, los permisos sobre ellos, etc.. Posee muchos parámetros de configuración, incluyendo tiemouts, etc., pero suele ser suficiente con modificar los parámetros:

```
mail_location = mbox:~/mail:INBOX=/var/mail/%u
```

Donde le indicamos la localización de los buzones de correo de cada usuario, en nuestro caso indicamos que se encuentran dentro de un directorio llamado *mail* en el directorio raíz de cada usuario y dentro de */var/mail* en un fichero con el nombre del usuario. Además, el parámetro:

```
mail_access_groups = mail
```

Indica los grupos adicionales que puede utilizar *dovecot* para ejecutar sus tareas. En este caso le indicamos que puede utilizar el grupo *mail* para acceder, por ejemplo, a los buzones de correo de los usuarios, que suelen tener como grupo el grupo *mail*.

Por otra parte, el fichero *10-ssl.conf* contiene la configuración de SSL en *dovecot*. En concreto, la opción:

```
ssl = { no | yes | required }
```

Indica si se deshabilita el soporte de SSL (valor *no*), se habilita el soporte de SSL (valor *yes*) para la autenticación basada en usuario/contraseña (*plain*) o se requiere para cualquier tipo de autenticación (valor *required*). De forma general debería usarse:

```
ssl = required
```

Por ser la configuración más segura. Además, los valores:

```
ssl_cert = </etc/pki/dovecot/certs/dovecot.pem
ssl_key = </etc/pki/dovecot/private/dovecot.pem
```

Indica la localización del certificado y la clave privada utilizada por SSL. Si la clave privada esta protegida por contraseña, la contraseña de acceso a la misma debe ponerse en el valor de configuración:

```
ssl_key_password = <fichero_clave
```

Donde *fichero_clave* es un fichero, con permisos 0600 y propietario root, donde debe guardarse la contraseña que permite acceder a la clave privada[22].

Por último, los ficheros *20-imap.conf* y *20-pop3.conf* contienen valores para configurar el funcionamiento, los límites de tamaño de los correos, etc., aunque los valores por defecto suelen ser correctos para la mayoría de casos y no suelen modificarse estos ficheros.

Ejercicios.

1- Deseamos configurar un ordenador, de nombre correo.empresa.com, como servidor de sendmail de forma que permita enviar correo a través del mismo a los ordenadores del dominio 147.156.222.0/23, debiendo rechazar los correos de los dominios no resolubles.

2- Configurar un servidor de sendmail, de nombre correo.empresa.com, de forma que solo permita el envío de correo a los ordenadores del dominio 147.156.0.0/16, exigiendo además que los usuarios se identifiquen mediante su usuario y contraseña.

3- Escribir la configuración para que un ordenador, de IP 147.156.222.65, actúe como servidor de dovecot de forma que permita el protocolo POP3, pero solo sobre SSL y a la interfaz pública del ordenador.

[22] La contraseña también puede indicarse aquí directamente mediante la línea *ssl_key_password = password*, pero posee el problema de que cualquier usuario del sistema puede leer el fichero y por tanto acceder a la contraseña.

Apéndice A: Comandos de envío de mensajes mediante SMTP y ESMTP.

En la tabla siguiente se encuentra un listado de los comandos existentes en la definición de SMTP[23]:

Comando	Argumentos	Descripción
HELO	<dominio>	Identifica el ordenador remitente del correo al ordenador destinatario.
MAIL FROM:	<usuario origen>	Identifica el usuario emisor del correo.
RCPT TO:	<usuario destino>	Identifica un usuario individual destinatario del correo. Si es necesario identificar múltiples destinatarios es necesario repetir el comando.
DATA		Permite enviar las líneas de texto que forman el correo. El tamaño máximo de una línea es de 1.000 caracteres. Cada línea va seguida de un retorno de carro y avance de línea <CR><LF>. La última línea debe llevar únicamente el carácter punto "." seguido de <CR><LF>.
RSET		Aborta la transacción de correo actual.
SEND FROM:	<usuario origen>	Identifica el usuario emisor del correo e indica que si el destinatario está conectado, el mensaje se entregue directamente en el terminal.
SOML FROM:	<usuario origen>	Identifica el usuario emisor del correo e indica que si el destinatario está conectado, entrega el mensaje directamente al terminal; en caso contrario se entrega como correo convencional.
SAML FROM:	<usuario origen>	Identifica el usuario emisor del correo e indica que el mensaje se entregue en el buzón del destinatario y, que en caso de estar conectado, también se entregue en su terminal.
VRFY	<usuario>	Pide al receptor que verifique si el usuario destinatario del correo existe.
EXPN	<lista de correo>	Pide al receptor la confirmación de la existencia de la lista de correo y que devuelva los usuarios contenidos en la lista.
HELP	[comando]	Pide al ordenador destinatario información sobre los comandos disponibles o sobre un comando en concreto si es indicado como argumento.
NOOP		No operación. Indica al ordenador destino que envíe una respuesta positiva.
QUIT		Pide al ordenador destino que envíe una respuesta positiva y cierre la conexión.
TURN		El ordenador emisor pide que se inviertan los papeles, para poder actuar como receptor. El actual ordenador receptor puede negarse a dicha petición.

Los comandos que aparecen en cursiva son aquellos comandos que todo servidor de SMTP debe implementar por defecto. El resto de comandos son opcionales y pueden no estar implementados.

Cada comando SMTP enviado por el emisor es contestado por el receptor mediante una cadena de texto ASCII, que contiene en primer lugar un código numérico de tres dígitos que indica la ejecución, error en la ejecución, etc., del comando solicitado, y a continuación un texto indicativo del significado del código numérico. El

[23] En la especificación de los argumentos se utilizara la sintaxis de que los argumentos que se encuentran entre los símbolos < y > son obligatorios y los que se encuentran entre los símbolos [y] son optativos y pueden no aparecer.

texto indicativo es libre y puede no aparecer, pues el ordenador emisor solo analiza el código numérico de la respuesta ignorando el texto indicativo. Los códigos numéricos y su significado pueden verse a continuación:

Código	Descripción
211	Estado del sistema o respuesta al mensaje HELP.
214	Mensaje de ayuda. Esta respuesta es solo para uso de un humano y es ignorada por el ordenador.
220	Servicio preparado.
221	Servicio cerrando el canal de transmisión.
250	Solicitud completada con éxito.
251	Usuario no local, se enviará a <dirección de reenvío>
354	Comienzo del mensaje de correo, finalice con <CR><LF>.<CR><LF>.
421	Servicio no disponible, cerrando la conexión, generalmente cuando el ordenador destino esta apagándose.
450	Solicitud de correo no ejecutada, buzón de correo no disponible.
451	Ejecución de la acción requerida abortada, error local de procesamiento.
452	Ejecución de la acción requerida abortada, insuficiente espacio de almacenamiento en el sistema.
500	Error de sintaxis, comando no reconocido.
501	Error de sintaxis en parámetros o argumentos.
502	Comando no implementado.
503	Secuencia de comandos errónea.
504	Parámetro no implementado.
550	Solicitud no ejecutada, buzón no disponible.
551	Usuario no local, por favor pruebe <dirección de reenvío>.
552	Ejecución de la acción requerida abortada. Excedido el límite máximo de almacenamiento permitido para ese usuario.
553	Solicitud no realizada, buzón de correo no disponible.
554	Fallo en la transacción.

Cada comando SMTP puede ser respondido con un conjunto concreto de códigos numéricos de entre todos los anteriores, según suceda, falle o ocurra un error[24]. La relación de comandos SMTP y los códigos numéricos que pueden recibir como respuesta es la siguiente:

Comando	Ejecución	Códigos de respuesta
“Establecimiento de conexión”	Sucede	220
	Falla	421
HELO	Sucede	250
	Error	421, 500, 501, 504
MAIL FROM:	Sucede	250
	Falla	451, 452, 552
	Error	421, 500, 501
RCPT TO:	Sucede	250, 251
	Falla	450, 451, 452, 550, 551, 552, 553
	Error	421, 500, 501, 503
DATA	Intermedio	354 -> datos -> estado
	Sucede	250
	Falla	451, 452, 552, 554
RSET	Sucede	250
	Error	421, 500, 501, 503
SEND FROM:	Sucede	250
	Falla	451, 452, 552
	Error	421, 500, 501, 502
	Sucede	250

[24] Existe un cuarto estado que es el estado intermedio que aparece cuando ejecutamos el comando DATA y comenzamos a enviar los datos y que concluye en uno de los estados anteriores.

Comando	Ejecución	Códigos de respuesta
SOML FROM:	Falla	451, 452, 552
	Error	421, 500, 501, 502
SAML FROM:	Sucede	250
	Falla	451, 452, 552
	Error	421, 500, 501, 502
VRFY	Sucede	250, 251
	Falla	550, 551, 553
	Error	421, 500, 501, 502, 504
EXPN	Sucede	250
	Falla	550
	Error	421, 500, 501, 502, 504
HELP	Sucede	211, 214
	Error	421, 500, 501, 502, 504
NOOP	Sucede	250
	Error	421, 500
QUIT	Sucede	221
	Error	500
TURN	Sucede	250
	Error	502

Un ejemplo de dialogo entre un cliente (líneas que comienzan por C) y un servidor (líneas que comienzan por S) puede verse a continuación:

```
S 220 glup.uv.es ESMTP Sendmail 8.12.8/8.12.5; Sun, 18 Feb 2007 18:17:50 +0100
C HELO slabii.uv.es
S 250 glup.uv.es Hello slabii.informat.uv.es [147.156.16.31], pleased to meet you
C MAIL FROM: ebonet@slabii.uv.es
S 250 2.1.0 ebonet@slabii.uv.es... Sender ok
C RCPT TO: quique@glup.uv.es
S 250 2.1.5 quique@glup.uv.es... Recipient ok
C RCPT TO: root@glup.uv.es
S 250 2.1.5 root@glup.uv.es... Recipient ok
C DATA
S 354 Enter mail, end with "." on a line by itself
C Esta es una prueba de correo.
C
C Quique
C .
S 250 2.0.0 h48GHoLN008875 Message accepted for delivery
C MAIL FROM: root@slabii.uv.es
S 250 2.1.0 root@slabii.uv.es... Sender ok
C RCPT TO: quique@glup.uv.es
S 250 2.1.5 quique@glup.uv.es... Recipient ok
C DATA
S 354 Enter mail, end with "." on a line by itself
C Y esta es otra prueba.
C
C Quique
C .
S 250 2.0.0 h48GHoLO008875 Message accepted for delivery
C QUIT
S 221 2.0.0 glup.uv.es closing connection
```

En el ejemplo de diálogo anterior puede verse el establecimiento de una conexión desde un ordenador cliente (slabii.uv.es) con un ordenador servidor (glup.uv.es) y como en la conexión son enviados dos mensajes de correo distintos. El primero de los mensajes es enviado por el usuario ebonet del ordenador slabii.uv.es a dos usuarios (quique y root) del ordenador glup.uv.es. El segundo mensaje es enviado por el usuario root del ordenador slabii.uv.es al usuario quique del ordenador glup.uv.es.

Puede verse en el análisis detenido del diálogo que lo importante de las respuestas del servidor son los códigos numéricos, pues los mensajes que acompañan a

dichos códigos numéricos pueden cambiar. Además, puede comprobarse como es posible mandar más de un correo en una misma conexión y como es posible mandar un correo a varios usuarios especificando repetidamente el comando RCPT TO: para indicar cada uno de los destinatarios del correo.

Aunque el protocolo SMTP está bien definido por el RFC 821, pueden surgir algunos problemas. Así, algunas implementaciones más viejas no pueden manejar mensajes mayores de 64 Kbytes. Otro problema que puede aparecer en ocasiones son las tormentas de correo infinitas. Así, por ejemplo, si dos ordenadores A y B contienen listas de correo que tienen referencias una a la otra, un mensaje de correo enviado a esa lista de correo del ordenador A por el ordenador B (o viceversa), ocasionara que dicho mensaje se envíe de A a B y de B a A sin parar en un bucle infinito de generación de correo.

Para superar algunos de estos problemas, se ha definido el SMTP extendido (Extended SMTP) en el RFC 1425, eliminando la limitación de 64 Kbytes así como resolviendo los problemas de las tormentas de correo infinitas.

El ESMTP incorpora un nuevo comando y dos nuevos códigos de respuesta. El comando nuevo puede verse en la siguiente tabla:

Comando	Argumentos	Descripción
EHLO	<dominio>	Identifica el ordenador remitente del correo al ordenador destinatario.

Mientras que los nuevos códigos numéricos de respuesta son los siguientes:

Código	Descripción
455	El servidor no puede ejecutar temporalmente el comando solicitado. Este código debe ser usado en caso de que no exista otro código que responda “más correctamente” al error producido.
555	El servidor no reconoce o implementa uno o más parámetros del comando solicitado.

Los nuevos códigos pueden aplicarse a los comandos MAIL FROM: y RCPT TO: en caso de que suceda un error.

Un cliente identifica que un servidor es de ESTMP enviando el comando EHLO en lugar del comando HELO para identificarse. Si el servidor envía un código de respuesta 500 (comando no reconocido), el cliente sabe que el servidor es de SMTP, y entonces procede a saludarlo con HELO y establecer de esta forma la comunicación. En caso contrario, esto es, el servidor responde al comando EHLO con un código 250, el cliente sabe que se trata de un servidor de ESTMP. Un ejemplo abreviado de comunicación entre un cliente y un servidor que implementan el ESTMP es el siguiente:

```
220 glup.uv.es ESMTP Sendmail 8.12.8/8.12.5; Sun, 18 Feb 2007 18:48:41 +0100
EHLO slabii.uv.es
250-glup.uv.es Hello slabii.informat.uv.es [147.156.16.31], pleased to meet you
250-ENHANCEDSTATUSCODES
250-PIPELINING
250-8BITMIME
250-SIZE
250-DSN
250-ETRN
250-DELIVERBY
250 HELP
MAIL FROM: ebonet@slabii.uv.es
250 2.1.0 ebonet@slabii.uv.es... Sender ok
RCPT TO: quique@glup.uv.es
```

```
250 2.1.5 quique@glup.uv.es... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Otra prueba de correo.

Quique
.
250 2.0.0 h48GmfLN009320 Message accepted for delivery
QUIT
221 2.0.0 glup.uv.es closing connection
```

Donde como puede verse el servidor (glup.uv.es) ha reconocido el comando EHLO y responde al mismo con el código 250 junto con otras líneas, todas precedidas del código 250 que indican posibilidades, etc., que posee este servidor de ESMTP.

Un detalle final a comentar es que en el ESMTP, las tormentas infinitas de correo han sido solucionadas de una manera “tan sencilla” como es que el correo que envía un usuario a una lista de correo no se envía a ese usuario si es miembro de dicha lista de correo.

Apéndice B: Formato de los mensajes de correo RFC 822 y RFC 1521.

El correo electrónico original propuesto en 1982 consistía no solo en la definición de un protocolo de transmisión (RFC 821), sino que incluía también un formato para los mensajes de correo ASCII. Dicho formato venía descrito en el RFC 822 y aquí nos limitaremos a presentar una breve introducción al mismo.

Los mensajes con formato RFC 822 están formados por una envoltura primitiva (descrita en el RFC 821), algunos campos de cabecera, una línea en blanco, y el cuerpo del mensaje. Cada campo de cabecera consiste en una sola línea de texto ASCII, que contiene el nombre del campo, el símbolo de dos puntos (:) y, para la mayoría de los campos un valor. Los principales campos de cabecera que define el RFC 822 son:

Cabecera	Descripción
To:	Direcciones de correo de los destinatarios primarios.
Cc:	Direcciones de correo de los destinatarios secundarios. En términos de entrega no existe ninguna diferencia con los destinatarios primarios, siendo su diferencia únicamente formal.
Bcc:	Direcciones de correo de las copias al carbón ciegas[25]. En términos de entrega es como los dos campos anteriores, solo que este campo es borrado en las copias enviadas a los destinatarios primarios y secundarios.
From:	Persona o personas que crearon el mensaje.
Sender:	Dirección de correo del remitente. Puede omitirse si es igual al campo anterior.
Receiver:	Línea agregada por cada agente de transferencia en la ruta. La línea contiene la identidad del agente, la fecha y hora de recepción del mensaje. Sirve para conocer la ruta por la que llego un mensaje y detectar fallos en el sistema de enrutamiento del mensaje.
Return-Path:	Puede usarse para identificar una trayectoria de regreso al remitente.
Date:	Fecha y hora de envío del mensaje.
Reply-To:	Indica que la respuesta al mensaje debe ser enviada a la dirección de correo aquí especificada. Puede usarse para que la respuesta sea enviada a una dirección de correo distinta de las especificadas en From y en Sender.
Message-Id:	Identificador único del mensaje para referencias posteriores al mismo.
In-Reply-To:	Indica el identificador único del mensaje al que se esta respondiendo.

[25] Su nombre proviene de cuando para realizar copias en las máquinas de escribir había que introducir un papel carbón entre las hojas. Si dicho papel carbón no se introducía, la página situada en detrás quedaba marcada con la huella de las letras pero con el mismo color que el del papel.

Cabecera	Descripción
References:	Otros identificadores de mensaje.
Keywords:	Palabras clave de mensaje seleccionadas por el usuario.
Subject:	Resumen corto del mensaje para que pueda mostrarse en una sola línea.

El RFC 822 define explícitamente, que los usuarios pueden introducir nuevas cabeceras para su uso privado, bastando que dichas cabeceras comiencen con *X*-, pues el RFC 822 reserva todas las cabeceras que comienzan por *X*- para su uso particular.

En los años 80, el correo en Internet consistía casi exclusivamente en mensajes de texto escritos en inglés y codificados en ASCII. En ese entorno, el RFC 822 resolvía todos los problemas de formato de los mensajes.

Sin embargo, con el crecimiento de la red Internet, el formato de los mensajes de correo descritos en el RFC 822 presentaba grandes problemas debido a sus restricciones, pues empezaron a aparecer mensajes que correspondían a idiomas con acentos (español, francés, alemán, etc.) y símbolos no reconocidos en inglés (la ñ, etc.). Además, surgió la necesidad de poder enviar mensajes de correo en alfabetos no latinos (hebreo, ruso, etc.) e incluso en idiomas que no poseen un alfabeto (chino, japonés, etc.). Por último, se planteo la posibilidad de que el correo electrónico pudiera contener mensajes que no fueran de texto, tales como mensajes de audio y vídeo, fotografías, documentos de texto no ASCII, programas ejecutables, etc. Ante la aparición de estos problemas, se propuso una solución inicial en el RFC 1341, la cual fue actualizada en el RFC 1521.

El RFC 1521 define la extensión multipropósito del correo de Internet (Multipurpose Internet Mail Protocol), la cual es utilizada en la actualidad para el envío de los mensajes de correo.

La idea básica de MIME es continuar usando el RFC 822 como estructura básica del mensaje de correo, pero agregar una nueva estructura al cuerpo del mensaje y definir unas reglas de codificación para los mensajes no ASCII. Al no desviarse del RFC 822, los mensajes MIME pueden enviarse usando los programas y protocolos de correo electrónico existentes[26], siendo necesario tan solo la modificación de los programas que crean y muestran los mensajes de correo. Todo lo que tiene que cambiarse son los programas transmisores y receptores, esto es, los agentes de usuario sin modificar los agentes de transferencia de mensajes.

Para ello, MIME define cinco nuevas cabeceras de mensaje que pueden verse en la siguiente tabla:

Cabecera	Descripción
MIME-Version:	Identifica la versión de MIME utilizada por el correo. Si esta cabecera no existe se considera que es un mensaje tal y como describe el RFC 822.
Content-Description:	Indica el contenido del correo para que el destinatario pueda conocerlo con anterioridad y decidir si, en caso de ir codificado, si desea decodificar o no el mensaje.
Content-Id:	Identificador único del mensaje. Utiliza el mismo formato que el campo Message-Id de la cabecera del mensaje de correo descrita por el RFC 822.

[26] Recordar en este punto que los protocolos de correo usados en la actualidad siguen respondiendo al protocolo SMTP definido en el RFC 821 y su extensión a ESMTP definida en el RFC 1425.

Content-Transfer-Encoding:	Indica el formato de codificación en que ha sido enviado un mensaje de correo.
Content-Type:	Especifica el tipo del cuerpo del mensaje.

De todas estas nuevas cabeceras, las dos que más nos interesan son las dos últimas, esto es, *Content-Transfer-Encoding* y *Content-Type*.

La cabecera *Content-Transfer-Encoding* indica la manera en que está codificado el mensaje para su transmisión a través de una red donde se podría tener problemas con la mayoría de los caracteres distintos de letras, números y signos de puntuación. El RFC 1521 define cinco tipos de codificación de los mensajes. Estos tipos de codificación, conocidos con el nombre de esquemas de codificación son los siguientes:

1. El esquema más sencillo es simplemente texto ASCII. Los caracteres ASCII usan 7 bits y pueden transmitirse directamente mediante el protocolo de correo electrónico siempre y cuando ninguna línea exceda de 1000 caracteres. Se representan mediante *Content-Transfer-Encoding: 7bit*.

2. El siguiente esquema más sencillo es lo mismo, pero usando caracteres de 8 bits, es decir, todos los valores de 0 a 255. Este esquema de codificación viola el protocolo original del correo electrónico de Internet, pero puede ser utilizado siempre y cuando se tenga en cuenta que los caracteres con código ASCII superior a 127 pueden ser distintos según el idioma que se use, por lo que pueden aparecer cambiados. La declaración de esta codificación no elimina esos problemas, pero hacerla explícita puede, cuando menos, explicar el porque algunos caracteres han cambiado en un correo enviado y/o recibido. Los mensajes que usan codificación de 8 bits deben adherirse a la longitud máxima de línea estándar de 1000 caracteres. Se representan mediante *Content-Transfer-Encoding: 8bit*.

3. Los mensajes que usan codificación binaria. Éstos son archivos binarios arbitrarios que no sólo utilizan los 8 bits, sino que ni siquiera respetan el límite de 1000 caracteres por línea. Los programas ejecutables, imágenes, etc., corresponden a esta categoría. MIME no garantiza que estos mensajes lleguen correctamente, es más, no garantiza ni que lleguen, pero hacer explicita su codificación permite conocer los motivos. Se representan mediante *Content-Transfer-Encoding: binary*.

4. La manera correcta de codificar mensajes binarios, imágenes, etc., es usar codificación base64, a veces llamada armadura ASCII. En este esquema de codificación, se dividen grupos de 24 bits en unidades de 6 bits, enviándose cada unidad como un carácter ASCII legal. La codificación es ‘A’ para el valor 0, ‘B’ para el valor 1, etc., seguidas por las 26 letras minúsculas, los 10 dígitos y, por último, + y / para los valores 62 y 63 respectivamente. Las secuencias == y = se usan para indicar que el último grupo contenía solo 8 o 16 bits, respectivamente. Los retornos de carro y avances de línea se ignoran y la longitud de la línea debe mantenerse en 76 bytes, sin contar los bytes de retorno de carro y avance de línea, excepto en la última línea que tendrá la longitud de los bytes codificados restantes. Se representa esta codificación mediante *Content-Transfer-Encoding: base64*.

5. En el caso de mensajes que son casi completamente ASCII, pero con algunos caracteres no ASCII, la codificación base64 es algo ineficiente[27]. En cambio, se puede usar una codificación conocida como codificación entrecomillada-imprimible. Ésta codificación es ASCII de 7 bits, con la todos los caracteres por encima de 127 codificados como un signo de igual seguido del valor del carácter en dos dígitos hexadecimales. En caso de que exista un signo de igual en el mensaje a enviar (valor ASCII 61) este se codifica como los caracteres de código ASCII superior a 127. Este esquema se indica mediante *Content-Transfer-Encoding: quoted-printable*.

La cabecera *Content-Type* especifica la forma del cuerpo del mensaje. Existen siete tipos definidos en el RFC 1521, cada uno de los cuales tiene uno o más subtipos. El tipo y el subtipo se separan mediante el carácter /, de la forma:

```
Content-Type: tipo/subtipo.
```

La lista original de tipos y subtipos especificada en el RFC 1521 puede verse en la tabla siguiente[28]:

Tipo	Subtipo	Descripción
Text	Plain	Texto sin formato.
	Richtext	Texto con comandos de formato sencillos.
Image	Gif	Imagen fija en formato GIF.
	Jpeg	Imagen fija en formato JPEG.
Audio	Basic	Sonido.
Video	Mpeg	Película en formato MPEG.
Application	Octet-stream	Secuencia de bytes no interpretada.
	Postscript	Documento imprimible PostScript.
Message	Rfc822	Mensaje MIME RFC 822.
	Partial	Mensaje dividido para su transmisión.
	External-body	El mensaje mismo debe obtenerse de la red.
Multipart	Mixed	Partes independientes en el orden especificado.
	Alternative	Mismo mensaje en diferentes formatos.
	Parallel	Las partes deben verse simultáneamente.
	Digest	Cada parte es un mensaje RFC 822 completo.

Repasemos la lista de tipos. El tipo *text* es para texto normal. La combinación *text/plain* es para mensajes ordinarios que pueden visualizarse como se reciben, sin necesidad de ningún procesamiento. Esta opción permite el transporte de mensajes ordinarios en formato MIME con sólo unas pocas cabeceras extra.

El subtipo *text/richtext* permite la inclusión de un lenguaje de marcación sencillo en el texto. Este lenguaje proporciona una manera independiente del sistema para indicar negritas, cursivas, tamaños en puntos menores y mayores, sangrías, etc. El lenguaje de marcación se basa en el SGML (Standard Generalized Markup Language), también usado como base del HTML usado en el Web.

[27] Pues sin contar los retornos de carro y avances de línea introducidos convierte cada 3 bytes a enviar en 4 caracteres (4 bytes), por lo cual aumenta el tamaño del mensaje a enviar un 33% como mínimo.
[28] Desde la definición inicial de los tipos y subtipos de MIME se ha agregado muchos nuevos tipos, siendo añadidas nuevas entradas a medida que surge la necesidad.

El siguiente tipo MIME es *image*, que se usa para transmitir imágenes fijas. Hoy día se usan muchos formatos para almacenar y transmitir imágenes, tanto con compresión como sin ella. Dos de estos, *gif* y *jpeg* son los tipos iniciales.

Los tipos *audio* y *video* son para sonido e imágenes en movimiento, respectivamente. Nótese que *video* sólo incluye la información visual, no la pista de sonido. Si se debe transmitir una película con sonido, puede ser necesario transmitir las partes de vídeo y de audio por separado, dependiendo del sistema de codificación usado.

El tipo *application* es un tipo general para los formatos que requieren procesamiento externo no cubierto por ninguno de los otros tipos. Un *octet-stream* simplemente es una secuencia de bytes no interpretados. A la recepción de una de tales secuencias, un agente de usuario debería presentar la secuencia de bytes en la pantalla, sugiriendo al usuario que se copie en un archivo y solicitando un nombre de archivo. El otro subtipo definido es *postscript*, que se refiere al lenguaje PostScript producido por Adobe Systems y usado por muchas páginas impresas. Aunque un agente de usuario puede llamar a un intérprete PostScript externo para visualizar los archivos PostScript entrantes, hacerlo no está exento de riesgos al ser PostScript un lenguaje de programación completo, lo cual permite que alguien, además de exhibir texto, pueda leer, modificar o borrar los archivos del usuario y tener otros efectos secundarios desagradables.

El tipo *message* permite que un mensaje esté encapsulado por completo dentro de otro. Este esquema es útil para reenviar, por ejemplo, correo electrónico. Cuando se encapsula un mensaje RFC 822 completo en un mensaje exterior, debe usarse el subtipo *rfc822*. El subtipo *partial* hace posible dividir un mensaje encapsulado en pedazos y enviarlos por separado. Los parámetros hacen posible ensamblar correctamente todas las partes en el destino. Por último el subtipo *external-body* puede usarse para mensajes muy grandes, por ejemplo películas de vídeo. En lugar de incluir el archivo mpeg en el mensaje, se da una dirección de FTP y el agente de usuario del receptor puede obtenerlo a través de la red cuando se requiera.

El último tipo es *multipart*, que permite que un mensaje contenga más de una parte, con el comienzo y el fin de cada parte claramente delimitados. El subtipo *mixed* permite que cada parte sea diferente. En contraste, el subtipo *alternative* indica que cada parte contiene el mismo mensaje, pero expresado en un medio o codificación diferente. El subtipo *parallel* se usa cuando todas las partes deben “verse” simultáneamente. Por ejemplo, las películas con frecuencia tienen un canal de audio y un canal de vídeo. Por último, el subtipo *digest* se usa cuando se juntan muchos mensajes en un mensaje compuesto.

Veamos un ejemplo de mensaje MIME:

```
Return-Path: <Enrique.Bonet@uv.es>
X-Sieve: cmu-sieve 2.0
Return-Path: <Enrique.Bonet@uv.es>
Received: from glup.uv.es (glup.irobot.uv.es [147.156.222.65])
       by sello.uv.es (8.12.2/8.12.2) with ESMTP id h48HMame032495
       (version=TLSv1/SSLv3 cipher=EDH-RSA-DES-CBC3-SHA bits=192 verify=NOT)
       for <Enrique.Bonet@uv.es>; Thu, 8 May 2003 19:22:36 +0200
Received: from sello.uv.es (sello.ci.uv.es [147.156.1.112])
       by glup.uv.es (8.12.8/8.12.5) with ESMTP id h48HUdLN010033
       for <quique@glup.uv.es>; Thu, 8 May 2003 19:30:39 +0200
Received: from amparo (amparo.irobot.uv.es [147.156.222.34])
```

```
       by sello.uv.es (8.12.2/8.12.2) with SMTP id h48HMZmd032489
       for <quique@glup.uv.es>; Thu, 8 May 2003 19:22:35 +0200
Message-ID: <007a01c31587$18ebca20$22de9c93@uv.es>
From: "Enrique Vte Bonet Esteban" <Enrique.Bonet@uv.es>
To: <quique@glup.irobot.uv.es>
Subject: Mensaje de correo con contenido MIME
Date: Sun, 18 Feb 2007 19:27:27 +0100
MIME-Version: 1.0
Content-Type: multipart/mixed;
       boundary="----=_NextPart_000_0077_01C31597.DC63D140"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Outlook Express 6.00.2800.1158
X-MimeOLE: Produced By Microsoft MimeOLE V6.00.2800.1165

This is a multi-part message in MIME format.

------=_NextPart_000_0077_01C31597.DC63D140
Content-Type: text/plain;
       charset="iso-8859-1"
Content-Transfer-Encoding: 8bit

Este mensaje de correo lleva algunos ficheros adjuntos, etc., para
mostrar la estructura de un mensaje MIME.

============================================
Enrique Vicente Bonet Esteban (Enrique.Bonet@uv.es)
Instituto de Robótica
Universitat de València
Tlf. (34) 96 - 354 35 54
Fax (34) 96 - 354 35 50

------=_NextPart_000_0077_01C31597.DC63D140
Content-Type: image/x-icon;
       name="favicon.ico"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
       filename="favicon.ico"

AAABAAEAICAQAAAAAADoAgAAFgAAACgAAAAgAAAAQAAAAAEABAAAAAAAAgAIAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAACAAACAAAAAgIAAgAAAAIAAgACAgAAAgICAAMDAwAAAAP8AAP8AAAD//wD/AAAA
/wD/AP//AAD///8A////////////////////////////////////////////////////////////
////////zMzP/MzP///MzMz//////////////////////8zMzM/8zM///8zMzMz/////////
////////////////zMzMz/zMz///zMzMzM/////////////////////MzP///MzP/////8zM
z//////////////////////8zP///8zM//////MzP//////////////////////zM////zM
z//////MzM//////////////////////Mz////MzMzMzMzMzM//////////////////////
/8zP///8zMzMzMzMzP//////////////////////zM////zMzMzMzMz/////////////////
//////////Mz////MzP///8zMz//////////////////////8zP///8zM////zMzM//////
////////////////////zM/8zMzMzMzMzMzM//////////////////////MzMz/zMzMzMzMzM
zM//////////////////////8zMzP/MzMzMzMzMzP////////////////////////////////
//////////////8AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAA==

------=_NextPart_000_0077_01C31597.DC63D140
Content-Type: application/octet-stream;
       name="define.h"
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment;
       filename="define.h"

#ifndef __DEFINE_H

#define __DEFINE_H

#define TAM_BUFFER 10000

#endif

------=_NextPart_000_0077_01C31597.DC63D140--
```