ANEXO 9: CORREO

Índice general

Índice de códigos fuente

Capítulo 1

Correo

Contents

TODO: actualizar la configuración de Dovecot 1.2 a Dovecot 2, y de Exim 4.72 a Exim 4.80.

1.1. Arquitectura del servicio de correo

Muy esquemáticamente, recordamos la terminología básica en un sistema de correo electrónico:

Rol	Significado	Funcionalidad	Implem.
MTA	Mail Transport Agent	Son los intercambiadores de correo	Exim4
MDA=LDA	Mail Delivery Agent	Quien entrega el correo en el buzón	Dovecot/Procmail
MUA	Mail User Agent	Programa usado por el usuario final	TB/Mutt
MSA	Mail Submission Agent	Programa que manda el correo al MTA. Suele "ser" también el MUA.	TB/Mutt
IMAP server	servidor IMAP	Permite acceder en remoto al buzón	Dovecot
MRA	Mail Retrieval Agent	Aprovecha el servidor IMAP	TB/Mutt

- En la table, TB se corresponde con el software Mozilla Thunderbird.
- Dovecot es, además del MDA (su proceso "deliver"), el servidor IMAP (su proceso "imap").
- Los intercambiadores de correo son quien se suele llamar habitualmente servidor de correo.

El MUA (Thunderbird) asiste al usuario en la revisión y redacción de un correo, para hacer efectivo el envío el MUA suele lanzar al MSA (suele ser también el MUA) de forma que el correo nuevo se haga llegar al MTA (Exim4). El MTA lo redirigirá o no a otro MTA según sea o no él el destino final, pero cuando lo sea, suele lanzar al MDA (Dovecot deliver) para que entregue el correo. Si el correo ha quedado en un lugar remoto respecto del usuario destinatario, éste utilizará un MRA (suele ser también el MUA) para acceder al servidor IMAP (Dovecot imap) y así al correo.

Otros elementos que aparecerán en este despliegue son los filtros de contenido (anti-spam, anti-virus) o los mecanismos de verificación de remitente. Normalmente, pero no necesariamente, se implementan en el MTA o incluso MDA (como en nuestro tratamiento anti-spam).

1.2. DKLAB1

1.2.1. Instalación del software principal (Exim4, Dovecot)

Vamos a instalar inicialmente la parte principal del software que necesitamos en el servidor; puesto que el sistema de correo es complejo de configurar, requiriendo la edi-

ción y comprobación de numerosos ficheros, instalar ahora éstos hará que los vayamos encontrando disponibles sin más interrupción.

Exim4

```
apt-get install exim4-daemon-heavy pfqueue spfqtool libmail-spf-perl
#Inicialmente instalado y configurado exim4, necesitamos sin embargo
#este paquete sufijado "-heavy", con soporte para consultas ldap etc.
dpkg-reconfigure -plow exim4-config
```

```
 -"General type of mail configuration:"
internet site; mail is sent and received directly using SMTP
 -"System mail name:"
dklab1.casafx.dyndns.org
 -"IP-addresses to listen on for incoming SMTP connections:"
0.0.0.0;0.0.0.0.587;::1;::1.587
 -"Other destinations for which mail is accepted:"
casafx.dyndns.org
 -"Domains to relay mail for:"
<ninguno>
 -"Machines to relay mail for:"
<ninguna ip>
 -"Keep number of DNS-queries minimal (Dial-on-Demand)?"
No
 -"Delivery method for local mail:"
mbox format in /var/mail/
 -"Split configuration into small files?"
No
```

```
invoke-rc.d exim4 stop
mkdir ~/mailtest
cp /etc/init.d/exim4 ~/mailtest/etc-init.d-exim4
```

Dovecot

```
apt-get install dovecot-common dovecot-imapd mutt sieve-connect
```

```
...
Creating generic self-signed certificate: /etc/ssl/certs/dovecot.pem
...
```

```
invoke-rc.d dovecot stop; pkill ssl-build-param
```

Nota: tras la instalación, la ausencia de los parámetros DH provoca que se llame a ssl-build-param para crearlos; si nuestra máquina/emulador es lenta, la creación de los parámetros DH puede tardar mucho tiempo, acaparando intensivamente el tiempo del procesador; si ésto nos resulta molesto en este momento, podemos crear unos parámetros inseguros ahora para evitar la llamada a ssl-build-param, y lidiar con ello más tarde (además y como se verá, en el archivo de configuración se puede indicar cada cuánto renovar esos parámetros DH).

```
cat <<EOF | base64 -d > /var/lib/dovecot/ssl-parameters.dat
AAIAAEgAAAAwRgJBAK6j4/F9kt0TMUBNTMMdj7FnoFwkpla7zda7iUWX6KqxQ1dMfu1zJshTREDz
aJgWr5xSkXxxvRQL5Z3JvRg5QTMCAQIABAAAigAAADCBhwKBgQDQCaM146ukwsw4ZiRNO1R3hzcT
vGoGns1C5fiBGWr9hfl8cYfsj9cExER1Ev+p7aNbk/YBMjwU/HCGH3Q5MZJ0aUtf0ONUos9hKymz
khZWHmTeshfTxw1CxXSPhP2by0Eiz2AnDJ6Zqc0Unz8rT9/Vja7sIpFoompMCGcxdUOIqwIBAgAA
AAA=
EOF
```

1.2.2. Soporte TLS

Justificación

Al enviar correo saliente entre los MTA de nuestro dominio y los de otros ajenos, es TLS el mecanismo disponible para proveer de confidencialidad a la comunicación SMTP. Para las comunicaciones entre nuestros usuarios y los servidores (comunicaciones LMTP, IMAP, MANAGESIEVE), se incluye una fase de autenticación basada en SASL-GSSAPI. El rfc de SASL-GSSAPI especifica la posibilidad del establecimiento de una subsecuente capa de seguridad basada en las credenciales kerberos, haciendo innecesario el uso de TLS aquí. Sin embargo, implementaciones como GSASL (usada por exim4) no la soportan

aún. Por su lado, la nueva implementación creada para dovecot, Dovecot-SASL (usada por dovecot imap, dovecot managesieve y opcionalmente por exim4, es decir todos nuestros servidores), tampoco lo hace: la documentación no lo refiere, pero tras tracear el tráfico se pudo confirmar:

```
# dovecot imap:
tshark -V -i ovpnCASAFX-SRV -d tcp.port==143,imap -R tcp.port==143 -x -S
  ...
  0130  46 72 6f 6d 3a 20 61 65 6d 75 20 63 61 73 61 66   From: aemu casaf
  0140  78 20 75 73 65 72 20 3c 61 65 6d 75 40 63 61 73   x user <aemu@cas
  0150  61 66 78 2e 64 79 6e 64 6e 73 2e 6f 72 67 3e 0d   afx.dyndns.org>.
  0160  0a 54 6f 3a 20 75 6d 65 61 40 63 61 73 61 66 78   .To: umea@casafx
  0170  2e 64 79 6e 64 6e 73 2e 6f 72 67 0d 0a 53 75 62   .dyndns.org..Sub
  0180  6a 65 63 74 3a 20 61 65 6d 75 32 75 6d 65 61 0d   ject: aemu2umea.
  ...

# dovecot managesieve:
tshark -V -i ovpnCASAFX-SRV -R tcp.port==4190 -x -S
  ...
  0030  03 1e 38 09 4c 49 53 54 53 43 52 49 50 54 53 0d   ..8.LISTSCRIPTS.
  ...
  0040  4f 4b 20 22 4c 69 73 74 73 63 72 69 70 74 73 20   OK "Listscripts
  0050  63 6f 6d 70 6c 65 74 65 64 2e 22 0d 0a            completed."..
  ...

# exim4:
tshark -V -i ovpnCASAFX-SRV -d tcp.port==587,smtp -R tcp.port==smtp -x -S
  ...
  0000  53 75 62 6a 65 63 74 3a 20 41 73 75 6e 74 6f 20   Subject: Asunto
  0010  64 65 6c 20 63 6f 72 72 65 6f 0d 0a 46 72 6f 6d   del correo..From
  ...
```

Por lo tanto, TLS es fundamental en todas las comunicaciones.

Creación de los recursos criptográficos

A la vista del script /usr/share/doc/exim4-base/examples/exim-gencert, no es necesario ninguna PMI especial, usamos nuestra CA tal como está. Algo parecido respecto a los servicios proporcionados por dovecot.

```
cd /usr/local/lib/easy-rsa/2.0
bash
export KEY_EMAIL="postmaster@casafx.dyndns.org"
declare -r KEY_EMAIL
source vars
export SAN="DNS:casafx.dyndns.org,
            DNS:dklab1.casafx.dyndns.org,
            DNS:mx1.casafx.dyndns.org"
export SAN="${SAN},
            DNS:dklab2.casafx.dyndns.org,
            DNS:mx2.casafx.dyndns.org"
./build-key-server exim4.casafx.dyndns.org
```

```
...
Enter pass phrase for /usr/local/lib/easy-rsa/2.0/keys/ca.key: asdf
...
```

La razón por la que no creamos un par para exim4 en dklab2 (sin dklab1 en atributo SAN) y otro para dklab1 (sin dklab2 en atributo SAN) es el uso de TKIM (una sóla llave pública para firmar). Podría, alternativamente, tener 3 pares: uno para TKIM, y otros diferentes para dklab2 y dklab1, pero no hay necesidad mientras que así ponemos a prueba el mecanismo de SAN (Subject Alternative Name).

```
export SAN="DNS:casafx.dyndns.org,
            DNS:dklab1.casafx.dyndns.org,
            DNS:imap1.casafx.dyndns.org"
./build-key-server dovecot-1.casafx.dyndns.org

export SAN="DNS:casafx.dyndns.org,
            DNS:dklab2.casafx.dyndns.org,
            DNS:imap2.casafx.dyndns.org"
./build-key-server dovecot-2.casafx.dyndns.org
```

```
unset SAN
exit

ls keys/
cat keys/serial keys/index.txt
openssl req     -text -in keys/exim4.casafx.dyndns.org.csr \
        | ccze -A | less -R
openssl x509    -text -in keys/exim4.casafx.dyndns.org.crt \
        | ccze -A | less -R
openssl rsa     -text -in keys/exim4.casafx.dyndns.org.key \
        | ccze -A | less -R
```

```
cp keys/exim4.casafx.dyndns.org.crt \
   keys/exim4.casafx.dyndns.org.key \
   /etc/exim4/
cp keys/dovecot-1.casafx.dyndns.org.crt \
   keys/dovecot-1.casafx.dyndns.org.key \
   /etc/dovecot/

cd /etc/exim4
chown root:Debian-exim \
      exim4.casafx.dyndns.org.key \
      exim4.casafx.dyndns.org.crt
chmod 640 \
      exim4.casafx.dyndns.org.key \
      exim4.casafx.dyndns.org.crt
#
cd /etc/dovecot
chown root:root \
      dovecot-1.casafx.dyndns.org.key \
      dovecot-1.casafx.dyndns.org.crt
chmod 600 \
      dovecot-1.casafx.dyndns.org.key \
      dovecot-1.casafx.dyndns.org.crt
#

cd ~
```

Tras configurar TLS en exim4, podemos dejar que se creen los parámetros DH a la primera conexión (modo de operar de gnutls, con quien se enlaza exim4, pero que puede ser muy costoso en tiempo), o podemos precomputar una caché /var/spool/exim4/gnutls-params usando: /usr/share/exim4/exim4_refresh_gnutls-params

```
openssl dh -text -in /var/spool/exim4/gnutls-params | ccze -A | less -R
cat /var/spool/exim4/gnutls-params
```

```
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEArYVHPGLKJ8ugLSZhvJXKI7L/AJI+8RvtKkCt9WCntDIwG8ZdUQFP
YNZC5dk4yISA3aw34DS6ihFIhQUm/eLt6/mO34/gNdoWTqRdk1K2+TpwEcHSZ8ir
ynyqqiQGWxoHe9YJyLzeAY+FlYh55JEnx2yiPRQxQCnQYR7e5PxCm0SDqkCVcHwh
VELpKbgsdlM43NZ/RXesBR/o95H21TMbz42LzaV7tqoT9r/iDlrXCHM3XeY4HjbZ
cRp13LaCqORMOsEccVugTE/1RFUIla/CnDSuwKbIBDksEMvbusI61wre5yrd1tIY
3X5z7sMtkaAKfhBqc65svjVRS8FbOy12SwIBAg==
-----END DH PARAMETERS-----
```

En el caso de dovecot, sus parámetros DH se crean (por /usr/lib/dovecot/ssl-build-param) cuando no existe el fichero /var/lib/dovecot/ssl-parameters.dat o cada lo que diga la variable ssl_parameters_regenerate ("0"para deshabilitar) en /etc/dovecot/dovecot.conf.

1.2.3. Soporte para DKIM en DNS

Dos de los subservicios para identificar al remitente de un correo electrónico, SPF y DKIM, precisan registros dedicados en el servidor de nombres de dominio y, en el caso de DKIM, el contenido de esos registros depende de una llave pública que acabamos de generar. Por tanto, no pudimos definir ese registro para DKIM anteriormente cuando configuramos bind9 y lo hacemos ahora:

Averiguamos la llave pública en base64 (desde su llave privada):

```
openssl rsa -pubout -outform PEM \
            -in /etc/exim4/exim4.casafx.dyndns.org.key 2>/dev/null \
        | grep -v ^- | tr -d \\n
```

```
MIGfMA0GCSqGSIb3D...bt9G2ycCz/OcEl8OCDiYWDelXV/gzGHOfUkphsrVL4jiSQIDAQAB
```

Habilitamos el registros TXT "selector" que anuncia la llave pública. Veamos qué contiene en este momento el fichero de configuración, respecto a DKIM:

```
view /etc/bind/db.casafx.dyndns.org-VPN
```

```
...
                    ; a) DKIM selector record
;exim4dkim._domainkey.casafx.dyndns.org.     IN TXT "v=DKIM1; s=email; g=*; t=y:s; k=rs
                    ; b) policy record (old OSP and new ADSP)
_domainkey.casafx.dyndns.org.           IN TXT "t=y; o=~;"
_adsp._domainkey.casafx.dyndns.org.     IN TXT "dkim=unknown"
...
```

La segunda línea está comentada (precedida por ";"), debemos entonces descomentarla y sustituir <public key>, tras 'p=', por la llave pública.

```
vim /etc/bind/db.casafx.dyndns.org-VPN
```

```
invoke-rc.d bind9 restart
```

Si todo ha ido bien, debería poder ser recuperada con:

```
dig @dklab1.casafx.dyndns.org exim4dkim._domainkey.casafx.dyndns.org. TXT +short
```

1.2.4. Soporte para centralización de metadatos: LDAP

Esquema qmail-ldap en database cn=config de dklab1 y dklab2

```
cd /tmp
wget http://www.nrg4u.com/qmail/qmail-ldap-1.03-20060201.patch.gz
gunzip qmail-ldap-1.03-20060201.patch.gz
patch -p0 -f < qmail-ldap-1.03-20060201.patch 2>&1 >/dev/null
cp qmail-ldap/qmail.schema ~/ldif
cd ~
```

Este esquema usa un subespacio del espacio de identificadores OID 1.3.6.1.4.1.7914 que la IANA ha asignado a la organización "qmail-ldap project" (http://www.oid-info.com/cgi-bin/display?nb%3D7914&father_oid%3D1.3.6.1.4.1&action%3Ddisplayref, véase la jerarquía de la que depende junto a información de registro http://www.alvestrand.no/objectid/1.3.6.1.4

Nosotros no usaremos pues esta rama de OID para no publicar modificaciones en su nombre pero, otro espacio (el 1.3.6.1.3), está reservado (http://www.openldap.org/lists/openldap-software/200301/msg00412.html->http://tools.ietf.org/html/draft-ietf-ldapbis-iana-06) para pruebas privadas (a grosso modo, algo parecido al conocido subespacio 192.168.1 de direcciones IP, usables pero no "publicables"). Así, a falta de un OID registrado, nuestras modificaciones usarán un subespacio de ése, por ejemplo 1.3.6.1.3.1.3.6.1.4.1.7914

```
cp ~/ldif/qmail.schema  ~/ldif/qmail.schema.orig
vim ~/ldif/qmail.schema
```

```
...
attributetype ( 1.3.6.1.4.1.7914.1.4.1.10 NAME 'qlaMailHostList'
        DESC ''
        EQUALITY caseIgnoreIA5Match
        SUBSTR caseIgnoreIA5SubstringsMatch
        SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )

####fx:
#Acorde con el manual, el requisito del atributo para las
#preferencias de usuario spamassassing es ser texto multivaluado.
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.1
   NAME 'spamassassinUserPrefs'
   DESC 'SpamAssassin user preferences'
   EQUALITY caseIgnoreIA5Match
   SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )
####endfx
```

```
# Object Class Definitions

####fx:
#-objectclass ( 1.3.6.1.4.1.7914.1.2.2.1 NAME 'qmailUser'
#-      DESC 'QMail-LDAP User'
#-      SUP top
#-      AUXILIARY
#-      MUST ( mail )
#-      MAY ( uid $ mailMessageStore $ homeDirectory $ userPassword $
#-            mailAlternateAddress $ qmailUID $ qmailGID $
#-            mailHost $ mailForwardingAddress $ deliveryProgramPath $
#-            qmailDotMode $ deliveryMode $ mailReplyText $
#-            accountStatus $ qmailAccountPurge $
#-            mailQuotaSize $ mailQuotaCount $ mailSizeMax ) )
#Redefinimos el objeto qmailUser para incluir el atributo spamassassin
objectclass ( 1.3.6.1.3.1.3.6.1.4.1.7914.101 NAME 'qmailUser'
        DESC 'QMail-LDAP User'
        SUP top
        AUXILIARY
        MUST ( mail )
        MAY ( uid $ mailMessageStore $ homeDirectory $ userPassword $
              mailAlternateAddress $ qmailUID $ qmailGID $
              mailHost $ mailForwardingAddress $ deliveryProgramPath $
              qmailDotMode $ deliveryMode $ mailReplyText $
              accountStatus $ qmailAccountPurge $
              mailQuotaSize $ mailQuotaCount $ mailSizeMax $
              spamassassinUserPrefs) )
####endfx
```

```
...
####fx:
#Soporte para utilizar un smarthost condicional ante determinados dominios
#destino. Contaremos con un objeto y sus 7 atributos que definimos previamente:
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.2
    NAME 'domainMakesUsUseSmarthost'
    DESC 'domain needs a recognized mail provider to receive mail from us'
    EQUALITY caseIgnoreIA5Match
    SUBSTR caseIgnoreIA5SubstringsMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.3
    NAME 'smarthostLoginName'
    DESC 'Account'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256}
    SINGLE-VALUE )
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.4
    NAME 'smarthostPass'
    DESC 'Pass'
    EQUALITY octetStringMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.40
    SINGLE-VALUE )
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.5
    NAME 'smarthostName'
    DESC 'Hostname'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256} )
```

```
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.6
    NAME 'smarthostPort'
    DESC 'Port, although this information can be defined in hostnameRule'
    EQUALITY integerMatch
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
    SINGLE-VALUE )
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.7
    NAME 'smarthostRoute'
    DESC 'Thought to be a route_data list according exim4 manualroute'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256}
    SINGLE-VALUE )
attributetype ( 1.3.6.1.3.1.3.6.1.4.1.7914.1.8
    NAME 'smarthostID'
    DESC 'Identificator name'
    EQUALITY caseIgnoreIA5Match
    SYNTAX 1.3.6.1.4.1.1466.115.121.1.26{256}
    SINGLE-VALUE )
```

```
objectclass ( 1.3.6.1.3.1.3.6.1.4.1.7914.102 NAME 'smartHost'
        DESC 'Smarthost Data, for exim4 manualroute route driver'
        SUP top
        AUXILIARY
        MUST ( smarthostID $ smarthostName $ smarthostRoute $
               domainMakesUsUseSmarthost )
        MAY ( smarthostLoginName $ smarthostPass $ smarthostPort ) )
#...no se define STRUCTURAL para poder ser usado "per user".
####endfx
```

```
cp ~/ldif/qmail.schema /etc/ldap/schema/qmail.schema
mkdir ~/ldif/mail_schema/
#
echo "include /etc/ldap/schema/core.schema" \
     > ~/ldif/mail_schema/schema_convert.conf
echo "include /etc/ldap/schema/cosine.schema" \
     >> ~/ldif/mail_schema/schema_convert.conf
echo "include /etc/ldap/schema/nis.schema" \
     >> ~/ldif/mail_schema/schema_convert.conf
echo "include /etc/ldap/schema/qmail.schema" \
     >> ~/ldif/mail_schema/schema_convert.conf
#
slapcat -f ~/ldif/mail_schema/schema_convert.conf \
        -F ~/ldif/mail_schema/ -s "cn=qmail,cn=schema,cn=config"
#Finalmente hay que hacer algunos cambios sobre el qmail.ldif generado,
#en general modifican la cabeza y la cola del fichero eliminando
#timestamps e identificadores. Automatizamos con sed:
sed -i'.pre_sed.BAK' \
    -e s/'dn: cn={3}qmail'/'dn: cn=qmail,cn=schema,cn=config'/g \
    -e s/'cn: {3}qmail'/'cn: qmail'/g \
    -n -e '/structuralObjectClass: olcSchemaConfig/,$!p' ~/ldif/qmail.ldif
```

Ya podemos cargar el schema:

```
ldapmodify -a -QY EXTERNAL -H ldapi:/// -f ~/ldif/qmail.ldif
```

```
 adding new entry "cn=qmail,cn=schema,cn=config"
```

```
ldapsearch -LLLQY EXTERNAL -H ldapi:/// -b cn=config cn={5}qmail
```

Podemos añadir el esquema también en dklab2 puesto que las bases de datos cn=config no se replican entre dklab1 y dklab2 pero sí dc=casafx,dc=dyndns,dc=org que vamos a

usar a continuación para añadir los objetos y atributos del esquema a sus entidades de la rama ou=users.

```
scp ~/ldif/qmail.ldif dklab2.casafx.dyndns.org:/root/ldif/qmail.ldif

ssh dklab2.casafx.dyndns.org \

    ldapmodify -a -QY EXTERNAL -H ldapi:/// -f /root/ldif/qmail.ldif
```

Uso del esquema qmail-ldap

"Vacation text":

```
(echo '

Thank you for your email.

I am out of the office, I will respond upon my return.


Regards.'

 echo

 echo '---'

 echo '

Gracias por su correo.

Estare lejos de la oficina, pero procurare responder a mi regreso.


Saludos.'

 echo)\

 > /root/ldif/generic_vacation.txt

#nota: "estare" y procurare" debieran llevar tilde en la vocal final.
```

```
vim ~/ldif/umea+mail_attributes.ldif
```

```
dn: uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
add: objectClass
objectClass: qmailUser
-
add: accountStatus
accountStatus: active
-
add: mailSizeMax
mailSizeMax: 10000
-
add: MailQuotaSize
mailQuotaSize: 5000
-
add: mailQuotaCount
mailQuotaCount: 0
-
add: spamassassinUserPrefs
spamassassinUserPrefs: required_score 5.00
spamassassinUserPrefs: use_bayes 1
-
```

```
add: mailAlternateAddress
mailAlternateAddress: jabbermaster@casafx.dyndns.org
mailAlternateAddress: vpnmaster@casafx.dyndns.org
mailAlternateAddress: camaster@casafx.dyndns.org
mailAlternateAddress: postmaster@casafx.dyndns.org
-
add: mailForwardingAddress
mailForwardingAddress: aemu@casafx.dyndns.org
-
add: deliveryProgramPath
deliveryProgramPath: |/usr/bin/procmail
-
add: mailMessageStore
mailMessageStore: Maildir
-
add: mailReplyText
mailReplyText:< file:///root/ldif/generic_vacation.txt
```

```
kinit -p root/admin
ldapmodify -Qf ~/ldif/umea+mail_attributes.ldif
ldapsearch -LLL \
           -b "uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org" \
           -s base |less
```

Un ejemplo de modificación a posteriori podría ser:

```
vim ~/ldif/umea+mail_attributes-delete_deliveryMode_noforward.ldif
```

```
dn: uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
add: deliveryMode
deliveryMode: noforward
```

```
ldapmodify -Qf ~/ldif/umea+mail_attributes-delete_deliveryMode_noforward.ldif
```

Vamos con aemu:

```
vim ~/ldif/aemu+mail_attributes.ldif
```

```
dn: uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
add: objectClass
objectClass: qmailUser
-
add: accountStatus
accountStatus: active
-
add: mailSizeMax
mailSizeMax: 10000
-
add: MailQuotaSize
mailQuotaSize: 5000
-
add: mailQuotaCount
mailQuotaCount: 0
-
add: spamassassinUserPrefs
spamassassinUserPrefs: required_score 5.00
spamassassinUserPrefs: use_bayes 1
-
```

```
add: mailAlternateAddress
mailAlternateAddress: postmaster@casafx.dyndns.org
-
add: deliveryProgramPath
deliveryProgramPath: |/usr/bin/procmail
-
add: deliveryMode
deliveryMode: noforward
deliveryMode: noprogram
-
add: mailMessageStore
mailMessageStore: Maildir
-
add: mailReplyText
mailReplyText:< file:///root/ldif/generic_vacation.txt
```

```
ldapmodify -Qf ~/ldif/aemu+mail_attributes.ldif
ldapsearch -LLL \
           -b "uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org" \
           -s base |less
```

1.2.5. Soporte para autenticación KERBEROS y almacenamiento en AFS

```
kadmin.local
 >addprinc -policy service -randkey smtp/dklab1.casafx.dyndns.org
 >addprinc -policy service -randkey imap/dklab1.casafx.dyndns.org
 >addprinc -policy service -randkey sieve/dklab1.casafx.dyndns.org
 >ktadd -k /etc/keytab.d/exim4.keytab -norandkey smtp/dklab1.casafx.dyndns.org
 >ktadd -k /etc/keytab.d/dovecot.keytab -norandkey imap/dklab1.casafx.dyndns.org
 >ktadd -k /etc/keytab.d/dovecot.keytab -norandkey smtp/dklab1.casafx.dyndns.org
 >ktadd -k /etc/keytab.d/dovecot.keytab -norandkey sieve/dklab1.casafx.dyndns.org
 >quit

 #Seguidamente se justifican los principals y sus exportaciones.
```

Uso de los principal: servicios kerberizados

Es conveniente aclarar cuál y cómo será el uso de estos principal kerberos. Dovecot (su proceso hijo dovecot-auth) necesita unos principal exportados en un keytab para autenticarse ante el KDC y permitir así que los clientes IMAP y MANAGESIEVE se autentifiquen usando sus ticket de servicio. Los principal requeridos son imap/dklab1.casafx.dyndns.org y sieve/dklab1.casafx.dyndns.org, Ambos están exportados en /etc/keytab.d/dovecot.keytab y, puesto que el proceso dovecot-auth se ejecuta como root, el keytab basta con que pueda ser leído por root.

Nota: en lo que sigue hablaremos sólo de imap, pues la situación de managesieve es prácticamente la misma.

Por su lado, exim4 tiene la misma necesidad para SMTP-AUTH pero tiene la particularidad de que es un proceso que se ejecuta como Debian-exim, por tanto: su principal es smtp/dklab1.casafx.dyndns.org, está exportado en /etc/keytab.d/exim4.keytab y éste debe poder leerse sólo por la entidad Debian-exim. Ahora bien, siendo ésto previsible, hemos optado por otro esquema: exim4 puede delegar la autenticación SMTP-AUTH a dovecot-auth y, si bien quizás sea menos flexible, es suficiente y ventajoso en el sentido de tener esa parte de la configuración en un sólo lugar. Puesto que es entonces el proceso dovecot-auth (que se ejecutaba, recordemos, como root) el que tiene que usar el principal smtp/dklab1.casafx.dyndns.org, y puesto que dovecot-auth entiende que su keytab es /etc/keytab.d/dovecot.keytab, entonces exportaremos ese principal en ese keytab legi-

ble por root donde ya residía previamente la exportación del principal para imap, como dijimos. Conclusión pues: todos los principals a /etc/keytab.d/dovecot.keytab.

Es claro que, si tomamos este segundo camino al configurar SMTP-AUTH, sólo el dovecot.keytab es usado: el exim4.keytab pues, puede eliminarse según ésto y según otro argumento que daremos a continuación.

Nota: parece que dovecot-auth exige que el nombre de los principal sean los usados smtp/<instance>, imap/<instance>, sieve/<instance>(los registrados por IANA).

```
chown root:root /etc/keytab.d/dovecot.keytab

chmod go-rw /etc/keytab.d/dovecot.keytab

su root -m -c 'klist -t -K -k /etc/keytab.d/dovecot.keytab'


#


chown Debian-exim:root /etc/keytab.d/exim4.keytab

chmod go-rw /etc/keytab.d/exim4.keytab

su Debian-exim -m -c 'klist -t -K -k /etc/keytab.d/exim4.keytab'

#... o, como hemos argumentado, podemos simplemente eliminar exim4.keytab.
```

Uso de los principal: cliente kerberos

El uso de los principal kerberos no acaba aquí. Hasta ahora (dovecot-auth) hace el típico uso de los principal como SERVICIO KERBERIZADO; sin embargo, también debemos desplegar su uso como CLIENTE KERBEROS. ¿Por qué? Exim4 lanzará a los programas "deliver" y "procmail" y, por su lado, dovecot lanzará a los programas "imap" y "managesieve"; la finalidad de todos esos programas es manipular las mailboxes en AFS y, así, los principal anteriores serán necesarios para conseguir las credenciales AFS: primero un TGT kerberos, con éste un ticket del servicio AFS y, satisfaciendo el esquema krb5 de autentificación de openafs, por fin el token AFS objetivo. Puesto que esos programas no son interactivos, el keytab donde los principals están exportados vuelve, también, a entrar en juego (ahora para conseguir un TGT, actuar de cliente KERBEROS pues).

Dicho de forma sencilla: ésta es la primera vez que necesitamos que un servidor acceda a AFS. Además, la no interactividad de los servidores y las peculiaridades de la caché de credenciales en AFS hace que debamos profundizar algo más en el sistema de autorización AFS para comprender esta parte:

En kerberos, los tickets TGT expedidos son almacenados en /tmp, y la condición que permite usarlos la impone el poder relacionarse con esa hoja del árbol de ficheros (según los permisos unix, el uid/gid del proceso que intenta acceder a ellos, etc).

En afs, el token expedido se almacena en memoria por el cache manager (el módulo openafs), y la condición que permite usarlo la impone el poder relacionarse también según un árbol, pero no el de ficheros sino el de procesos: asociada a la expedición del token, el cache manager crea una PAG (process authentication group), de forma que el proceso que hace la petición y sus procesos hijos se encuentran registrados en esa PAG y pueden usar el token.

Así, exim4 o dovecot ha de ser llamado por el proceso que pide el token para pertenecer a su PAG, y entonces sus procesos hijo "delivery" y "procmail" (caso de exim4), o "imap" y "managesieve" (caso de dovecot) también pertenecerán al PAG y podrán usar el token.

¿Por qué es importante todo ésto? Porque invalida el esquema de utilizar a k5start desde /etc/inittab. La necesidad de usar k5start se daba cuando un proceso necesitaba indefinidamente hacer de cliente kerberos (usar un TGT), ya que las credenciales kerberos caducaban en 24h. Era, entonces, k5start el que se las renovaba puntualmente y, además, aseguraba las relaciones según el árbol de ficheros, pues dejaba los tickets en /tmp/krb5cc_<uid>con unos permisos que lo hiciesen legible para el proceso que lo necesitaba. Pero, como hemos explicado, éso no asegura las relaciones según el árbol de procesos que son las que nos permiten usar las credenciales AFS que son objetivo ahora.

El nuevo esquema que garantiza que un servidor pueda acceder a AFS será: modificaremos a /etc/init.d/exim4 (caso de exim4) y /etc/init.d/dovecot (caso de dovecot) para que llamen a k5start, éste conseguirá como siempre las credenciales pero, adicionalmente, lanzará cual proceso padre a los servidores ("exim4" en el caso del k5start en /etc/init.d/exim4, y "dovecot" en el caso del k5start en /etc/init.d/dovecot).

Pongamos el ejemplo 1, susceptible de ser introducido en el /etc/init.d/dovecot:

```
KRB5CCNAME=MEMORY: \
/usr/bin/k5start -b -t -u imap -f /etc/keytab.d/dovecot.keytab -K 10 -l 24h \
                 -- /usr/sbin/dovecot -F -c /etc/dovecot/dovecot.conf
```

Código fuente 1: Ejemplo de k5start que obtiene credenciales AFS y lanza un servicio como dovecot.

- La primera línea indicará a k5start (específicamente a la librería mit kerberos que usa éste) que la caché de credenciales kerberos se almacene en memoria: ya que nadie más las necesitará, no hay razón para hacerlas accesible en el árbol de archivos (en /tmp/krb5cc_<uid>).
- La segunda línea hace que se consiga el TGT utilizando del keytab /etc/keytab.d/dovecot.keytab el principal cuyo primario es "imap". Gracias a -t, subsecuentemente pedirá un token AFS utilizando el TGT.
- La tercera línea hace que k5start lance dovecot, posibilitando que los procesos hijos de dovecot (imap y managesieve) puedan utilizar el token AFS (que no el TGT previo, el TGT les es indiferente - incluso aunque sin él no se hubiera conseguido el token - utilizarán sólo el token).

Un detalle importante es que a k5start le hemos pedido que utilice el principal imap/dklab1.casafx.dyn éste se mapeará a la identidad AFS imap.dklab1 y se usará cuando dovecot lance al proceso "imap". Sin embargo dovecot también lanza al proceso managesieve luego, ¿debería también usar el principal sieve/dklab1.casafx.dyndns.org y una identidad AFS sieve.dklab1? No; nos da igual el principal que usar siempre que pueda mapearse a una identidad AFS que nos otorgue permisos sobre las mailboxes. Efectivamente nos da igual porque, al contrario por ejemplo de lo que pasa cuando los principal se usan para el rol de servicio kerberizado, no necesitamos que el principal diferencie la identidad del proceso que lo usa. De hecho nos resultará más sencillo el despliegue si tanto dovecot (y por tanto imap y managesieve) como exim4 (y por tanto deliver y procmail), utilizan el mismo principal en su rol de cliente KERBEROS.

Los scripts bajo /etc/init.d/ se ejecutan como root, luego tanto el k5start en /etc/init.d/dovecot como el k5start en /etc/init.d/exim4 tienen asegurado el acceso de lectura al keytab que van a usar.

Expuesto todo lo anterior, éste es el esquema que seguir para el rol de cliente:

- el script /etc/init.d/dovecot creará con ayuda de k5start una PAG usando el principal imap/dklab1.casafx.dyndns.org almacenado en el dovecot.keytab, por tanto el token AFS otorga la identidad AFS "imap.dklab1" a los procesos imap y managesieve.
- el script /etc/init.d/exim4 creará con ayuda de k5start una PAG usando el principal imap/dklab1.casafx.dyndns.org almacenado en dovecot.keytab, por tanto el token AFS otorga la identidad AFS "imap.dklab1" a los procesos deliver y procmail.
- las mailboxes en AFS tendrán acl permisivas para el grupo AFS "mailgroup", que crearemos y en el que incluiremos a la identidad afs "imap.dklab1". De esta forma si en un futuro nos conviene crear alguna otra identidad AFS relacionada con el sistema de correo, bastaría con hacerla miembro del grupo "mailgroup" para que tenga permisos sobre las mailboxes etc. Tanto el grupo "mailgroup" como la identidad "imap.dklab1" tenemos que crearlas aún.

Algunas anotaciones:

- Ninguna de estas decisiones afecta a las que se tomaron para desarrollar el rol de servicio kerberizado.
- K5start renovará puntualmente, como siempre, todas las credenciales.
- Puesto que el principal imap/dklab1.casafx.dyndns.org es el primero exportado en dovecot.keytab, podemos pasar -U, en lugar de -u <principal>, a k5start. Puede comprobarse con:

-
```
klist -k /etc/keytab.d/dovecot.keytab
```

- KRB5CCNAME=MEMORY: no será usado: man k5start recomienda indicarlo usando esa variable (y omitiendo -k), pero no parece tener efecto y siguen guardándose como un fichero bajo "/tmp", desconocemos por qué. Mientras no se investigue ésto, optamos por mantener dos tickets TGT (sin uso alguno tras conseguir el token AFS) en /tmp pasando las opciones

  ```
  -k <file> -o <uid>
  ```

  de k5start. Puede comprobarse con:

-
```
env KRB5CCNAME=MEMORY: /usr/bin/k5start -t -U -f /etc/keytab.d/dovecot.keytab\
        -K 10 -l 24h -- cat /proc/self/environ \
        | xargs --null --max-args=1 echo|grep KRB
#...devuelve KRB5CCNAME=/tmp/krb5cc_0_9xwRPj
env KRB5CCNAME=MEMORY: /usr/bin/k5start -t -U -f /etc/keytab.d/dovecot.keytab\
    -K 10 -l 24h -- klist | grep Ticket
#...devuelve Ticket cache: FILE:/tmp/krb5cc_0_HKbC3m
#http://newsgroups.derkeiler.com/Archive/Comp/comp.protocols.kerberos/\
#                                                2010-10/msg00063.html
```

Ejecución de las modificaciones recién aclaradas

K5start en los scripts de inicio. El resultados será algo críptico porque tenemos que hacer un trabajo de integración, pero la llamada a k5start es básicamente la que se ha explicado en 1. Sólo modificamos la función "start" de cada script, así que si queremos reiniciar un servicio es más seguro hacer "invoke-rc.d <servicio>stop" && "invoke-rc.d <servicio>start" que, directamente, "invoke-rc.d <servicio>restart".

```
vim /etc/init.d/dovecot
```

```
...
do_start()
{
####fx:
    local DAEMON='/usr/bin/k5start'
    local CONF="${PIDFILE} -b -t -U -f /etc/keytab.d/dovecot.keytab -K 10 -l 24h
                -k /tmp/.krb5cc_`id -u vmail`_dovecot -o vmail
                -- dovecot -F -c ${CONF}"
#${PIDFILE} (en verdad -c ${PIDFILE} porque le va a preceder -c al ser
# evaluada la variable CONF) hace que escriba el pid de dovecot en ese fichero.
#-b hace que k5start se ejecute en background
#-t es la responsable de que se pida token afs en una nueva PAG, que va
# a heredar dovecot y que va a ser distinta a la PAG y token de exim4.
#-o vmail hace que el TGT kerberos sea accesible solamente para el uid vmail.
#-k ${KRB5CCNAME} implica que no puede sobreescribir los tickets
# pedidos por el k5start de exim4, pues la cache tiene distinto nombre.
#NOTA: KRB5CCNAME=MEMORY: no dio resultado, por ello el uso de
#      -k <file> -o <uid>
#AFS es necesario para dovecot:
#    -al lanzar imap (con uid=vmail)
#    -al lanzar managesieve (cou uid=vmail).
####endfx
...
```

```
vim /etc/init.d/exim4
```

```
...
start_exim()
{
####fx:
local K5START__="/usr/bin/k5start -b -c ${PIDFILE}
                 -t -U -f /etc/keytab.d/dovecot.keytab -K 10 -l 24h
                 -k /tmp/.krb5cc_‘id -u vmail‘_exim4 -o vmail
                 -- "
      start_daemon -p  "$PIDFILE"  ${K5START__} \
        "$DAEMON" -bdf "-q${QFLAGS}${QUEUEINTERVAL}" \
        ${COMMONOPTIONS} \
        ${QUEUERUNNEROPTIONS} \
        ${SMTPLISTENEROPTIONS}
#-    start_daemon -p "$PIDFILE" \
#-      "$DAEMON" -bd "-q${QFLAGS}${QUEUEINTERVAL}" \
#-      ${COMMONOPTIONS} \
#-      ${QUEUERUNNEROPTIONS} \
#-      ${SMTPLISTENEROPTIONS}
#NOTA: KRB5CCNAME=MEMORY: no dio resultado, por ello el uso
#      de -k <file> -o <uid>
#AFS es necesario para exim4:
#    -en la entrega, al lanzar a deliver, con uid=vmail
#    -en la entrega opcional con procmail, cou uid=vmail
####endfx
...
```

Cuenta posix con que lanzar los procesos hijos (deliver, procmail; imap, managesieve):

```
getent group 800
getent passwd 800
addgroup --system --gid 800 vmail
adduser --system --no-create-home --disabled-login --uid 800 --gid 800 vmail
```

```
Adding system user ‘vmail’ (UID 800) ...
Adding new user ‘vmail’ (UID 800) with group ‘vmail’ ...
Not creating home directory ‘/home/vmail’.
#--system hace que no tenga shell (/bin/false).
#adduser vmail mail no es necesario (ni siquiera para dovecot), aunque posible.
```

Creación de la identidad AFS "imap.dklab1":

```
kinit root/admin; aklog


pts listentries -users
```

Debemos elegir un nombre que cumpla dos condiciones: ser válido para openAFS, ser mapeable desde el principal Kerberos. Citando al manual: http://docs.openafs.org/AdminGuide/ch13

Válido “User’s username (the character string typed at login): It is best to limit the name to eight or fewer lowercase letters, because many application programs impose that limit. The AFS servers themselves accept names of up to 63 lowercase letters. Also avoid the following characters: colon (:), semicolon (;), comma (,), at sign (@), space, newline, and the period (.), which is conventionally used only in special administrative names”.

Mapeable “When using aklog, be aware that AFS uses the Kerberos v4 principal naming format, not the Kerberos v5 format, when referring to principals in PTS ACLs, UserList, and similar locations. AFS will internally map Kerberos v5 principal names to the Kerberos v4 syntax by removing any portion of the instance after the first period (generally the domain name of a host principal), changing any ’/’to ’.’, and changing an initial principal part of ’host’to ’rcmd’. In other words, to create a PTS entry for the Kerberos v5 principal úser/admin’, refer to it as úser.admin’, and for the principal ’host/shell.example.com’, refer to it as ’rcmd.shell”’.

Por tanto el principal “imap/dklab1.casafx.dyndns.org”pasa al afs id “imap.dklab1”:

```
pts createuser imap.dklab1 -id 30000     # desde imap/dklab1.casafx.dyndns.org
pts listentries -users

pts listentries -groups
pts creategroup mailgroup  #al ser admin no necesito owner de prefijo, algo
                           #como imap/dklab1.casafx.dyndns.org:mailgroup
pts listentries -groups

pts adduser imap.dklab1 -group mailgroup # imap/dklab1.casafx.dyndns.org:mailgroup
pts membership mailgroup                 # users que esta'n en ese grupo?
pts membership imap.dklab1               # grupos en que esta' ese user?
```

1.2.6. Soporte antivirus con Clamav

```
apt-get install clamav-daemon clamav-freshclam
```

```
 -"Virus database update method:"
daemon
 -"Local database mirror site:"
db.local.clamav.net (u otro apropiado geograficamente).
 -"HTTP proxy information (leave blank for none):"
<nada>
 -"Number of freshclam updates per day:"
24 (si bien durante las pruebas puede ser menos molesto un valor como 1)
 -"Should clamd be notified after updates?"
Yes
```

```
view /usr/share/doc/clamav-base/README.Debian.gz
```

Según las instrucciones del README.Debian, además de declarar su uso en exim4 como luego veremos, debemos hacer que los recursos de uno sean accesibles por el otro.

Nos ocupamos de ésto último aquí:

```
ls -l /var/run/clamav/clamd.ctl
```

```
srw-rw-rw- 1 clamav clamav 0 2011-10-28 12:32 /var/run/clamav/clamd.ctl
```

Ok, el socket unix de clamd es legible/escribible por cualquiera. Vamos con el directorio de spooling:

```
ls -ld /var/spool/exim4/scan
```

```
drwxr-x--- 2 Debian-exim Debian-exim
```

En ese directorio debe acceder rw clamd, el cual se ejecuta como la entidad clamav; la solución es añadir clamav al grupo Debian-exim, permitir ese esquema en la configuración de clamav (así viene por defecto) y dar permisos "w" de escritura para el grupo a ese directorio.

```
adduser clamav Debian-exim

mkdir -p /var/spool/exim4/scan; chmod g+w /var/spool/exim4/scan

grep ^AllowSupplementaryGroups /etc/clamav/clamd.conf
```

```
AllowSupplementaryGroups true
```

```
invoke-rc.d clamav-daemon restart
```

Test

```
echo 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*' \
 | su Debian-exim -m -c 'clamdscan -v -'
```

```
stream: Eicar-Test-Signature FOUND

----------- SCAN SUMMARY -----------

 Infected files: 1
```

1.2.7. Soporte antispam con Spamassassin

```
apt-get install spamassassin libnet-ldap-perl
vim /etc/spamassassin/local.cf
```

```
...
####fx:
include /etc/spamassassin/local-fx-01.cf
####endfx
```

```
vim /etc/spamassassin/local-fx-01.cf
```

(La sentencia definiendo user_scores aparece dividida en 4 partes delimitadas por \ y cambio de línea. Realmente debe aparecer en una sóla línea, sin \ y cambio de línea adicionales:)

```
####fx:
user_scores_dsn ldap://dklab1.casafx.dyndns.org/\
ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?\
spamassassinUserPrefs?sub?\
(&(uid=__USERNAME__)(objectClass=qmailUser))
#...en una so'la li'nea, recordamos.

#Desafortunadamente, no parece que spamd pueda usar registros SRV para
#descubrir los servidores LDAP, ni permite declarar varias
#variables user_scores_dsn ni otras manipulaciones. Por tanto, esta
#instancia de spamd usa al servidor dklab1.casafx.dyndns.org y a la
#db dc=casafx,dc=dyndns,dc=org.
#Se recomienda consultar /usr/share/doc/spamassassin/README.ldap.gz

#user_scores_ldap_username       uid=exim,dc=example,dc=com
#user_scores_ldap_password       pass
 #http://cpansearch.perl.org/src/FELICITY/Mail-SpamAssassin-3.0.0/ldap/README

clear_headers
add_header all Flag _YESNO_
add_header spam Result _SCORE_/_REQD_ (_TESTS_)
#
# Nota: Por defecto, escucha en ip 127.0.0.1 tcp 783.
####endfx
```

```
vim /etc/default/spamassassin
```

```
...
# Change to one to enable spamd
####fx:
#-ENABLED=0
ENABLED=1
####endfx
...
####fx:
#-OPTIONS="--create-prefs --max-children 5 --helper-home-dir"
OPTIONS="-x --ldap-config -u nobody --max-children 5"
#-x             disable user-config files
#--ldap-config  enable ldap-config
#-u nobody      para lanzarse como uid nobody en lugar de root, pues no es necesari
#                acceder a preferencias en el home del usuario.
####endfx
```

Test

Para comprobar que funciona, utilizando un correo ejemplo de spam:

```
spamassassin -t < /usr/share/doc/spamassassin/examples/sample-spam.txt | less
```

```
...
X-Spam-Checker-Version: SpamAssassin 3.3.1 (2010-03-16) on
        dklab1.casafx.dyndns.org
X-Spam-Flag: Yes
X-Spam-Result: 1000.0/5.0 (GTUBE,NO_RECEIVED,NO_RELAYS)
```

... por tanto ha detectado el correo ejemplo de spam como spam y lo ha modificado para incluírle la cabecera X-Spam-Flag: Yes, lo cual permitiría posteriormente al MDA reconocerlo y almacenarlo en un mailbox aparte (en otros escenarios, el filtro de spam actúa en tiempo SMTP y hace que el correo no sea aceptado).

Por otro lado, para comprobar que spamd puede recabar las preferencias de usuario

desde ldap, podemos ejecutar en foreground una instancia de spamd así:

```
spamd -D -x --ldap-config -u nobody --max-children 5
```

y desde otra consola ejecutamos spamc así:

```
cat /usr/share/doc/spamassassin/examples/sample-spam.txt | spamc -u umea
```

de vuelta a la consola con spamd, éste mostrará mensajes relativos a ldap y las preferencias del usuario ("retrieving prefs for...").

```
dbg: ldap: URL is ldap://dklab1.casafx.dyndns.org/\
ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?\
spamassassinUserPrefs?sub?\
(&(uid=__USERNAME__)(objectClass=qmailUser))

dbg: ldap: host=dklab1.casafx.dyndns.org, port=389,\
base='ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',\
attr=spamassassinUserPrefs, scope=sub,\
filter='(&(uid=umea)(objectClass=qmailUser))'
dbg: ldap: retrieving prefs for umea: required_score 5.00
dbg: info: user has changed
```

Si todo fue bien, con CTRL-C podemos cerrar la instancia spamd de prueba, y ejecutar la definitiva:

```
invoke-rc.d spamassassin start
```

Más sobre spamassassin

Consúltese /usr/share/doc/spamassassin/README.Debian.gz para conocer la posibilidad de precompilar algunas reglas (sa-compile etc), almacenar en postgresql los parámetros estadísticos de aprendizaje bayesiano, o también el soporte por defecto a los distintos tests/plugins/RBC-lists. Respecto a ésto último, en general, algunos subsistemas están desactivados, otros están activados pero necesitan instalar software adicional (ej. pyzor) y otros están activados por defecto. Así, para habilitar o no los distintos subsistemas, habría que editar los archivos en /etc/spamassassin, en concreto los sufijados con ".pre", como init.pre, v310.pre...

Información offline: man spamassasin

Lista de posibles preferencias de usuario (bajo "USER PREFERENCES"), si bien el tipo de preferencias almacenables en ldap son (según /usr/share/doc/spamassassin/README.ldap.gz local scores, whitelist_from(s) y required_score. Cualquier otra no causará problema pero será ignorarada.

man Mail::SpamAssassin::Conf

http://spamassassin.apache.org/full/3.2.x/doc/Mail_SpamAssassin_Conf.html#user_preferenc

http://wiki.apache.org/spamassassin/UsingPyzor http://spamassassin.apache.org/full/3.3.x/doc

http://wiki.apache.org/spamassassin/WebUserInterfaces (sólo si en sql, no ldap)

dkim: http://es.wikipedia.org/wiki/DomainKeys_Identified_Mail spf : http://es.wikipedia.org/w rbc : http://en.wikipedia.org/wiki/Blacklist_%28computing%29

1.2.8. MTA

Dejemos que exim4 en debian se presente a sí mismo, con:

```
exim4 -bV
```

```
...
Support for: crypteq iconv() IPv6 PAM Perl Expand_dlfunc \
GnuTLS move_frozen_messages Content_Scanning DKIM Old_Demime
Lookups: lsearch wildlsearch nwildlsearch iplsearch cdb dbm \
dbmnz dnsdb dsearch ldap ldapdn ldapm mysql nis nis0 passwd pgsql \
sqlite
Authenticators: cram_md5 cyrus_sasl dovecot plaintext spa
Routers: accept dnslookup ipliteral iplookup manualroute \
queryprogram redirect
Transports: appendfile/maildir/mailstore/mbx autoreply lmtp pipe smtp
Fixed never_users: 0 (no "local delivery" para root, en su lugar
                      se usa usuario mail y mailbox /var/mail/mail)
...
Configuration file is /var/lib/exim4/config.autogenerated
```

```
id Debian-exim
man -k exim4
man exim4-config_files
```

Anotación: puesto que la versión en debian 6 "squeeze" es 4.72, su documentación online está en http://www.exim.org/exim-html-4.72/doc/html/spec_html/, si bien al usar la (imprescindible) caja "Search" de dicha página, tiende a hacer las búsquedas en la rama "current": http://www.exim.org/exim-html-current/doc/html/spec_html/. Es muy posible que las funcionalidades usadas en este despliegue coincidan en ambas versiones al ser exim4 un proyecto consolidado (y así estable), por tanto podemos aceptar ese comportamiento de la web. Pero, si en algún momento la documentación y el comportamiento de exim4 nos pareciesen difierir, deberíamos tener en cuenta el estar consultando la versión incorrecta de la documentación como posible causa.

Sistema de configuración de exim4

En un primer paso, diremos que, tras la instalación, el programa update-exim4.conf.template concatenó los ficheros de configuración en el árbol /etc/exim4/conf.d/ para formar un archivo "monolítico", /etc/exim4/exim4.conf.template.

Este archivo contiene la configuración junto a macros cuya inicialización se produce en ése y otros ficheros de configuración adicionales, como /etc/exim4/update-exim4.conf.conf (véase man -k exim4).

En un segundo paso ocurre que update-exim4.conf tomó este /etc/exim4/exim4.conf.template y el resto de ficheros adicionales para, finalmente, generar el /var/lib/exim4/config.autogenerated definitivo, que es lo que lee exim4 como fichero de configuración.

"invoke-rc.d exim4 start" re-realiza automáticamente todo el proceso para construir ese /var/lib/exim4/config.autogenerated definitivo.

Además, si se indicó a debconf "Split configuration into small files? No", el sistema obbia el primer paso y es en el fichero monolítico lo que se edita para incluir nuestras configuraciones (éste es nuestro caso).

Puede usarse algo como "grep ^dc_use_split_config= /etc/exim4/update-exim4.conf.conf" para comprobarlo.

A modo de conclusión, diremos que en lo que sigue, editaremos /etc/exim4/exim4.conf.template para configurar exim4. Adicionalmente /etc/default/exim4 (editable a mano o con update-exim4defaults) y /etc/exim4/update-exim4.conf.conf (editable a mano o con dpkg-reconfigure -plow exim4) controlan algunas variables particulares. Finalmente, ordenaremos que se apliquen los cambios reiniciando exim4 con invoke-rc.d exim4 stop && invoke-rc.d exim4 start

¿Cómo es y cómo se edita el fichero de configuración monolítico? /etc/exim4/exim4.conf.template, tiene la siguiente estructura grep ^begin /etc/exim4/exim4.conf.template:

```
grep ^begin /etc/exim4/exim4.conf.template
```

```
<prembule>
begin acl
 ...
begin routers
 ...
begin transports
 ...

begin retry
 ...
begin rewrite
 ...
begin authenticators
 ...
```

En el preámbulo se especifican opciones generales del servidor. Las 3 siguientes tratan el flujo usual del servicio: para cada fase de SMTP habrá una regla de control de acceso (sección acl); cuando se ha aceptado el correo habrá unas reglas para decidir el destino del correo (sección routers) por ejemplo si somos el receptor final y qué técnica de almacenamiento precisaría; cuando se decide el destino por cumplir todas las condiciones de algún router, habrá que configurar la forma de hacerlo efectivo (sección transports; por ejemplo el router que decide si el correo ha de almacenarse localmente, lo lleva a un transport que dice que ese almacenamiento se haga utilizando el formato Maildir). Por último, las tres últimas secciones (retry, rewrite, authenticators), se ocupan de situaciones especiales como la gestión de la cola, la reescritura de direcciones o la autentificación SMTP-AUTH.

Para configurar todas esas secciones, exim4 provee un DSL (domain specific language), es decir provee mecanismos como secuenciación, condicionamiento o iteración, junto a una serie de tipos de datos, variables, operadores y funciones predefinidas, todos ellos orientados al procesamiento del correo. Además, el administrador puede ayudarse de macros que serán evaluados en una fase previa de preprocesamiento.

- Macros: Puede usarse, como en el preprocesador de C, predirectivas para saltos condicionales o para incluir en línea ficheros externos; los macros en sí comienzan siempre por mayúscula, como CONFDIR en este ejemplo:
- Tipos de datos:
  - listas “domainlist”, “hostlist”, “addresslist”, o “localpartlist”. Por ejemplo, se suele definir la lista "local_domains" que comprende nombres de dominio de los que este sistema de correo se considera destino final. Por tanto es una lista

de tipo "domainlist", de identificador "local_domains" y tras el operador de asignación "=" le sigue una lista de nombres de dominio separados por ":" (o por ";" si la lista comienza por "<;") y donde la sintaxis +nombreotralista permite que se incluyan unas listas dentro de otras: domainlist local_domains = localhost:my.dom.example:+otros[1].

- resto de variables. No llevan tipo, corresponden a las opciones de configuración típicas y a built-in variables disponibles siempre para poder consultar cabeceras etc del correo que se está procesando. Diremos también que se puede consultar su valor con exim4 -bP [<varname>...], por ejemplo: "exim4 -bP +local_domains". También anotaremos que si alguna variable contiene información sensible (passwords...), puede definirse prefijada con "hide " y exim4 -bP ya no mostraría su valor.

- Funciones: son, en general, built-in luego no se definen, sino que se llaman directamente y conforme a las reglas de "string-expansion". Veamos qué significa "string-expansion":

La posibilidad de sustituir el identificador de una variable por su valor, o el de una función y sus argumentos por el valor de retorno, etc es llamado por exim4 "String Expansion": Many strings in Exim's run time configuration are expanded before use. Some of them are expanded every time they are used; others only once". http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch11.html. En exim4, aquello susceptible de ser expandido es precedido por $, ejemplos:

- variables(o listas): ${local_part}@${domain}
- función: ${lookup <search type>{<query>} {<string1>} {<string2>}}
- evaluación de condiciones: ${if <condition>{<string1>}{<string2>}}. Anotación: se considera al valor lógico falso: <empty string>, “0”, “no”, o “false”.
- operadores: ${... or {{<cond1>}{<cond2>}...} ...}

exim4 -be "<code>" es usado para practicar con la expansión del código <code>desde la línea de comandos. Es una herramienta fundamental de exim4. También existe la variedad con fichero de mensaje: exim4 -bem fichero_rfc2822 "$h_subject"[2].

- Secciones/drivers: Por último, las entidades anteriores aparecen agrupadas en secciones y drivers.

[1] http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch10.html#SECTnamedlists

[2] Anotación: exim4 -be, es decir, sin especificar <code>, abre un promp especial interpretado por exim4 -be. Este promp no tiene funcionalidades de historial y otras facilidades, pero se las podemos añadir gracias al comando rlwrap (gnu readline wrapper, del paquete rlwrap). Esta forma de usar el evaluador de expansiones de exim4 es muy útil cuando hay dificultades, precisamente por disponer de un historial y edición de línea en el prompt; sin embargo, puesto que haría farragosa la lectura de este documento, la exponemos aquí pero en lo que sigue, por hacer el texto más claro, no se hará mención.
rlwrap -D -s1000000 -H ~/.exim4_-be_history xim4 -be

- Las secciones se separan por begin <nombre de sección>, como se expuso.
- Además en las secciones acl, routers, transports, y authenticators se definen bloques de procesamiento. La sintaxis de estos drivers es <nombre driver>seguido de ":", y suelen tener una opción también de nombre driver = <nombre>que indica internamente el código de exim4 que usa la sección, y por tanto las opciones de configuración obligatorias, válidas o inválidas que puede tener ese bloque. Por ejemplo, aquí mostramos la definición de un driver en la sección routers, que tiene por nombre "mirouter_llama_mitransport" y que usa el código "accept" de exim4 para la sección routers:

```
MI_MACRO_LOOKUP = ${lookup ldap{...}}
...
begin acl
...
begin routers                          <---- En la seccio'n routers
 mirouter_llama_mitransport:           <---- un bloque/driver
  driver = accept                      <---- que usa el mo'dulo accept de exim4
  condition = ${if MI_MACRO_LOOKUP\
                   {true}\
                   {false}\
               }
  transport = mitransport              <---- y si se cumple la condicio'n llama a
 ...
begin transports                       <---- En la seccio'n transports
 mitransport:                          <---- se define el transport que llamaba el
  ...
...
```

Como se ve, los transports pueden considerarse como funciones llamadas por los routers.

- Flujo: exim4 salta a cada sección según en qué fase SMTP se encuentre y por tanto al configurar exim4 no programamos su control de flujo, con una excepción: en la sección routers el flujo es secuencial y se salta a un transport u otro según las opciones y condiciones evaluadas en cada driver, como se adelantó. Por tanto los

mecanismos típicos de control de flujo como un "if" o un "forall" que efectivamente existen en exim4, se usan restringidos en el string-expansion, no para programar el flujo global del programa, que es automático en los términos que acabamos de explicar.

Conclusiones:

El lenguaje de configuración de exim4 permite una gran flexibilidad, existiendo cerca de un millar de posibilidades entre opciones, funciones predefinidas y flags del programa. A costa de esa flexibilidad, sin embargo, se hace difícil dominar el sistema hasta el punto de poder escribir un fichero de configuración completo. Así, en la práctica la configuración consiste en comprender el fichero de configuración que se distribuye con debian, buscar en la lista de correo y otras webs recetas sobre aspectos concretos de nuestro despliegue para intentar incluirlas en aquel y, por último, afinar esa configuración usando la documentación y las amplias posibilidades de testeo de exim4:

- "exim4 -be <expression>" para comprobar el uso de bases de datos externas o cualquier otra operación de string-expansion, como se comentó.
- "exim4 -bP <optionname1>..." para consultar la configuración actual, como se comentó.
- "exim4 -d+all -bt <user>@<domain>" para simular envíos a una dirección dada, de forma que podamos comprobar qué routers se hacen cargo del correo.

Anotación sobre string expansion de las búsquedas en ldap (ldap lookups):

Buena parte de nuestro trabajo se realiza consultando la base de datos ldap, es decir, haciendo string-expansion con la función built-in "lookup", llamada convenientemente para acceder a ldap, ejem: ${lookup ldap{ldap:///ou=users,...}}. Entonces es conveniente conocer que:

- Existen dos tipos de lookups para acceder a ldap: "ldapm" y "ldap". Se diferencian en poder o no poder devolver resultados de múltiples entradas (nodos) en ldap.
- Los resultados del lookup se devulven siguiendo un "Format of data returned by LDAP" http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch09.html. Resultados de múltiples entradas (ldapm), se separan por salto de línea. Y ya dentro de cada resultado:
  - Si se preguntó por un sólo atributo, el resultado es compuesto porque el atributo sea multievaluado: se separa por comas sin más.
  - Si se preguntó por varios atributos, el formato anterior ya no vale, y se opta por formatear inequívocamente prefijando el nombre del atributo en cada caso: <atributename>="<valordevuelto>" ... Úsese extract para parsear.
  - Según en qué contexto se haga la búsqueda, lo que cuenta puede ser el resultado en sí, o simplemente si la búsqueda es o no exitosa.

Algunos ejemplos de string-expansion con lookups ldap/ldapm

```
cat <<EOF > ~/mailtest/mail.txt
From: aemu@casafx.dyndns.org
To: umea@casafx.dyndns.org
Cc:
Bcc:
Subject: hello
Reply-To:

This is the body of an rfc2822 message.
EOF
```

Algunas precauciones:

- exim4 "cachea" los resultados de las queries a cierto nivel (per message, suponemos)
- no úsense espacios tras filtro ldap y el símbolo "}", es decir, no úsese algo como "<filtro>) }}".
- Tras un escape de línea "\", el espacio en blanco hasta el primer caracter de la siguiente línea no es usado en absoluto, permitiéndonos crear márgenes de sangría para clarificar las expansiones.

Ejemplo con variable reelevante:

```
exim4 -bP +local_domains
```

Ejemplos útiles en /etc/exim4/local-main-00-gnl (véase luego):

```
exim4 -be 'ou=users,ou=accounts,dc=${sg{casafx.dyndns.org}{\\.}{,dc=}}'
exim4 -be '${if exists{/etc/ssl/certs/casafx-ca.crt}
            {/etc/ssl/certs/casafx-ca.crt}{/dev/null}}'
```

Ejemplos útiles en /etc/exim4/local-main-01-gnl (véase luego):

```
BE='-t -bem /root/mailtest/mail.txt -bfl umea -bfd casafx.dyndns.org'
exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?}}'
exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?homeDirectory?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uidNumber?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?mailQuotaSize?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?mailQuotaCount?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?mailSizeMax?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?deliveryMode?sub?
(uid=aemu)}}'
```

```
exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?mailReplyText?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?
(mailAlternateAddress=postmaster@casafx.dyndns.org)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?mailForwardingAddress?sub?
(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?deliveryProgramPath?sub?
(uid=umea)}}'

exim4 $BE '${lookup
ldap{ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?homeDirectory?sub?
(uid=umea)}}'
```

Ejemplos útiles en /etc/exim4/local-router-250-ldap (véase luego):

```
exim4 $BE '${lookup ldap{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=${sg{casafx.dyndns.org}{\\.}{,dc=}}?mailSizeMax?su
(uid=umea)}}'

exim4 $BE '${map{<\n${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?
(mailAlternateAddress=postmaster@casafx.dyndns.org)}}}
{${item}@casafx.dyndns.org}}'

exim4 $BE '${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?deliveryMode?sub?
(&(uid=umea)(deliveryMode=nolocal))}}'

exim4 $BE '${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?
(&(uid=umea)(uid=umea))}}'
#... que unidas:
exim4 $BE '${if and{{!eq{nolocal}{${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?deliveryMode?sub?
(&(uid=umea)(deliveryMode=nolocal))}}}}{eq{umea}
{${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?
(&(uid=umea)(uid=umea))}}}}}{yes}{no}}'
```

/etc/exim4/exim4.conf.template

```
vim /etc/exim4/exim4.conf.template
```

```
...
#                    MAIN CONFIGURATION SETTINGS                        #
#########################################################################
####fx:
.include /etc/exim4/local-main-00-gnl
####endfx
...
# Deny if the message contains malware. Before enabling this check, you
# must install a virus scanner and set the av_scanner option in the
# main configuration.
####fx:
.include /etc/exim4/local-acl-config-check-data-00-clamav
####endfx
...
### end acl/40_exim4-config_check_data
#######################################################
####fx:
.include /etc/exim4/local-acl-check-dkim
####endfx
...
### end router/200_exim4-config_primary
#####################################################
####fx:
.include /etc/exim4/local-router-210-spamassasing
.include /etc/exim4/local-router-250-ldap
####endfx
...
```

```
### end transport/35_exim4-config_address_directory
#########################################################
####fx:
.include /etc/exim4/local-transports-40-ldapusers
.include /etc/exim4/local-transports-50-spamassasing
####endfx
...
### end retry/00_exim4-config_header
#########################################################
####fx:
.include /etc/exim4/local-retry-20-ldap
####endfx
...
### end auth/30_exim4-config_examples
#########################################################
####fx:
.include /etc/exim4/local-auth-40-ldapusers-dovecotsasl
####endfx
```

/etc/exim4/local-main-00-gnl

Nota precaución end

```
vim /etc/exim4/local-main-00-gnl
```

```
####fx:
MAIN_LDAP_ENABLE = true
#No consulta DNS SRV RR, luego hay que definir los servidores ldap:
MAIN_LDAP_DEFAULT_SERVERS = dklab1.casafx.dyndns.org::389:dklab2.casafx.dyndns.org
MAIN_LDAP_VERSION = 3
MAIN_LDAP_BASEDN = ou=users,ou=accounts,dc=${sg{$domain}{\\.}{,dc=}}
MAIN_LDAP_TIMEOUT = 15
#MAIN_LDAP_BINDDN =
#MAIN_LDAP_BINDPW =

.include /etc/exim4/local-main-01-ldap
#
MAIN_TLS_ENABLE = true
MAIN_TLS_CERTIFICATE = /etc/exim4/exim4.casafx.dyndns.org.crt
MAIN_TLS_PRIVATEKEY  = /etc/exim4/exim4.casafx.dyndns.org.key
MAIN_TLS_VERIFY_CERTIFICATES = ${if exists{/etc/ssl/certs/casafx-ca.crt}\
                                  {/etc/ssl/certs/casafx-ca.crt}\
                                  {/dev/null}}
#
av_scanner = clamd:/var/run/clamav/clamd.ctl
#...el soporte para el filtro antivirus va a necesitar, adicionalmente,
#   modificar "acl_check_data:"
.ifndef CHECK_RCPT_IP_DNSBLS
CHECK_RCPT_IP_DNSBLS = cbl.abuseat.org:dnsbl.njabl.org:sbl.spamhaus.org
.endif  #...es "warn", se puede cambiar a deny en .ifdef CHECK_RCPT_IP_DNSBLS
.ifndef CHECK_RCPT_SPF
CHECK_RCPT_SPF = true
.endif
```

```
#
#auth_advertise_hosts = * por defecto y se aplica si has definido al menos un dri
#
# Para usar DKIM y firmar correo saliente:
DKIM_DOMAIN = ${lc:${domain:$h_from:}}
DKIM_FILE = /etc/exim4/exim4.casafx.dyndns.org.key
DKIM_PRIVATE_KEY = ${if exists{DKIM_FILE}{DKIM_FILE}{0}}
DKIM_SELECTOR = exim4dkim
DKIM_CANON = relaxed
#... todas son, si definidas, usadas en el transport "remote_smtp"
#    definido en /etc/exim4/exim4.conf.template.
# Para utilizar DKIM verificando firmas en correo entrante:
acl_smtp_dkim = acl_check_dkim
# ...definiremos la acl de nombre "acl_check_dkim" posteriormente.
####endfx
```

/etc/exim4/local-main-01-ldap

```
vim /etc/exim4/local-main-01-ldap
```

```
####fx:
# Este fichero se incluye desde /etc/exim4/local-main-00-gnl
# y no desde el /etc/exim4/exim4.conf.template.
.ifdef MAIN_LDAP_ENABLE

.ifndef MAIN_LDAP_DEFAULT_SERVERS
MAIN_LDAP_DEFAULT_SERVERS = dklab1.casafx.dyndns.org::389:dklab2.casafx.dyndns.org:
.endif
ldap_default_servers = MAIN_LDAP_DEFAULT_SERVERS

.ifndef MAIN_LDAP_VERSION
MAIN_LDAP_VERSION = 3
.endif
ldap_version = MAIN_LDAP_VERSION

.ifndef MAIN_LDAP_BASEDN
#MAIN_LDAP_BASEDN = ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
MAIN_LDAP_BASEDN = ou=users,ou=accounts,dc=${sg{${lc:$domain}}{\\.}{,dc=}}
.endif
LDAP_BASEDN = MAIN_LDAP_BASEDN
```

```
#.ifndef MAIN_LDAP_BINDDN
#MAIN_LDAP_BINDDN = cn=admin,dc=example,dc=com
#MAIN_LDAP_BINDPW = CHANGE
#.endif
#LDAP_BINDDN = MAIN_LDAP_BINDDN
#LDAP_BINDPW = MAIN_LDAP_BINDPW
```

```
.ifndef MAIN_LDAP_TIMEOUT
MAIN_LDAP_TIMEOUT = 15
.endif
LDAP_TIMEOUT = MAIN_LDAP_TIMEOUT

# Determines how aliases are handled during a search. This option is available
# only with OpenLDAP 2.0.
# NOTE: DEREF can be one of the following values: never, searching, finding,
#       always. If not specified, aliases are never dereferenced.
#
.ifndef MAIN_LDAP_DEREF
MAIN_LDAP_DEREF = never
.endif
LDAP_DEREF = MAIN_LDAP_DEREF
```

```
LDAP_UID = uidNumber
#lo uso porque procmail parece necesitarlo; si por ejemplo
#si usa vmail, entiende que el home debe ser el de
#vmail etc. Necesito investigarlo +, pero LDAP_UID debe ser definido.
#No usado: LDAP_GID = gidNumber, uso vmail
#No usado: LDAP_MAILSTORE = mailMessageStore, en su lugar uso:

# The LDAP_HOMEDIR attribute MUST exist, and we MUST be able to chdir to it. It
# is used as $home during transport.
#
LDAP_HOMEDIR = homeDirectory
# If homeDirectory is not an absolute path, define the root of the relative
# paths in LDAP_MAILROOT.
# LDAP_MAILROOT = /
# The LDAP_MAILDIR attribute is OPTIONAL, and specifies the location of target
# maildir. If specified as relative path, $home is automatically prepended to
# it. If not specified, $home/Maildir will be used.
# LDAP_MAILDIR = Maildir

LDAP_QUOTA_SIZE = mailQuotaSize
LDAP_QUOTA_COUNT = mailQuotaCount

LDAP_MAXSIZE = mailSizeMax
```

```
# Multi field entries of these keywords:
# - (no entry): Default delivery, put message into maildir (localdelivery),
#   plus forward and program delivery if specified.
# - noforward: Do not forward (ignores forwarding entry).
# - nolocal: Do not put message into maildir.
# - noprogram: Do no do program deliveries.
# - reply: Send a vacation message with text from LDAP_REPLYTEXT
LDAP_MODE = deliveryMode

LDAP_REPLYTEXT = mailReplyText
```

```
# No usado (...en jabberd2 no lo uso, aqui tampoco debiera):
#
#The status of a user account. Values:
# - active (no restrictions).
# - noaccess (only mail delivery but no pop/imap access).
# - disabled (bounce incoming messages).
# - deleted (bounce incoming messages and mark for deletion, see LDAP_PURGE)
# NOTE: no accountStatus is equal to active.
#
#LDAP_ISACTIVE = accountStatus
#
# If accountStatus set to 'deleted', the earliest date when the
# mailMessageStore including all remaining content will be deleted from the
# filesystem. Values: date and time in seconds since Jan. 1, 1970 (the epoch)
# NOTE: This deletion has to be done by an external helper program, for
#       example periodically run from cron. A sample script is included under
#       the name qmailAccountPurge.sh. Handle automatic deletions with care!
# todo: Not yet implement.
#
#LDAP_PURGE = qmailAccountPurge
#
# Y por tanto, tampoco uso este filtro (aqui se equivocaria el de jabberd2, btw)
```

```
LDAP_DEFAULT_FILTER = (LDAP_MAIL=${local_part})
#LDAP_DEFAULT_FILTER = (&LDAP_FILTER(LDAP_MAIL=${local_part}@${domain}))

LDAP_DEFAULT_MAXSIZE = ${lookup ldap{\
  LDAP_DEFAULT_CONTROLS \
  ldap:///LDAP_BASEDN?LDAP_MAXSIZE?sub?LDAP_DEFAULT_FILTER\
}}

# This macro is a short-hand for LDAP_HOMEDIR query.
LDAP_DEFAULT_HOMEDIR = ${lookup ldap{\
  LDAP_DEFAULT_CONTROLS \
  ldap:///LDAP_BASEDN?LDAP_HOMEDIR?sub?LDAP_DEFAULT_FILTER\
}}

# This macro is a short-hand for LDAP_MAILSTORE query.
#LDAP_DEFAULT_MAILSTORE = ${lookup ldap{\
#  LDAP_DEFAULT_CONTROLS \
#  ldap:///LDAP_BASEDN?LDAP_MAILSTORE?sub?LDAP_DEFAULT_FILTER\
#}}

.endif
####endfx
```

/etc/exim4/local-router-250-ldap

Diseño y flujo El flujo global de exim4 era automático, no lo "programábamos" nosotros excepto la sección routers que discriminan qué hacer con un correo aceptado. Cuando exim4 salta a la sección begin routers, recordemos:

- Se hace una evaluación secuencial de los routers.
- Un router se usa si su "condition" se cumple y, cuando el driver es redirect, "data"

no está vacía. Entonces, se salta al transporte indicado.

- Si el router llevaba "unseen", el flujo se desdobla y además de saltar al transporte también continúa por el resto de routers; unseen puede establecerse de forma condicional, siendo ésta la forma más flexible de usarlo.
- Si el router es de "driver = redirect", el correo (tras transformaciones efectuadas por el router) vuelve a pasar por todos los routers, salvo algún caso (en ese caso veremos que en "data" aparece :fail: etc...).

Dadas esas premisas, nuestros routers, en orden, resuelven que:

- Si el correo es demasiado grande, debiéramos avisar de ello y no almacenarlo: usamos "driver = redirect"; "data" contiene :fail: ; "condition" es el peso del correo y el aviso está también en "data"; no usamos "unseen" para que no se desdoble y continúe.
- Si el correo es para alguien que está de vacaciones, debemos avisar además de almacenarlo: usamos driver accept; "condition" es que el deliveryMode en ldap sea "reply"; "unseen" para que siga y se almacene.
- Puede que el correo corresponda a una cuenta ldap a través de un alias: uso "driver = redirect"; la condición es simplemente que "data" contenga la resolución/es del alias al nombre de la cuenta; no "unseen".
- Puede que el correo a esa cuenta deba ser también reenviado a otra: usamos "driver = redirect"; la "condition" es que el forwarding esté explícitamente permitido según el metadato deliveryMode en ldap y que "data" contenga la dirección/es de forwarding; "unseen" hará que se entregue también aquí en virtud del siguiente router, si bien para permitir que sólo se haga forwarding y no entrega, "unseen" lo ponemos condicional a la existencia de nolocal como valor del atributo mailDelivery en ldap.
- Ahora viene el caso principal, el que nos permite almacenar el correo: El correo es de una cuenta centralizada, y debe ser aceptado para hacer el delivery por el transport por defecto correspondiente, sin menoscabo de otros posibles mecanismos seleccionables más tarde: usamos driver accept; la "condition" es que el usuario sea de una cuenta en ldap y el deliveryMode lo permita (no anuncie nolocal); "unseen" lo ponemos condicional a que el siguiente router vaya a ser usado (que deliveryMode no incluya noprogram).
- El correo puede ser entregado por otros mecanismos declarados en ldap en el atributo deliveryProgramPath, si bien éste es el último tratamiento para un correo enviado a las cuentas centralizadas: uso "driver = redirect"; "condition" es que el deliveryMode no incluya noprogram; así como que "data" contenga el path del programa que hará la entrega adicional; no "unseen".
- Finalmente, si un correo para una cuenta centralizada no ha sido entregado ya, se hará fallar y se enviará un correo informativo. Tal como se hizo para el filtro por tamaño: usamos "driver = redirect"; "data" contiene :fail: ; "condition" como se ha descrito; el aviso está también en "data"; no "unseen".

Routers Ya estamos en condiciones de escribir los routers que corresponden a cada uno de los puntos aclarados.

```
vim /etc/exim4/local-router-250-ldap
```

```
####fx:
.ifdef MAIN_LDAP_ENABLE
# - Si el correo es demasiado grande, avisaremos de ello y no lo
# almacenaremos: usamos "driver = redirect"; "data" contiene :fail: ;
# "condition" es el peso del correo y el aviso se encuentra igualmente en
# "data"; no usamos "unseen" para que no se desdoble y pueda continuar.
#
# This router is a bit tricky: we only defer message if both mailSizeMax is
# positive (i.e. valid) and $message_size > mailSizeMax. Unset mailSizeMax or
# set mailSizeMax as 0 will disable this checking.
ldap_maxsize:
  debug_print = "R: ldap_maxsize for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  condition = ${if \
                  and{\
                      {>{LDAP_DEFAULT_MAXSIZE}{0}}\
                      {>{$message_size}{LDAP_DEFAULT_MAXSIZE}}\
                     }\
                  {yes}{no}\
                }
  data = :fail:\n\Your message is too big.\n\
    Your message was rejected because the user ${local_part}@${domain}\n\
    does not accept message larger than LDAP_DEFAULT_MAXSIZE.
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
```

```
# - Si el correo es para alguien que se encuentra ausente, debemos avisar
# y almacenarlo: usamos driver accept; "condition" es que el
# deliveryMode en ldap sea "reply"; "unseen" para que siga y se almacene.
#
# Vacation need special LDAP transport for reply text fetching.
ldap_vacation:
  debug_print = "R: ldap_replytext for $local_part@$domain"
  driver = accept
  no_verify
  no_expn
  unseen
  condition = ${if \
                  and{\
                      {!match{$h_precedence:}{(?i)junk|bulk|list}}\
                      {match{${lookup ldap{\
                                LDAP_DEFAULT_CONTROLS \
                                ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
                               (&LDAP_DEFAULT_FILTER(LDAP_MODE=reply))}}\
                             }{reply}}\
                  }\
                  {yes}{no}\
                }
  transport = ldap_reply

# - Puede que el correo corresponda a una cuenta ldap a mediante de un alias:
# uso "driver = redirect"; el condicionante es simplemente que "data"
# contenga resuelto/s el alias al nombre de la cuenta; no "unseen".
#
# A single user can contain multiple alias addresses.
```

```
ldap_mailalternate:
  debug_print = "R: ldap_mailalternate for ${local_part}@${domain}"
  driver = redirect
  no_verify
  no_expn
  #unseen
  check_ancestor
  data = ${map{<\n${lookup ldapm{\
                                LDAP_DEFAULT_CONTROLS \
                                ldap:///LDAP_BASEDN?LDAP_MAIL?sub?\
                                (LDAP_MAILALTERNATE=\
                                ${local_part}@${domain})\
                               }}\
                }{${item}@${domain}}}
```

```
  #data = ${lookup ldapm{\
  #  LDAP_DEFAULT_CONTROLS \
  #  ldap:///LDAP_BASEDN?LDAP_MAIL?sub?(LDAP_MAILALTERNATE=${local_part}@${domain})
  #}}
  #...puesto que la consulta ldap solamente va a devolver uid, necesitamos el
  #@domain final. El problema igualmente es que ldapm puede devolver varios,
  #por ello hay que utilizar map y ldapm:
  #${map{ ${lookup ldapm{}}  }{$item@{domain}}}
  #Map: http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch11.html
  #Format of data returned by LDAP:
  # http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch09.html
  # Resultados de diferentes entradas (ldapm), se separan por salto de linea.
  # Y ya dentro de cada resultado:
  # -Si se pregunta por solamente un atributo, el resultado es compuesto
  #  porque el atributo sea multievaluado: se separa por comas simplemente.
  # -Si se pregunta por varios atributos, se formatea de esta manera:
  #  <atributename>="<valordevuelto>" ...  Se usa extract para parsear.

# - Puede que el correo a esa cuenta deba ser igualmente reenviado a otra:
# usamos "driver = redirect"; la "condition" es que el forwarding sea
# permitido al dictarlo el metadato deliveryMode en ldap y que
# "data" contenga la direccion/es de forwarding; "unseen" hace que se
# entregue igualmente y en virtud del siguiente router, si bien para
# permitir que solamente se haga forwarding y no entrega, "unseen" lo ponemos
# condicional a la existencia de nolocal como valor del atributo
# mailDelivery en ldap.
#
# Don't forget unseen! Or else we will only forward mail without local
# delivery.
```

```
ldap_forwards:
  debug_print = "R: ldap_forwards for ${local_part}@${domain}"
  driver = redirect
  check_ancestor
  #unseen

  #Evitamos que el correo para una cuenta centralizada pase por los routers
  #para correo local, haciendo el "unseen" condicional a que efectivamente el
  #correo se vaya a someter a la entrega por defecto para usuarios centralizados.
  unseen = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?(&LDAP_DEFAULT_FILTER(LDAP_MODE=nolocal))\
  }}{nolocal} {no}{yes}}

  data = ${lookup ldapm{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_FORWARDS?sub?LDAP_DEFAULT_FILTER\
  }}
  condition = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=noforward))\
  }}{noforward} {no}{yes}}
```

```
# - Ahora viene el caso principal, el que nos permite almacenar el correo:
# El correo es de una cuenta centralizada, y debe ser aceptado para
# hacer el delivery por el transport por defecto correspondiente, sin
# menoscabo de otros posibles mecanismos seleccionables luego: usamos
# driver accept; la "condition" es que el usuario sea de una cuenta en
# ldap y el deliveryMode lo permita (no anuncie nolocal); "unseen" lo
# ponemos condicional a que el siguiente router vaya a ser usado
# (que deliveryMode no incluya noprogram).
#
#-If a user is allow for local delivery, transfer it to ldap_file transport.
#-ldap_mailstore:
#-  debug_print = "R: ldap_mailstore for ${local_part}@${domain}"
#-  driver = accept
#-  unseen
#-  condition = ${if match{${lookup ldap{\
#-    LDAP_DEFAULT_CONTROLS \
#-    ldap:///LDAP_BASEDN?LDAP_MODE?sub?(&LDAP_DEFAULT_FILTER(LDAP_MODE=nolocal))\
#-  }}}{nolocal} {no}{yes}}
#-  local_part_suffix = -*
#-  local_part_suffix_optional
#-  retry_use_local_part
#-  transport = ldap_file
```

```
ldap_dovecot_mda:
  debug_print = "R: ldap_dovecot_mda for ${local_part}@${domain}"
  driver = accept
  #Evitamos que el correo para una cuenta centralizada pase por los routers
  #para correo local, haciendo el "unseen" condicional a que noprogram se
  #encuentre declarado.
  #noprogram debiera estar declarado cuando no hay mailProgramPath alguno.
  unseen = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
   ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
   (&LDAP_DEFAULT_FILTER(LDAP_MODE=noprogram))\
  }}}{noprogram} {no}{yes}}
  condition = ${if and{\
                       {!eq{nolocal}\
                          {${lookup ldap{ LDAP_DEFAULT_CONTROLS \
           ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
          (&LDAP_DEFAULT_FILTER(LDAP_MODE=nolocal))}}}\
                       }\
                       {eq{$local_part}\
                          {${lookup ldap{ LDAP_DEFAULT_CONTROLS \
           ldap:///LDAP_BASEDN?LDAP_MAIL?sub?\
           (&LDAP_DEFAULT_FILTER(LDAP_MAIL=${local_part}))}}}}\
                      }\
                    {yes}{no}}
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
  transport = dovecot_mda
```

```
# - El correo puede ser entregado por otros mecanismos declarados en ldap
# en el atributo deliveryProgramPath, si bien nos encontramos en el u'ltimo
# tratamiento para un correo enviado a las cuentas centralizadas: uso
# "driver = redirect"; "condition" es que el deliveryMode no incluya
# noprogram; por tanto como que "data" contenga el path del programa que
# hace la entrega adicional; no "unseen".
ldap_program:
  debug_print = "R: ldap_program for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  #unseen
  data = ${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_PROGRAM?sub?LDAP_DEFAULT_FILTER\
  }}
  condition = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=noprogram))\
  }}}{noprogram} {no}{yes}}
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
  pipe_transport = ldap_pipe
  reply_transport = address_reply
```

```
# - Finalmente, si un correo para una cuenta centralizada no ha sido
# entregado ya, se hace fallar y se manda un correo informativo. Tal
# como se hizo para el filtro por "size": usamos "driver = redirect";
# "data" contiene :fail: ; "condition" como se ha descrito; el aviso se
# encuentra igualmente en "data"; no "unseen".
ldap_fail:
  debug_print = "R: ldap_fail for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  #En la v4.72 de exim4 no existe bool_lax, luego para que el and pueda
  #interpretar las consultas a ldap, tengo que usar bool+strlen y bool+eq con 0
  #(porque necesito construir algo como not{bool} que no parece existir, ni
  #siquiera !bool)
```

```
condition = ${if \
               and{\
                   {bool{\
                    ${strlen:\
                      ${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                      ldap:///LDAP_BASEDN}}\
                     }\
                    }\
                   }\
                   {eq{\
                       ${strlen:\
                         ${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                                  ldap:///LDAP_BASEDN?LDAP_MAIL?sub?
                        }\
                      }\
                      {0\
                      }\
                    }\
                  }\
                {yes}{no}}
```

```
  data = :fail:\n\Delivery Status Notification (Failure)\n\
    Delivery to the following recipient failed permanently:\n\
    ${local_part}@${domain}\n\
    The error returned was:\n\
    The email account that you tried to reach does not exist.
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
.endif
####endfx
```

/etc/exim4/local-transports-40-ldapusers

```
vim /etc/exim4/local-transports-40-ldapusers
```

```
####fx:
.ifdef MAIN_LDAP_ENABLE
#
# No puedo utilizar el driver appendfile porque exim4 no se encuentra preparado
# para hacer el delivery en AFS; a su vez solamente ese driver
# tiene implementados controles de quota, luego no podemos ejemplificar
# el uso de quotas en exim4. En cualquier caso remarcamos que, al igual
# que dovecot-lda, las soporta.
#
# Procmail igualmente permite configurar quotas:
# http://www.perlcode.org/tutorials/procmail/procmail_cookbook.txt
#
```

```
#-# When a message is redirected to a "mail folder", the folder name given must
#-# always be an absolute path.
#-# It works like this:
#-#   IF LDAP_HOMEDIR exists
#-#     IF LDAP_HOMEDIR is absolute
#-#       LDAP_HOMEDIR is used as $home
#-#     ELSE
#-#       LDAP_HOMEDIR is prepend with LDAP_MAILROOT and used as $home
#-#     FI
#-#   ELSE
#-#     LDAP_MAILROOT is used as $home
#-#   FI
#-#   IF LDAP_MAILSTORE exists
#-#     IF LDAP_MAILSTORE is absolute
#-#       LDAP_MAILSTORE is used as target directory
#-#     ELSE
#-#       LDAP_MAILSTORE is prepend with $home and used as target directory
#-#     FI
#-#   ELSE
#-#     $home/Maildir is used as target directory
#-#   FI
#-# NOTE: LDAP_DEFAULT_HOMEDIR/LDAP_DEFAULT_MAILSTORE are used as short-hand of
#-#       LDAP_HOMEDIR/LDAP_MAILSTORE for simpler syntax.
```

```
#-ldap_file:
#-  debug_print = "T: ldap_delivery for $local_part@$domain"
#-  driver = appendfile # este driver permite funcionalidad de quota
#-  home_directory = ${if !eq{LDAP_DEFAULT_HOMEDIR}{} \
#-    {${if eq{${substr_0_1:LDAP_DEFAULT_HOMEDIR}}{/} \
#-      {LDAP_DEFAULT_HOMEDIR} \
#-      {LDAP_MAILROOT/LDAP_DEFAULT_HOMEDIR} \
#-    }} \
#-    {LDAP_MAILROOT} \
#-  }
#-  directory = ${if !eq{LDAP_DEFAULT_MAILSTORE}{} \
#-    {${if eq{${substr_0_1:LDAP_DEFAULT_MAILSTORE}}{/} \
#-      {LDAP_DEFAULT_MAILSTORE} \
#-      {$home/LDAP_DEFAULT_MAILSTORE} \
#-    }} \
#-    {$home/Maildir} \
#-  }
#-  user = ${lookup ldap{\
#-    LDAP_DEFAULT_CONTROLS \
#-    ldap:///LDAP_BASEDN?LDAP_UID?sub?LDAP_DEFAULT_FILTER\
#-  }}
#-  group = ${lookup ldap{\
#-    LDAP_DEFAULT_CONTROLS \
#-    ldap:///LDAP_BASEDN?LDAP_GID?sub?LDAP_DEFAULT_FILTER\
#-  }}
```

```
#-  .ifdef LDAP_QUOTA_SIZE   # todo: debo poner 0 para deshabilitar la quota
#-  quota = ${lookup ldap{\
#-    LDAP_DEFAULT_CONTROLS \
#-    ldap:///LDAP_BASEDN?LDAP_QUOTA_SIZE?sub?LDAP_DEFAULT_FILTER\
#-  }}
#-  .endif
#-  .ifdef LDAP_QUOTA_COUNT  # todo: debo poner 0 para deshabilitar la quota
#-  quota_filecount = ${lookup ldap{\
#-    LDAP_DEFAULT_CONTROLS \
#-    ldap:///LDAP_BASEDN?LDAP_QUOTA_COUNT?sub?LDAP_DEFAULT_FILTER\
#-  }}
#-  .endif
#-  quota_is_inclusive = false   # no incluyas el peso del mail actual
#-  maildir_use_size_file          # memoriza el peso en fichero maildirsize
#-  maildir_quota_directory_regex = ^(?:cur|new|\.(?!Trash).*)$
#-  create_directory
#-  delivery_date_add
#-  envelope_to_add
#-  return_path_add
#-  maildir_format
```

```
#-  .ifdef MAILDIR_HOME_DIRECTORY_MODE
#-  directory_mode = MAILDIR_HOME_DIRECTORY_MODE
#-  .else
#-  directory_mode = 0700
#-  .endif
#-  .ifdef MAILDIR_HOME_MODE
#-  mode = MAILDIR_HOME_MODE
#-  .else
#-  mode = 0600
#-  .endif
#-  mode_fail_narrower = false
#-  quota_warn_threshold = 80%
#-  quota_warn_message = "\
#-    To: $local_part@$domain\n\
#-    Subject: Mailbox quota warning\n\n\
#-    This message was automatically generated by the mail delivery software.\n\n\
#-    You are now using over 80% of your allocated mail storage quota.\n\n\
#-    If your mailbox fills completely, further incoming message will be automatic
#-    returned to their senders.\n\n\
#-    Please take note of this and remove unwanted mail from your mailbox.\n"
#-
```

```
dovecot_mda:
  debug_print = "T: dovecot_mda for $local_part@$domain"
  driver = pipe
  command = /usr/lib/dovecot/deliver -d ${local_part}@${domain}  \
            -f ${sender_address}
  message_prefix =
  message_suffix =
  delivery_date_add
  envelope_to_add
  return_path_add
  log_output
  user = vmail
  temp_errors = 64 : 69 : 70: 71 : 72 : 73 : 74 : 75 : 78



# Don't use "command" here as we are not mapping to a static program but
# dynamic from deliveryProgramPath. It should come from router, for a complete
# command syntax, with "|" prefix.
```

```
ldap_pipe:
  debug_print = "T: ldap_pipe for $local_part@$domain"
  driver = pipe
  return_output
  log_output
  home_directory = ${if !eq{LDAP_DEFAULT_HOMEDIR}{} \
                        {${if eq{${substr_0_1:LDAP_DEFAULT_HOMEDIR}}{/} \
                              {LDAP_DEFAULT_HOMEDIR} \
                              {LDAP_MAILROOT/LDAP_DEFAULT_HOMEDIR} }} \
                        {LDAP_MAILROOT} }
  user = ${lookup ldap{LDAP_DEFAULT_CONTROLS \
         ldap:///LDAP_BASEDN?LDAP_UID?sub?LDAP_DEFAULT_FILTER}}
  #user = vmail
  #... si no resuelve el uid y, en cambio, usa vmail, la regla de procmail
  #    intenta acceder al home de vmail, no de ${local_part}. Al menos
  #    procmail no parece utilizar la variable home_directory anterior, sino
  #    que resuelve el home mediante el uid.
  #group = ${lookup ldap{LDAP_DEFAULT_CONTROLS\
  #        ldap:///LDAP_BASEDN?LDAP_GID?sub?LDAP_DEFAULT_FILTER}}
  group = vmail
  return_path_add
  delivery_date_add
  envelope_to_add
  path = "/bin:/usr/bin:/usr/local/bin"
```

```
ldap_reply:
  debug_print = "T: ldap_reply for ${local_part}@${domain}"
  driver = autoreply
  from = "${local_part}@${domain}"
  to = ${sender_address}
  subject = "Autoreply from ${local_part}@${domain}"
  text = ${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_REPLYTEXT?sub?LDAP_DEFAULT_FILTER\
  }}
.endif
####endfx
```

/etc/exim4/local-retry-20-ldap

```
vim /etc/exim4/local-retry-20-ldap
```

```
####fx:
#Si no usamos el sistema de quotas de exim4, no hay nada que configurar:
#.ifdef MAIN_LDAP_ENABLE
#*                      quota
#.endif
####endfx
```

/etc/exim4/local-auth-40-ldapusers-dovecotsasl

```
vim /etc/exim4/local-auth-40-ldapusers-dovecotsasl
```

```
####fx:
 #http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch33.html
 #http://wiki.dovecot.org/HowTo/EximAndDovecotSASL
#Uncomment the following lines in dovecot config
# socket listen {
#  client {
#     path = /var/run/dovecot/auth-client
#     user = dovecot
#     group = exim
#     mode = 0660
#  }
# }
#Add gssapi to  mechanisms = line in dovecot config like this:
#auth default {
#   mechanisms = gssapi plain
#  ...}
#[[http://www.dovecot.org/list/dovecot/2011-February/057178.html][ref]]:
dovecot_gssapi:
    driver = dovecot
    public_name = GSSAPI
    server_socket = /var/run/dovecot/auth-client
    server_advertise_condition = yes
    # Cuidado, la siguiente posibilidad "might break several headers in
    # mails sent by authenticated smtp"
    server_set_id=GSSAPI-${quote:$auth1}
                  #otra posibilidad es server_set_id = $auth1 ...
```

```
#   # Las siguientes no funcionan con driver=dovecot, porque
#   # se especifican en dovecot.conf
#   # server_mech = GSSAPI
#   # server_service = smtp
#   # server_hostname = ${primary_hostname} ...dklab1.casafx.dyndns.org
#   # server_realm = ${domain}              ...CASAFX.DYNDNS.ORG
#   http://wiki.dovecot.org/HowTo/EximAndDovecotSASL

####
#gssapi_sasl_server:
# driver = cyrus_sasl
# public_name = GSSAPI
# server_set_id = $auth1
# #otras:
# server_mech = GSSAPI
# server_service = smtp
# #en server_hostname, puedo poner dklab1.casafx.dyndns.org o:
# server_hostname = $primary_hostname
# #en server_realm, CASAFX.DYNDNS.ORG o:
# server_realm = ${domain}
####

# Dejo los siguientes (login y plain) para el sistema local
####endfx
```

/etc/exim4/local-acl-config-check-data-00-clamav

```
vim /etc/exim4/local-acl-config-check-data-00-clamav
```

```
####fx:
 deny
    malware = *
    message = This message was detected as possible malware ($malware_name).
####endfx
```

/etc/exim4/local-acl-check-dkim

```
vim /etc/exim4/local-acl-check-dkim
```

```
####fx:
acl_check_dkim:
 #Check DKIM signature for gmail (dkim_signers y dkim_status son condiciones)
    deny message            = GMail/Paypal/Ebay sender with invalid \
                              or missing DKIM signature
    sender_domains  = gmail.com:paypal.com:ebay.com
        dkim_signers    = gmail.com:paypal.com:ebay.com
        dkim_status     = none:invalid:fail
    #...en este momento, parece razonable verificar DKIM
    #   solamente en ciertos dominios: Gmail por volumen
    #   y Ebay/Paypal por su informacion sensible
    #   parecen buenos candidatos y es conocido que usan DKIM,
    accept
####endfx
```

/etc/exim4/local-router-210-spamassasing

```
vim /etc/exim4/local-router-210-spamassasing
```

```
####fx:
spamcheck_router:
  debug_print = "R Spamassassin"
  driver = accept
  # Este router, desde el punto de vista de exim4, manda el correo a otra
  # instancia de exim4 (en un modo especial "batched" para protocolo
  # "spam-scanned"). A consecuencia de ello, el rastro de
  # los tests de tipo "exim4 -bt <address>" se para sistema'ticamente
  # en este punto.
  # Se soluciona evitando que este router sea tenido en cuenta en estos tests,
  # usando no_address_test:
  no_address_test
  no_verify
  #no_expn
  #domains = +local_domains : +relay_to_domains # si restringimos a correo nuestro
  #
  # When to scan a message (independientemente del sistema antispam usado luego)
  #   -   message size within limits
  #   -   it isn't already flagged as spam
  #   -   it isn't already scanned
  condition = ${if and { {<{$message_size}{100K}}\
 {!def:header_X-Spam-Flag:} {!eq {$received_protocol}{spam-scanned}}} {1}{0}}
  transport = spamcheck
####endfx
```

/etc/exim4/local-transports-50-spamassasing

```
vim /etc/exim4/local-transports-50-spamassasing
```

```
####fx:
spamcheck:
    driver = pipe
    debug_print = "T: spamassassin_pipe for ${local_part}@${domain}"
    command = /usr/sbin/exim4 -oMr spam-scanned -bS # -bS para batched
    use_bsmtp = true
    transport_filter = /usr/bin/spamc -u ${local_part}
    home_directory = "/dev/shm"
    current_directory = "/dev/shm"
    #...nota: /dev/shm va a ser cambiado por /run/shm a corto plazo:
    #         http://wiki.debian.org/ReleaseGoals/RunDirectory
    user = Debian-exim
    group = mail
    log_output = true
    return_fail_output = true
    return_path_add = false
    message_prefix =
    message_suffix =
    #ignore_status = true #si servidores sobrecargados que rebotan\
    #el correo con 421 SMTP timeouts
####endfx
```

Dspam es otro scanner antispam, más orientado aún a aprendizaje adaptativo que spamassasin, si bien parece no tener soporte para recabar las preferencias de usuario desde ldap (http://dspam.expass.de/wiki/Preferences_Attributes):
http://ubuntuforums.org/archive/index.php/t-77766.html Comparativa sa
http://ejohansson.se/archives/2006/07/25/my-dspam-configuration/ Config
http://wiki.exim.org/DspamWithExim Config sa + ds

Comentario sobre la funcionalidad de sufijos

En la configuración expuesta anteriormente, aparecen comentadas las opciones "local_part_suffix = -*" y "local_part_suffix_optional". El cometido de ambas es permitir

que un correo con la parte local "umea-<sufijo cualquiera>" pueda ser interpretado como para umea de la misma forma que ocurriría si la parte local fuese simplemente "umea". Este sistema de sufijos se ideó para permitir automáticamente la entrega en diferentes mailboxes, sin necesidad de escribir filtros dedicados en algún sistema como sieve o procmail. Aunque parece una funcionalidad interesante, no se puede desplegar correctamente (sí a nivel de exim4 -y probablemente procmail- pero no a nivel de nuestro principal sistema de entrega, dovecot deliver). Los detalles se exponen a continuación:

- Dovecot Deliver: En dovecot utilizamos la parte local para construir la ruta a la mailbox, éso implicaría que necesitamos algún operador para extraer lo que haya antes del guión, sin embargo ésto no está implementado: maildir:/afs/ %Ld/service/mail/ %1n/ %2n/ ¿? %n /Maildir.
- Exim4 necesitaría algunos cambios: exim4 -bt umea-misufijo@casafx.dyndns.org sí haría match en los routers ldap_dovecot_mda y en ladp_program. Sin embargo, el forwarding a aemu no funcionaría porque con "umea-misufijo" no se puede acceder al atributo con el forwarding, habría que transformarlo sin -misufijo al hacer la búsqueda en ldap. A este respecto "${extract{1}{-}{$local_part}}" podría utilizarse (funciona como el awk -F- '{print $1}'de la línea de comandos). Respecto a los aliases: exim4 -bt postmaster-yea@casafx.dyndns.org no hará match porque con el sufijo no es un alias de nadie, podríamos alterar de nuevo la consulta y después sufijar la respuesta con la pregunta (ejemplo postmaster ->umea ->correo a umea-postmaster finalmente).
- Por último, el caso de poder utilizar este esquema en un futuro implicaría también el tener la política de no crear nombres de usuario con guiones, sino usando por ejemplo con '.én el caso de que fuese conveniente un símbolo separador.

Tests

Haremos algunas comprobaciones parciales de distitos subsistemas; en la sección sobre los clientes se hará una prueba completa.

Sintaxis; inicio de exim4 con la nueva configuración Podemos realizar una comprobación de corrección sintáctica con:

```
update-exim4.conf -v
```

... si todo está bien, no debiera haber salida alguna; en otro caso indicará dónde y en qué consiste el error. Tras solucionarlo, es también conveniente asegurarse de que /var/log/exim4/paniclog esté vacío:

```
:> /var/log/exim4/paniclog
```

Por fir iniciamos exim4:

```
invoke-rc.d exim4 start
```

MX DNS RR Comprobaremos que están ahí:

```
dig @dklab1.casafx.dyndns.org. casafx.dyndns.org MX +short
```

```
    0 dklab1.casafx.dyndns.org.
    0 dklab2.casafx.dyndns.org.
```

SASL-GSSAPI para SMTP-AUTH con dovecot-auth y smtptest Puesto que hemos configurado exim4 para delegar la autenticación al componente dovecot-auth de Dovecot, esta prueba sólo se podrá realizar cuando hallamos instalado y configurado Dovecot.

Sender Policy Framework La base de la validación de SPF es que la IP de origen en la conexión SMTP esté admitida como origen de un correo. Esa validación se declara en un registro DNS asociado al domain-sender (parte tras la arroba en la dirección de correo del remitente).

```
dig @dklab1.casafx.dyndns.org. casafx.dyndns.org SPF +short
```

```
    "v=spf1 +mx/24 ?all"
```

```
dig @dklab2.casafx.dyndns.org. casafx.dyndns.org SPF +short
```

```
    "v=spf1 +mx/24 ?all"
```

Era un ejemplo simple pero compatible y suficiente con nuestro despliegue. La interpretación del contenido de este registro SPF sería:

- Versión 1 de spf (v=spf1). Un correo cuyo domain-sender sea casafx.dyndns.org podrá (+) ser enviado desde la subred formada por la IP de/de los registros MX correspondientes a casafx.dyndns.org enmascarando sus 3 últimos octetos (mx/24). Para cualquier otro caso (all) se mantendrá un comportamiento neutral (?, si bien tras un período de pruebas pudiéramos cambiarlo a "-all", para denegar si no hubo coincidencia con los mecanismos anteriores). Nótese que al definir la regla utilizando mx en lugar de, directamente, una IP, la regla se puede integrar con escenarios cuyo DNS tiene varis vistas.

- El registro SPF es relativamente reciente y algunos MTA, a pesar de soportar el mecanismo SPF, no leen desde un RR SPF sino los antiguos TXT. En un entorno en producción se sigue recomendanda ofrecer ambos (así ocurre en el nuestro).

Podemos hacer una prueba de los registros utilizando el intérprete spfqtool: Nota: spfqtool, al menos la versión en debian 6, no soporta SPF RR, sino sólo TXT, según hemos comprobado.

Si el sender es umea@casafx.dyndns.org y la IP de origen es 10.168.1.1:

```
spfqtool -i 10.168.1.1 -s umea@casafx.dyndns.org \
         -h dklab1.casafx.dyndns.org
```

```
SPF short result:   pass
SPF verbose result: policy result: [pass] from rule [mx/24]
RFC2822 header:     Received-SPF: pass (dklab1.casafx.dyndns.org: domain of
                    umea@casafx.dyndns.org designates 10.168.1.1 as permitted
                    sender) receiver=dklab1.casafx.dyndns.org;
                    client_ip=10.168.1.1; envelope-from=umea@casafx.dyndns.org;
```

Pero si la IP de origen no casa con MX/24:

```
spfqtool -i 1.2.3.4 -s umea@casafx.dyndns.org \
         -h dklab1.casafx.dyndns.org
```

```
SPF short result:   neutral
SPF verbose result: policy result: [neutral] from rule [?all]
RFC2822 header:     Received-SPF: neutral (nobody: domain of
                    umea@casafx.dyndns.org is neutral about designating
                    1.2.3.4 as permitted sender)
```

...si nuestra regla spf acabase en "-all" en lugar de "?all", el resultado sería "fail" y no "neutral":

```
SPF short result:   fail
SPF verbose result: policy result: [fail] from rule [-all]
RFC2822 header:     Received-SPF: fail (dklab1.casafx.dyndns.org: domain of
                    umea@casafx.dyndns.org does not designate 1.2.3.4 as
                    permitted sender) receiver=dklab1.casafx.dyndns.org;
                    client_ip=1.2.3.4; envelope-from=umea@casafx.dyndns.org;
```

Nota: El MTA puede añadir una cabecera con los resultados de SPF, por ejemplo:

```
Received-SPF: neutral (google.com: 79.155.184.237 is neither permitted
              nor denied by best guess record for domain of
              umea@casafx.dyndns.org) client-ip=79.155.184.237;
```

Nota Sender-ID: Los servicios de correo de Microsoft no utilizan SPF sino un sistema similar llamado SenderID (la diferencia es que éste no opera exactamente durante el comando MAIL FROM de SMTP). Para darles la oportunidad de hacer una verificación de origen, básicamente habría que declarar nuestro registro SPF en su web:
https://support.msn.com/eform.aspx?productKey=senderid&page=\
support_senderid_options_form_byemail&ct=eformts&scrx=1

Ref: http://www.digitalsanctuary.com/tech-blog/debian/setting-up-spf-senderid-domain-keys-and-dkim.html

DKIM DKIM es un sistema más complejo que SPF: no utiliza la IP de origen, sino una firma digital, encontrándose la llave pública para verificarla, y no una mención a direcciones IP, en DNS. Conjuntamente, el sistema MTA que manda el correo debería entonces firmarlo con la correspondiente llave privada.

```
dig @dklab1.casafx.dyndns.org exim4dkim._domainkey.casafx.dyndns.org. TXT +short
```

```
"v=DKIM1\; s=email\; g=*\; t=y\; k=rsa\; p=MIGfMA0GCSqG..."
```

```
dig @dklab1.casafx.dyndns.org _domainkey.casafx.dyndns.org. TXT +short
```

```
"t=y\; o=~\;"
```

```
dig @dklab1.casafx.dyndns.org _adsp._domainkey.casafx.dyndns.org. TXT +short
```

```
 "dkim=unknown"
```

El primer registro, con la llave, indica:

- v=DKIM1 versión DKIM utilizada.
- s=email tipo de servicio email (aún no hay definido otro, luego equivale a '*')
- g=* para todas la posibles partes-locales en el remitentes, es decir en From: <local_part>@<domain>, cualquier <local_part>.
- t=y:s flag "s" e "y", es decir esa llave no es válida para subdominios y, respectivamente, anuncia que el sistema está en modo de pruebas y por tanto deberían generarse mensajes adicionales.
- k=rsa tipo de sistema criptográfico
- p="..." llave criptográfica que extraímos

El segundo y tercer registro TXT de DKIM (viejo OSP y nuevo ADSP), indican la política de firmado que tenemos:

- t=y specifies that you are in test mode and this should be removed when you are certain that your domain key setup is functioning properly.
- o=~ specifies that some of the mail from your domain is signed, but not all. You could also specify o=- if all of the mail coming from your domain will be signed.
- dkim=unknown mismo significado que o=~ pero en registro ADSP, el equivalente a o=- en ADSP sería dkim=all, pero por precaución utilizamos unknown al menos durante un período de pruebas.

No parecen existir paquetes sencillos para probar DKIM (véanse no obstante opendkim y dkim-filter), la prueba definitiva sería comunicándonos con otros servidores ajenos a nuestro despliegue pero, aún así, podemos llegar bastante lejos usando las capacidades de depuración de exim4:

Verificación DKIM (correo entrante) Cuando un correo entrante es verificado, aparece en el log de exim4 algo como:

```
DKIM: d=gmail.com s=gamma c=relaxed/relaxed a=rsa-sha256 [verification succeeded]
```

Localmente, podemos utilizar la posibilidad de exim4 para recibir una sesión SMTP desde stdin; así simularemos que ha llegado un correo de gmail.com con una cabecera DKIM-Signature inventada:

```
vim ~/mailtest/receive_dkim.sh
```

```
(sleep 2; \
printf "EHLO smtp.gmail.com\r\n" | tee /dev/stderr; sleep 1; \
printf "MAIL FROM: cocnarf@gmail.com\r\n" | tee /dev/stderr; sleep 1; \
printf "RCPT TO:aemu@casafx.dyndns.org\r\n" | tee /dev/stderr; sleep 2; \
printf "DATA\r\n" | tee /dev/stderr; sleep 1; \
printf "Subject: dkim\r\n" | tee /dev/stderr; \
printf "From: cocnarf@gmail.com\r\n" | tee /dev/stderr; \
printf "To: aemu@casafx.dyndns.org\r\n" | tee /dev/stderr; \
printf "DKIM-Signature: v=1; a=rsa-sha256; q=dns/txt;
c=relaxed/relaxed; d=casafx.dyndns.org; s=exim4dkim;
h=Content-Type:MIME-Version:Message-ID:Subject:To:From:Date;
bh=mxBrUUox23UseRX2N7nW3GvifUa4W01XXlWCe9ZcMtQ=;b=
jcbip6cxlU5ykdDkMStiOWkW50REOOtyamDMGF4qmvFZAHFkY0G8a+g12aCEx
/YJsM2mKmWkfMCxFnYpK0JDWGOasJdzo1LFSKb/NYDECFNkHmqmjfg0DbH2FlNQ50Cg6Z
Rznw1YLGw0sotWQiwsdGkSuqA8hWeoRGReeFE+f4M=;\r\n" | tee /dev/stderr; \
printf "\r\n" | tee /dev/stderr; \
printf "\r\n" | tee /dev/stderr; \
printf "Hola cara de bola\r\n" | tee /dev/stderr; \
printf ".\r\n" | tee /dev/stderr; \
printf "QUIT\r\n" | tee /dev/stderr ) \
| exim4 -bh `dig smtp.gmail.com A +short | tail -n 1`
# smtp.gmail.com, es decir 209.85.229.108
```

```
sh ~/mailtest/receive_dkim.sh 2>&1 | grep DKIM
```

... y veremos aparecer información del proceso de verificación que lleva a cabo exim4. Puesto que la cabecera DKIM-Signature es inventada, acaba fallando, pero el despliegue en sí parece funcionar:

```
...
LOG: 1RSkKI-0001nW-Iq DKIM: d=casafx.dyndns.org s=exim4dkim c=relaxed/relaxed
     a=rsa-sha256 [verification failed - signature did not verify (headers
     probably modified in transit)]
...
```

Firmado DKIM (correo saliente) Por su lado, la cabecera que exim4 añadirá al firmar correos salientes, tendrá la forma:

```
DKIM-Signature: d=casafx.dyndns.org s=exim4dkim <signature>
```

...es decir, la información suficiente (dominio y selector) para hacer la petición dns que devuelva la llave pública con que verificar <signature>. Normalmente se añade algo más como número de versión y otros.

Una forma algo compleja de comprobar que firma los correos salientes, es mandar un correo y ver la salida que produce exim4 en modo depuración (-d+all):

Este script hará SMPT-AUTH y mandará un correo a una cuenta no gestionada por nosotros, como por ejemplo bit-bucket@test.smtp.org (como su nombre sugiere, es una cuenta libremente disponible para este tipo de pruebas):

```
cat <<EOF > ~/mailtest/send_dkim.sh
(sleep 2; \
printf "MAIL FROM: umea@casafx.dyndns.org\r\n" | tee /dev/stderr; sleep 1; \
printf "RCPT TO:bit-bucket@test.smtp.org\r\n" | tee /dev/stderr; sleep 2; \
printf "DATA\r\n" | tee /dev/stderr; sleep 1; \
printf "Subject: dkim\r\n" | tee /dev/stderr; \
printf "From: umea@casafx.dyndns.org\r\n" | tee /dev/stderr; \
printf "To: bit-bucket@test.smtp.org\r\n" | tee /dev/stderr; \
printf "\r\n" | tee /dev/stderr; \
printf "Testing dkim signing.\r\n" | tee /dev/stderr; \
printf ".\r\n" | tee /dev/stderr; \
printf "QUIT\r\n" | tee /dev/stderr ) \
| smtptest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
  -f /dev/stdin dklab1.casafx.dyndns.org
EOF
```

```
cp ~/mailtest/send_dkim.sh /tmp
```

Con el scrip anterior vamos a mandar un correo (con el objetivo de que sea firmado por el código DKIM de exim4), pero nuestra configuración actual hace obligatoria la autenticación y, sabemos, aún no tenemos disponible al proceso encargado de ella, dovecot-auth. Podríamos deshabilitar la necesidad de hacer SMTP-AUTH ahora, pero vamos a tomar otro camino más sencillo e interesante. Vamos a utilizar temporalmente (sólo para esta prueba) la alternativa de exim4 para poder autenticar con SASL-GSSAPI, es decir, su enlace con la librería de cyrus-sasl en lugar de su interfaz a dovecot-auth. Ésto nos va a permitir dar un ejemplo de configuración de esta alternativa, además de apenas modificar los ficheros de configuración. Para ello escribirmos la configuración de uso con cyrus-sasl bajo /tmp/ y utilizando sed haremos que el ".include" relativo a la autenticación apunte temporalmente a él en lugar de al fichero definitivo bajo /etc/exim4/.

```
cat <<EOF > /tmp/exim4-local-auth-40-ldapusers-cyrussasl
#http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch33.html
gssapi_sasl_server:
 driver = cyrus_sasl
 public_name = GSSAPI
 server_set_id = \${auth1}
 server_advertise_condition = yes
# Otras:
 server_mech = GSSAPI
 server_service = smtp
# En server_hostname, puedo poner dklab1.casafx.dyndns.org o:
 server_hostname = \${primary_hostname}
# En server_realm, CASAFX.DYNDNS.ORG o:
 server_realm = \${domain}
EOF

sed -i 's/\.include.*local-auth.*dovecotsasl'/\
'\.include \/tmp\/exim4-local-auth-40-ldapusers-cyrussasl/g' \
/etc/exim4/exim4.conf.template
```

Lanzamos una instancia de exim4 con soporte de depuración y sin soporte AFS (pues no lo necesitamos a la vez que impide ver los mensajes de depuración en la consola). Para ello en lugar de utilizar el script init modificado, utilizamos el original que salvaguardamos antes de aquellas modificaciones.

Paramos la instancia actual:

```
invoke-rc.d exim4 stop
```

Indicamos que se use el código de depuración en la siguiente ejecución

```
update-exim4defaults --commonoptions '-d+all'
```

Exim4 directamente (en lugar de dovecot-auth) debe poder leer el keytab por lo que lo hacemos accesible al uid Debian-exim y lo apuntamos con la variable KRB5_KTNAME

```
chown Debian-exim /etc/keytab.d/dovecot.keytab
KRB5_KTNAME=/etc/keytab.d/dovecot.keytab ~/mailtest/etc-init.d-exim4 start 2>&1\
 | grep -i dkim
```

Con lo anterior, una instancia de exim4 está cargada y mandando mensajes en su terminal. Entonces, en otra consola distinta, hacíamos al usuario umea mandar el correo a través del script anterior:

```
login umea
sh /tmp/send_dkim.sh
exit
```

Si volvemos a la consola con exim4, veremos, entre otros:

```
...
dkim-signature:v=1;{SP}a=rsa-sha256;{SP}q=dns/txt;{SP}c=relaxed/relaxed;
{SP}d=casafx.dyndns.org;{SP}s=exim4dkim;
{SP}h=To:From:Subject:Message-Id:Date;{SP}bh=JWj+...
...
```

Todo correcto, firma el correo saliente. Pulsamos Control-c en la consola con exim4 para pararlo. Ya entonces, para volver el sistema al estado inicial:

```
chown root /etc/keytab.d/dovecot.keytab
```

```
ls -l  /etc/keytab.d/dovecot.keytab
  -rw------- 1 root root 1086 2011-12-20 17:10 /etc/keytab.d/dovecot.keytab
```

```
sed -i 's/\.include.*local-auth.*cyrussasl'/\
'\.include \/etc\/exim4\/local-auth-40-ldapusers-dovecotsasl/g' \
/etc/exim4/exim4.conf.template

update-exim4defaults --force --commonoptions ''
invoke-rc.d exim4 start
```

Nota: es conocida la forma de probar un despliegue dkim para el correo saliente man-

dando, un correo a check-auth@verifier.port25.com y esperando los resultados en un mail de respuesta. En un despliegue en producción, quizás sea posible recibir ese correo respuesta y por tanto sea interesante conocer este tipo de prueba. El contenido del correo respuesta parece ser algo como:

```
Summary of Results

==========================================================

SPF check: pass

DomainKeys check: pass

DKIM check: pass

Sender-ID check: pass

SpamAssassin check: ham
```

Enrutamiento del correo saliente

Nota introductoria sobre los tests de enrutamiento En terminología de exim4, "dominios locales" son todas aquellos domain-part (lo que sigue a la "@" en una dirección de correo) para los que se aceptará el correo y será entregado, en general, aquí (correo entrante). Estos dominios son, por defecto, localhost y el <FQDN>de la máquina. Nosotros añadimos otro, "casafx.dyndns.org", de hecho la pregunta "Other destinations for which mail is accepted:" al instalar exim4 nos pedía adelantar ésto y así lo introducimos.

```
exim4 -bP +local_domains
```

```
  @:localhost:casafx.dyndns.org
```

...el valor de la variable lo toma del contenido de la macro LOCAL_DOMAINS, que es inicializada a partir del /var/lib/exim4/config.autogenerate.

```
grep LOCAL_DOMAINS= /var/lib/exim4/config.autogenerated
```

```
  @:localhost:casafx.dyndns.org
```

Nota: "@" en ese domainlist corresponde al FQDN según la documentación.

```
exim4 -bP primary_hostname
```

El contenido de esta variable, +local_domains, da sentido a toda la secuencia de routers por las que pasa un correo para decidir su ruta y destino. Veamos; en orden descendente los routers serían:

- Un primer grupo de routers se encargan de comprobar que el domain-part no pertenece a +local_domains y es, entonces, un correo saliente. Son routers predefinidos por el paquete de exim4 en debian, que anteceden a los nuestros, y que pretendemos testear en este apartado.
- El segundo grupo de routers son los que hemos diseñado en este despliegue, y encamina los correos destinados a una cuenta con domain-part casafx.dyndns.org. Además si uno de estos correos no llega a coincidir completamente con ningún router, es rebotado (y no sigue al resto de routers del tercer grupo).
- El tercer y último grupo de routers, comprueban que el domain-part pertenezca a +local_domains, entonces hacen una entrega local. Por tanto, en la práctica, se ocupan de correos con domain-part "localhost" y el <FQDN>. Como los del primer grupo, son routers predefinidos por el paquete de exim4 en debian.

Nota: si se intenta enviar un correo a un destinatario no cualificado (sin indicar @<domain-part>) exim4 lo reescribe para sufijar con @${qualify_domain}, siendo qualify_domain una variable de exim4. Si rastreamos la configuración, esa variable recibe como valor el macro ETC_MAILNAME, que es inicializado con el contenido del fichero /etc/mailname. La pregunta "System mail name:" al instalar exim4 se refería a este contenido, y convenientemente respondimos el FQDN de la máquina. Conclusión: un destinatario sin cualificar se cualifica con @dklab1.casafx.dyndns.org en esta máquina. ([[http://wiki.debian.org/EtcMailName][re

```
cat /etc/mailname
#...fue incorporado a la configuracio'n, veremos, como ETC_MAILNAME:
```

```
  dklab1.casafx.dyndns.org
```

```
egrep '(^ETC_MAILNAME|quali.*ETC_MAILNAME)' /var/lib/exim4/config.autogenerated
```

```
ETC_MAILNAME=dklab1.casafx.dyndns.org
qualify_domain = ETC_MAILNAME
```

Tests del correo saliente Vamos a comprobar el buen funcionamiento de los routers del primer grupo; acabamos de decir que deciden si un correo es saliente (y si es así lo envían al transport remote_smtp -todos, por cierto-). Vamos a describir muy esquemáticamente la secuencia de estos routers predefinidos y las condiciones adicionales que diferencian a cada uno de los demás:

```
view /etc/exim4/exim4.conf.template
```

```
...
begin routers


domain_literal:
...
```

Routers correo saliente y su condición de aplicación	
domain_literal	si tras la @ viene la dirección IP del servidor SMTP remoto
hubbed_hosts	si para ciertos dominios no usa dns sino un fichero especial
dnslookup_relay_to_domains	usa dns; es igual que el siguiente, sólo que éste no lidia
dnslookup	con una situación especial

...tras ellos vienen macros ifdef para opciones que, puesto que no las escogimos cuando debconf nos lo preguntó al instalar exim4, no se aplicarán: ¿sistema de correo sólo local?, ¿sistema satélite de un smarthost?... etc. Posteriormente están definidos los routers que hemos diseñado nosotros, y que comprobaremos más tarde. Vamos entonces a comprobar éstos para el correo saliente. Suponiendo que cern.ch no pertenece a nuestros +local_domains:

```
exim4 -bt recruitment.service@cern.ch
```

... la salida debiera ser algo como:

```
recruitment.service@cern.ch
  router = dnslookup, transport = remote_smtp
  host cernmxgwlb2.cern.ch [137.138.144.183] MX=10
```

Si queremos saber todos los detalles, añadimos -v -d+all:

```
exim4 -v -d+all -bt recruitment.service@cern.ch 2>&1 | less
```

Nota: un correo desde dklab1 a dklab2 fallará (incluso tras desplegar el sistema de correo en éste último).

```
exim4 -v -f umea@casafx.dyndns.org -bt umea@dklab1.casafx.dyndns.org
```

```
...
R: dnslookup for aemu@dklab1.casafx.dyndns.org
aemu@dklab1.casafx.dyndns.org is undeliverable: Unrouteable address
```

La razón es que el router dnslookup ignora la resolución de dominios para ciertos

segmentos de red predefinidos:

```
grep -B10 -A10 ignore_target_hosts /etc/exim4/exim4.conf.template|less
```

```
  dnslookup:
    ...
    ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8 : 192.168.0.0/16 :\
                          172.16.0.0/12 : 10.0.0.0/8 : 169.254.0.0/16 :\
                          255.255.255.255
    no_more
    ...
```

... corresponde al administrador hacer o no modificaciones a esa lista.

Enrutamiento del correo a cuentas centralizadas Nos centramos ahora en correos enviados a cuentas cuyo domain-part es casafx.dyndns.org. Los routers que se evalúan son los que hemos diseñado en este despliegue, y que eran incluídos en /etc/exim4/exim4.conf.template tras los del correo saliente.

```
...
begin routers


domain_literal:
...
.include /etc/exim4/local-router-250-ldap
...
```

Para poder conocer con total exactitud el comportamiento de nuestro código (string-expansion etc) le pasaremos a exim4 los flags -v y, sobre todo, -d+all, activando el modo depuración. Además, siempre es conveniente pasar -f para que tenga disponible un remitente.

```
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt aemu@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt umea@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt postmaster@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt noexiste@casafx.dyndns.org 2>&1 | less
```

Enrutamiento del correo a cuentas locales en dklab1 Nos centramos ahora en correos enviados a cuentas cuyo domain-part es localhost, el FQDN o, equivalentemente a éste por el sistema de cualificación que explicamos, nada. Previamente, describimos muy esquemáticamente la secuencia de estos routers predefinidos y las condiciones que aplican:

```
...
begin routers


domain_literal:
...
.include /etc/exim4/local-router-250-ldap
...
real_local:
...
```

Routers para el correo a cuentas locales y su condición de aplicación	
real_local	si es dominio local _y_ usa sistema prefijos ${local_part}-real; no importa
system_aliases	si te encuentra en /etc/aliases
userforward	si encuentra ~/.forward
procmail	si encuentra ~/.procmail
maildrop	si encuentra ~/.mailfilter
lowuid_aliases	caso especial de aliases para entidades no persona
local_user	es un dominio local, y es un usuario local "check_local_user"
mail4root	el caso anterior especializado en el usuario “root”.

... local_user es el router cuyo transport almacena el correo en /var/mail/<local-

part>, y es el destino habitual de los correos a cuentas locales.

Nota: las cuentas de shell según /etc/passwd en nuestro testbed son "root" (la cuenta privilegiada uid 0) y "user":

```
getent passwd root user
```

```
root:x:0:0:root:/root:/bin/bash
user:x:1000:1000:Debian User,,,:/home/user:/bin/bash
```

Ya podemos efectuar las pruebas a cuentas con domain-part dklab1.casafx.dyndns.org:

```
exim4  -bt user@dklab1.casafx.dyndns.org
exim4  -bt root@dklab1.casafx.dyndns.org
```

...o, si queremos ver todos los detalles, añadimos -v -d+all:

```
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt user@dklab1.casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt root@dklab1.casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt umea@dklab2.casafx.dyndns.org 2>&1 | less
```

Prueba a cuentas sin domain-part que, puesto que serán cualificadas con @dklab1.casafx.dyndns.or se comportan igual que el caso anterior:

```
exim4 -bt user
```

Prueba a cuentas con domain-part localhost:

```
exim4 -bt user@localhost
```

Por último, y no menos importante, un correo a umea (cuenta centralizada, resolvible por NSS a través de ldap) cuando domain-part no es casafx.dyndns.org, no se tratará como un correo a cuenta centralizada sino como local y, entonces, se entrega en /var/mail/umea:

```
exim4 -bt umea
```

Para ver todos los detalles:

```
exim4 -d+all -v -f umea@casafx.dyndns.org -bt umea 2>&1 | less
```

Así pues, un correo para umea@casafx.dyndns.org, cuyo domain-part es casafx.dyndns.org, será aceptado por los routers para cuentas centralizadas. Un correo para umea@dklab1.casafx.dyndns.org no será aceptado por ellos, pero entonces pasará por los subsecuentes routers para correo de entrega local y será aceptado allí, de la misma forma que sería aceptado uno para root@dklab1.casafx.dyndns.org o root@localhost. Por tanto la local-part, siempre que pueda ser resuelta por el sistema NSS (y éso se hace, según nuestro /etc/nsswitch, a través de ldap y /etc/passwd), podrá ser aceptada por los routers posteriores que gestionan el comportamiento local, es decir correos a los dominios dklab1.casafx.dyndns.org y localhost.

Gestión de cola; monitorización Un SMTP MTA contempla la existencia de una cola de mensajes, de forma que ante algún problema en la entrega, el correo pueda quedar temporal o permanentemente retenido en ésta. Exim4 provee opciones para monitorizar y controlar su cola, veamos algunos ejemplos:

```
mailq
```

... en este momento la cola estará vacía, no obstante una salida típica podría ser:

```
 3m  2.6K 1RRPUG-0001IE-Ah <> *** frozen ***

          umea@casafx.dyndns.org
```

Donde:

- 1RRPUG-0001IE-Ah corresponde al identificador del correo.
- Frozen es su estado (en este ejemplo, congelado).
- umea@casafx.dyndns.org es el remitente.

Por su lado, runq lanza manualmente un ciclo de reintentos de cola (independientemente del ciclo automático)

```
runq
```

Otras posibilidades:

```
exim4 -bp       # Equivale a mailq, es decir muestra un sumario de la cola.
exim4 -q [<id1>...] # Equivale a runq, es decir lanza un reintento de
                    # todos o el correo indicado de la cola
exim4 -Mvh <id> # Ver cabeceras
exim4 -Mvb <id> # Ver cuerpo
exim4 -M   <id> # Intentar reenviar
exim4 -Mrm <id> # Eliminar. Para eliminar todos:
exim4 -bpru | awk {'print $3'} | xargs exim4 -Mrm
exim4 -Mf  <id> # Marcar frozen
exim4 -Mt  <id> # Desmarcar frozen
exim4 -Mmad mark all delivery # Marcar todos como entregados.
```

Véase también man exipick. Por ejemplo para mostrar todos los mensajes en cola:

```
exipick -i | while read L; do exim4 -Mvh $L; exim4 -Mvb $L; done
```

Aunque los comandos anteriores permiten hacer scripts de shell y automatizar tareas de gestión en la cola, puede ser tedioso usarlos manualmente. Por ello se instaló también pfqueue, una utilidad en línea de comandos con interfaz interactiva. Al ejecutarlo, aparece una lista con los <id>, "From" y "To" del contenido de la cola. Entonces, pulsando "?" podemos obtener una lista de acciones, y sus atajos de teclado, que efectuar. Un ejemplo de uso útil en nuestras pruebas: con los cursores podemos seleccionar un correo, pulsando "RETURN" podemos ver el cuerpo del correo ("space" para avanzar hacia el final), con "RETURN" terminamos de inspeccionarlo y con "d" lo borramos de la cola. Véase man pfqueue.

```
pfqueue
```

El diseño de exim4 concibe el encolamiento de un correo como algo excepcional, por tanto la entrega desde la cola es más ineficiente que en otros MTA, donde existe específicamente un programa para la entrega desde la cola.

Respecto a la monitorización:

```
tail -f /var/log/exim4/mainlog
exiwhat # pregunta a exim4 en por lo que le mantiene ocupado ahora.
geximon # supera al antiguo eximon4, es un panel en GTK en el que se ve
        #la entrada/salida de correos, la cola y otras funcionalidades.
```

Estadísticas Conforme exim4 vaya gestionando correos reales, /var/log/exim4/mainlog irá registrando toda esa actividad. El comando eximstats puede inspeccionar los losgs y presentarnos algunos estadísticos:

```
eximstats /var/log/exim4/mainlog* | less
```

También adminte otros formatos de salida, como html. Para proveer de gráficos a la salida html:

```
apt-get install libgd-gd2-noxpm-perl libgd-text-perl libgd-graph-perl
eximstats -html \
          -charts -chartdir /tmp/eximstats.charts \
          -chartrel /tmp/eximstats.charts\
          /var/log/exim4/mainlog* > /tmp/eximstats.html


w3m /tmp/eximstats.html
```

1.2.9. MDA, IMAP y MANAGESIEVE

Un mismo proyecto, Dovecot, proveerá todas las funcionalidades (IMAP server, MANAGESIEVE server y componente MDA del lado del servidor). Adicionalmente mostraremos las capacidades de MDA de otro software, procmail, también compatible con AFS y configurado como delivery opcional en nuestro despliegue de exim4 ("ldap_pipe" router, metadato en ldap "deliveryProgramPath") mostrando así otro estilo de entrega, esta vez bajo el home del usuario.

Sistema de configuración dovecot

Al contrario que con exim4, el comportamiento de dovecot es más convencional: /etc/dovecot/dovecot.conf controla el grueso de la configuración y, por convenio, las opciones comentadas en ese fichero, es decir precedidas por '#', indican el valor por defecto,

el que se usará si no se indica otra cosa. Por otro lado /etc/dovecot/dovecot-ldap.conf controla específicamente lo relativo al uso de ldap.

Adicionalmente, dejemos que dovecot en debian se presente a sí mismo, con:

```
dovecot --version

dovecot --build-options

getent passwd dovecot
```

En versiones posteriores se espera el soporte de la extensión Énotify XMPP'de Sieve, de forma que se puedan enviar notificaciones al sistema de mensajería instantánea tras la llegada de un nuevo correo. Consúltese http://pigeonhole.dovecot.org/download.html.

/etc/dovecot/dovecot.conf

```
vim /etc/dovecot/dovecot.conf
```

```
####fx:
#http://wiki.dovecot.org/MainConfig


#!include /etc/dovecot/conf.d/*.conf
#!include_try /etc/dovecot/extra.conf
#Nota: Los include no funcionan con el programa delivery


#mail_debug = yes
####endfx
...
```

```
...
####fx:
# Protocolos de internet:
protocols = imap imaps managesieve # pop3 pop3s
#-protocols = imap imaps
####endfx
...
```

```
...
####fx:
# Logging: por defecto, el logging se hace a syslog;
####endfx
log_timestamp = "%Y-%m-%d %H:%M:%S "
...
```

```
...
## SSL settings
##
####fx:
# TLS: http://wiki.dovecot.org/MainConfig#SSL_settings
ssl = yes #no, required. Adicionalmente, por defecto ssl_listen
          #coincide con listen.
ssl_cert_file = /etc/dovecot/dovecot-1.casafx.dyndns.org.crt #pem format
ssl_key_file =  /etc/dovecot/dovecot-1.casafx.dyndns.org.key #pem format
##ssl_verify_client_cert = no # Existe require en lugar de verify
###The CAfile should contain the CA-certificate(s) followed by the matching CRL(s).
##ssl_ca_file = /etc/ssl/certs/casafx-ca.crt
#verbose_ssl = no
ssl_parameters_regenerate = 0 # evita que el proceso ssl-build-params
                              # (por defecto cada semana), regenere DH.
                              # De utilidad si el ordenador/emulador es
                              # lento.
####endfx
...
```

```
...
#verbose_ssl = no
####fx:
#UID's: dovecot es -1. Lanzado como root (uid 0)
#                  -2. Ciertas acciones del procesamiento del login se hacen
#                      como user dovecot (o lo que diga "login_user"). Nada en
#                      el sistema debiera tener permiso para este usuario
#                      excepto lo que pueda crear el servidor. No pertenece a
#                      otro grupo que al suyo y a su grupo no pertenece nadie,
#                      , ni puede ser usado tampoco para los otros roles:
login_user = dovecot
#                  -3. El acceso al correo se hace con otro usuario,
#                      cuyo uid/gid se puede configurar ahora y ser el
#                      valor por defecto si no se especifica otra cosa
#                      por seccio'n auth {} y, alli', por usuario en
#                      el metadato userdb:
mail_uid = vmail
#mail_gid = vmail #Adicionalmente, para uid y gid se puede limitar por uid:
first_valid_uid = 800 #8 corresponde a la cuenta local "mail", pero
                     #nosotros usamos vmail 800.
last_valid_uid =  800
first_valid_gid = 800
last_valid_gid =  800
```

```
 #Acorde con la debian policy, el tradicional directorio spool del
 #correo unix (/var/mail) debe tener como grupo a "mail" y permisos de
 #grupo rws, de forma que los MUA puedan manejar los recipientes
 #mbox si su uid pertenece al grupo "mail". Si en un futuro necesitamos
 #dar soporte IMAP o mda para esas cuentas locales, compatiblemente
 #a los uid y gid anteriores podemos indicarle a dovecot el grupo
 #para esa situacio'n concreta:
#http://www.debian.org/doc/debian-policy/ch-customized-programs.html#s-mail-transpo
mail_privileged_group = mail
#                   -4. Finalmente, dovecot hace las resoluciones para
#                       userdb/passdb como root, o lo que diga "user" en
#                       seccio'n de auth (ver luego)
####endfx
...
```

```
...
## Login processes
##
####fx:
login_process_per_connection = yes # more secure vs faster
disable_plaintext_auth = yes
max_mail_processes = 512  #when this limit is reached, new users aren't
                          #allowed to log in
mail_process_size = 256   #in megabytes. Most of the memory goes to
                          #mmap()ing files
####endfx
...
```

```
## Mailbox locations and namespaces
##
####fx:
#Mail location: http://wiki.dovecot.org/MailLocation
 # mail_location define el valor por defecto, pero puede ser redefinido
 # en en la seccio'n auth {} por el metadato mail de la userdb
 # (adicionalmente, puede sobreescribirse si se utilizan imap
 # namespaces, no es el caso -ver wiki- pero sirven para mezclar
 # formatos o compartir algunas mailboxes
 # http://wiki.dovecot.org/SharedMailboxes/Public).
 # Los i'ndices se crean en el directorio maildir si no se dice otra cosa,
 # aunque puede redefinirse otra ruta para ellos, incluso almacenarse en memoria.
 #mail_location = maildir:~/Maildir
 #mail_location = mbox:~/mail:INBOX=/var/mail/%u
 #mail_location = maildir:~/Maildir:INBOX=~/Maildir/.INBOX
 #mail_location =
 #     maildir:/afs/cell/home/%u/Maildir:INDEX=/var/lib/dovecot-index/%u
mail_location = maildir:/afs/%Ld/service/mail/%1n/%2n/%n/Maildir
#
# AFS support:
#http://nocones.blogspot.com/2007/05/afs-postfix-dovecot-and-maildirs.html
#http://wiki.dovecot.org/MainConfig#Mail_processes
#http://wiki.dovecot.org/MainConfig#Maildir-specific_settings
#http://wiki.dovecot.org/NFS
```

```
 #Keep indexes out of AFS
 #...no puedo, creo, poner el index en memoria ni localmente, pero al fin
 #y al cabo los i'ndices son archivos comunes, solamente
 #se penaliza el desempen~o que se intenta ganar (ve'ase discusio'n en
 #http://wiki.dovecot.org/NFS)

 #Disable link() copies:
maildir_copy_with_hardlinks = no

 #Next, disable mmap and enable dotlocks. I would think you shouldn't need
 #these but this is what I ended up with that worked. Test yourself and see:
mmap_disable = yes
dotlock_use_excl = yes
lock_method = dotlock
#maildir_very_dirty_syncs = yes # Assume Dovecot is the only MUA
                                # accessing Maildir:
 #                                Scan cur/ directory only when its
 #                                mtime changes unexpectedly or when we
 #                                can't find the mail otherwise.
####endfx
...
```

```
...
# If you need to set multiple mailbox locations or want to change default
# namespace settings, you can do it by defining namespace sections.
####fx:
# Imap Namespaces: http://wiki.dovecot.org/Namespaces
# ( http://www.faqs.org/rfcs/rfc2342.html )
# Aqui' se van a usar para ofrecer mailboxes pu'blicas:
# http://wiki.dovecot.org/SharedMailboxes/Public
# Por defecto las mailboxes residen en un namespace privado que dovecot crea
# automa'ticamente. Para poder tener mailboxes de acceso pu'blicas, es necesario
# primero explicitar el namespace privado (por alguna razo'n no se hace ahora
# automa'ticamente) y a continuacio'n hay que declarar un namespace "Public"
# para las nuevas mailboxes pu'blicas:
namespace private {
  # Simplemente lo configuramos para que no cambie nade respecto a si no
  # estuvie'semos definiendo namespace's adicionales. Asi' pues, la
  # localizacio'n del correo en este espacio es la definida antes en
  # mail_location
  separator = /
  prefix =
  inbox = yes # so'lo un namespace puede afirmar tener la inbox del usuario
}
#... podri'amos definir otros namespace private {}, por ejemplo para tener
#    mailboxes en otro formato (ejem mbox). No ejemplificaremos ese caso.
```

```
#    Vayamos ya al namespace public:
namespace public {
  # Si'mbolo para separar los namespaces en jerarqui'a; debe ser el
  # mismo en todos.
  separator = /
  # Si se debiera listar este namespace si el cliente envi'a el comando imap NAMES
  hidden = no
  # Si hidden = no, que' nombre devolver para este namespace:
  prefix = Public/
  # Si este namespace y sus mailboxes debieran ser listadas ante
  # un comando LIST:
  list = yes # "children" hari'a que listase so'lo si el espacio
             # realmente contiene mailboxes.
```

Si este espacio gestiona las subscripciones a sus mailboxes o ésto se gestiona en el espacio padre (en este caso "/", en el namespace privado) lo diferenciamos con 'subscriptions = yesó 'subscriptions = no'respectivamente. En despliegues donde "Public/" corresponde a una ruta en el sistema de archivos en que el proceso que accede no puede escribir, (ej: deliver u otro sistema escribe al hacer la entrega, pero imap no puede) entonces eligir 'subscriptions = no'permite cambiar la ruta a la del espacio "/", donde presumiblemente sea posible crear y escribir un fichero 'subscriptions'(ej: imap puede escribir en el home del usuario). Pero nuestro caso no es típico, veamos entonces: la ruta para el espacio raiz "/" (/afs/<realm>/service/mail/<dobleNivelUser>) es escribible, pero la del espacio "Public/" también porque aunque la ruta para las mailboxes no lo es (/afs/<realm>/service/Public), la de los ficheros de control de ese espacio los declararemos separadamente bajo el home del usuario en un directorio escribible (/afs/<realm>/user/<dobleNivelUser>/.PublicMailboxIndexes).

Por tanto en nuestro caso ambas posibilidades permiten al usuario gestionar sus subscripciones porque imap puede a su vez gestionar un fichero 'subscriptionsén ambas localizaciones. Nos parece pertinente entonces que las subscripciones a este espacio se gestionen en la ruta de control de este espacio como siempre, así que elegimos:

(La sentencia "location.ªparece dividida en 3 partes delimitadas por \ y cambio de línea. Realmente debe aparecer en una sóla línea, sin \ y cambio de línea:)

```
  subscriptions = yes
  # Rutas para las mailboxes y rutas para los archivos de control:
  # nota: hay que ponerlo en una so'la li'nea, sin usar el escape "\"
  location = maildir:/afs/%Ld/service/mail/Public:\
             CONTROL=~/.PublicMailboxIndexes:\
             INDEX=~/.PublicMailboxIndexes

  #CONTROL=/afs/%Ld/user/%1n/%2n/%n/.PublicMailboxIndexes:\
  #INDEX=/afs/%Ld/user/%1n/%2n/%n/.PublicMailboxIndexes
}
```

```
#... podri'amos definir otros namespace public {}, por ejemplo para
#    crear una jerarqui'a bajo Public/, por ejemplo Public/SedeA
#    Public/SedeB... tambie'n existen namespace shared {} para
#    compartir entre grupos reducidos. No ejemplificaremos ambos casos,
#    pero lo anterior es suficiente para hacerse una idea y evaluar si
#    son necesarios en otro despliegue.
####endfx
...
```

...así con esas rutas, como se adelantó al hablar de las subscripciones:

- las mailboxes de este espacio estarán bajo /afs/<realm>/service/mail/Public, una carpeta no modificable por los usuarios, imap o deliver.
- Por ese carácter de sólo lectura los índices y archivos de control los hemos movido a un lugar escribible por imap (probablemente mejorará el desempeño si está en otro volumen afs, como el del home), /afs/<realm>/user/<DobleNivelUser>/.PublicMailboxIndexe donde los servicios de correo (grupo afs "mailgroup") sí pueden escribir.
- Nótese que el fichero 'subscriptionsén esa ruta bajo AFS, puede ser convertido en de sólo lectura y, así, fijar unas subscripciones al usuario para el espacio "Public/". Para ello consúltese la relación entre ACL afs y posix en http://docs.openafs.org/UserGuide/ch04s08. y revísese el capítulo que presentaremos más tarde para crear los recursos de almacenamiento en AFS.

```
protocol imap {
####fx:
# IMAP protocol: http://wiki.dovecot.org/MainConfig#IMAP_specific_settings
  listen = * #*,[::]
  login_executable = /usr/lib/dovecot/imap-login
  mail_executable = /usr/lib/dovecot/imap
  #mail_executable = /usr/lib/dovecot/rawlog /usr/lib/dovecot/imapn
  #mail_executable = /usr/lib/dovecot/gdbhelper /usr/lib/dovecot/imap

  mail_plugins = quota imap_quota
#}
 #protocol pop3 {
 # pop3_uidl_format = %08Xu%08Xv
 # mail_plugins = quota
 #}
####endfx
...
```

```
...
protocol managesieve {
####fx:
# Managesieve protocol: http://wiki.dovecot.org/ManageSieve
#                       http://tools.ietf.org/html/rfc5804
#Configuramos lo relativo al demonio y, lo relativo al lugar de
#almacenamiento se configurara' en la seccio'n "plugins" puesto
#que es comu'n con el soporte de sieve de la funcionalidad MDA.
#Nota: supongo que managesieve se lanza con uid el de mail_uid
#http://wiki.dovecot.org/ManageSieve/Configuration
  #Daemon:
  listen = *:4190
  mail_executable = /usr/lib/dovecot/managesieve
  login_executable = /usr/lib/dovecot/managesieve-login
  #Storage: ve'ase seccio'n "plugin"
#}
####endfx
...
```

```
## MDA (aka LDA) specific settings
##
####fx:
# Delivery: http://wiki.dovecot.org/LDA
protocol lda {
  #Cabeceras para rebotar mails:
   #From:
   postmaster_address = postmaster@casafx.dyndns.org
   #Para Message-IDs and in Reporting-UA:
   hostname = dklab1.casafx.dyndns.org

  #Soporte para reenvi'os:
  sendmail_path = /usr/sbin/sendmail

  #Socket para consultar metadatos userdb al proceso de auth de dovecot
  #(ver seccio'n auth) al llamar el lda con -d user@host:
  auth_socket_path = /var/run/dovecot/auth-master
```

```
  #Log: si utilizo syslog, no tengo que plantearme si el usuario vmail
  #puede escribir en el fichero de log de dovecot; para utilizar
  #syslog, vaci'o las variables:
  log_path =
  info_log_path =
  syslog_facility = mail

  #Plugins:
  mail_plugins = quota sieve
  #...la configuracio'n de los plugins se declara posteriormente en la
  #   seccio'n plugin {}
#Nota: adicionalmente se puede usar el deliver directamente (ejem en
#.forward, ver wiki)
}
####endfx
...
```

```
auth default {
####fx:
# Subservicios auth / metadata lookup: http://wiki.dovecot.org/Authentication
  # Las bu'squedas de metadatos van a realizar con la cuenta local:
  user = root

  #http://wiki.dovecot.org/Authentication/Mechanisms
  mechanisms = gssapi # plain login cram-md5...
  auth_default_realm = CASAFX.DYNDNS.ORG
  auth_krb5_keytab = /etc/keytab.d/dovecot.keytab
  auth_gssapi_hostname = dklab1.casafx.dyndns.org
  #auth_verbose
  #auth_debug
  #auth_debug_passwords

  userdb ldap {
    args = /etc/dovecot/dovecot-ldap.conf # Alli' declaramos igualmente
                                          # los metadatos esta'ticos
  }
```

```
#
#No utilizo base de datos de password al usar el mecanismo gssapi.
#You can use multiple databases, so if the password doesn't match in
#the first database, Dovecot checks the next one.
#http://wiki.dovecot.org/PasswordDatabase
#passdb ldap {
#  args = /etc/dovecot/dovecot-ldap.conf
#}
#Prefetch metadata de userdb cuando obtienes la de passdb.
#userdb prefetch { }
#

#Exportamos a otros programas los subservicios auth / metadata lookup:
socket listen {

  #Master socket, usado por el dovecot MDA (deliver) para hacer
  #lookups:
  master {
    path = /var/run/dovecot/auth-master
    #Los permisos se configuran teniendo en cuenta que a trave's de
    #este socket se resuelven metadatos userdb; por seguridad
    #restringimos a:
    mode = 0600
    user = vmail
  }
```

```
  #Client socket, usado por el MTA (exim4) para SMTP-AUTH
  client {
    path = /var/run/dovecot/auth-client
    #Es seguro exportarlo a cualquiera, en general, en este caso
    #lo fundamental es que sea accesible para la cuenta de exim4:
    mode = 0660
    user = Debian-exim
    group = mail
  }
}
```

```
#No uso modo proxy a otros dovecot en otros hosts donde realmente
#estuviese almacenado el correo:
#http://wiki.dovecot.org/PasswordDatabase/ExtraFields/Proxy
#http://wiki.dovecot.org/HowTo/ImapProxy
####endfx
...
####fx:
#Deshabilito, au'n en "auth default {}", la configuracio'n preexistente
#de mecanismos:
#-mechanisms = plain
####endfx
...
####fx:
#Deshabilito, au'n en "auth default {}", la configuracio'n preexistente
#de passdb
#-passdb pam {
####endfx
...
####fx:
#cierro llave de la seccio'n passdb pam deshabilitada.
#-}
####endfx
```

```
...
####fx:
#Deshabilito, au'n en "auth default {}", la configuracio'n preexistente
#de userdb
#-userdb passwd {
####endfx
...
####fx:
#cierro llave de la seccio'n userdb passwd deshabilitada.
#-}
####endfx
...
}
...
```

```
...
## Dictionary server settings
##
####fx:
# Subservicio de almacenamiento de pares llave-valor (no lo uso: nada
# que hacer)
####endfx
dict {
}
...
```

```
...
## Plugin settings
##
####fx:
# Finalmente, configuracio'n de plugins:
####endfx
...
```

```
...
plugin {
####fx:
 # Q u o t a: http://wiki.dovecot.org/Quota/1.1
 # "maildirsize" format y otros tipos de quota:
 #  http://wiki.dovecot.org/Quota/Maildir
 # (imap quota spec http://www.rfc-editor.org/rfc/rfc2087.txt)
 #
 # quota = <backend>:<quota root name>:<backend args>. Ejems:
 #  quota = maildir:user quota
 #  quota2 = maildir:Shared quota:ns=Public/
 #  quota3 = fs:Disk quota
 # quota_rule = <mailbox name>:<limit conf> #siendo los li'mites:
 #storage=nKBytes bytes=nBytes messages=nMensajes backend=tipo ignore
 #  quota_rule = *:storage=1G # '*' es para cualquiera que no tenga
 #                                quota_rule definida.
 #  quota_rule3 = Trash:storage=20%%
 #  quota_rule2 = SPAM:ignore
 # quota_warning[n] = <limit configuration> <command to run>  # O bien:
 # quota_exceeded_message = Quota exceeded, please go to\
 #    http://www.example.com/over_quota_help for instructions \
 #    on how to fix this.
```

```
#
 # Vamos alla':
  quota = maildir:User quota
 #quota_rule lo definiremos por cada usuario en dovecot-ldap.conf
  quota_warning = storage=90%% /usr/local/bin/quota-warning.sh 90
 #"the warning is ONLY executed at the exact time when the limit
 # is being crossed"



# S i e v e: http://wiki.dovecot.org/LDA/Sieve/Dovecot
#            http://wiki.dovecot.org/ManageSieve
# (sieve spec      : http://tools.ietf.org/html/rfc5228)
 # sieve_before apunta a un directorio con scripts .sieve ejecutados
 # en orden alfabe'tico previamente al de usuario. Nota: Existe
 # sieve_after.
 sieve_before = /afs/casafx.dyndns.org/service/mail/sieve/
 # Enlace simbo'lico al script activo de usuario (u'nico, pero puede
 # incluir otros)
 sieve = /afs/%Ld/service/mail/%1n/%2n/%n/.dovecot.sieve
 # Do'nde se almacenan los scripts subidos (los del espacio de nombres
 # :personal):
 sieve_dir = /afs/%Ld/service/mail/%1n/%2n/%n/sieve
 # Sieve permite hacer includes a scripts en espacio :personal o en
 #espacio :global accesibles por cualquiera, sieve_global_dir dice
 #do'nde residen:
 #sieve_global_dir = <path>
 # Script que se ejecutari'a si no se encuentran scripts de usuario:
 #sieve_global_path =
```

```
  # Todas las extensiones sieve esta'n activas (excepto obsoletas),
  # pero es modificable:
  #sieve_extensions = +imapflags -otraextension

  #http://www.earth.ox.ac.uk/~steve/sieve/procmail2sieve.pl
#}
####endfx
```

/etc/dovecot/dovecot-ldap.conf

```
vim /etc/dovecot/dovecot-ldap.conf
```

```
...
####fx:
#http://wiki.dovecot.org/AuthDatabase/LDAP/Userdb
#http://wiki.dovecot.org/AuthDatabase/LDAP
#Comprobe' que au'n no es capaz de usar los DNS RR SRV para ldap
#cuando no pongo uris:
#   "dovecot: auth(default): Fatal: LDAP: No uris or hosts set"
uris = ldap://dklab1.casafx.dyndns.org ldap://dklab2.casafx.dyndns.org

#dn = uid=dovecot,dc=example,dc=com
#dnpass = dovecotpass
ldap_version = 3
base = ou=users,ou=accounts,dc=%Dd
# %d es casafx.dyndns.org y %Dd es casafx,dc=dyndns,dc=org
scope = subtree
user_filter =\
(&(objectClass=qmailUser)(objectClass=posixAccount)(uid=%n))
user_attrs = =uid=vmail, =gid=vmail,\
=mail=maildir:/afs/%Ld/service/mail/%1n/%2n/%n/Maildir,\
mailQuotaSize=quota_rule=*:storage=%$,mailQuotaCount=quota_rule=*:\
messages=%$,homeDirectory=home
#pass_attrs = ...
#pass_filter = ...
#default_pass_scheme = LDAP-MD5
####endfx
```

Creación de volúmenes, directorios y permisos para el mail storage en AFS

```
kinit -p root/admin; aklog
```

Cada usuario tendrá un volumen, con una quota ajustada a la información en ldap, para almacenar sus mailboxes:

```
vos create dklab2.casafx.dyndns.org. a service.mail.aemu -maxquota 5000
vos create dklab1.casafx.dyndns.org. a service.mail.umea -maxquota 5000
```

Además hay un volumen para las mailboxes públicas:

```
vos create dklab1.casafx.dyndns.org. a service.mail.Public -maxquota 5000
```

Los volúmenes para el correo son montados en una rama bajo service, por ejemplo para el user umea: /afs/casafx.dyndns.org/service/mail/u/um/umea, y sólo pueden escribir y leer etc los pertenicientes al afs grupo "mailgroup", es decir dovecot. En el caso especial de las mailboxes públicas será en service/mail/Public, y, además, los índices estarán en el home del usuario bajo .PublicMailboxIndexes, donde puede escribir mailgroup (a no ser que el usuario mofifique éso).

```
cd /afs/casafx.dyndns.org/service
mkdir -p mail/u/um
fs mkmount mail/u/um/umea service.mail.umea -rw
fs listacl mail/u/um/umea
fs setacl  mail/u/um/umea mailgroup all
fs setacl  mail/u/um/umea system:anyuser ""
fs setacl  mail/u/um/umea system:authuser ""
mkdir mail/u/um/umea/Maildir # al ser subdirectorio, en principio tiene
                             # los mismos permisos
mkdir ../user/u/um/umea/.PublicMailboxIndexes
fs setacl ../user/u/um/umea/.PublicMailboxIndexes mailgroup rlidwk
fs setacl ../usr/u/um/umea mailgroup l # Imprescindible

mkdir -p mail/a/ae
fs mkmount mail/a/ae/aemu service.mail.aemu -rw
fs listacl mail/a/ae/aemu
fs setacl  mail/a/ae/aemu mailgroup all
fs setacl  mail/a/ae/aemu system:anyuser ""
fs setacl  mail/a/ae/aemu system:authuser ""
mkdir mail/a/ae/aemu/Maildir
mkdir ../user/a/ae/aemu/.PublicMailboxIndexes
fs setacl ../user/a/ae/aemu/.PublicMailboxIndexes mailgroup rlidwk
fs setacl ../usr/u/um/umea mailgroup l # Imprescindible
```

```
tree -a
env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'aklog;
    env KRB5CCNAME=/tmp/.krb5cc_800_dovecot
    ls -A /afs/casafx.dyndns.org/user/u/um/umea/.PublicMailboxIndexes'
env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'aklog;
    env KRB5CCNAME=/tmp/.krb5cc_800_dovecot
    ls -A /afs/casafx.dyndns.org/user/a/ae/aemu/.PublicMailboxIndexes'

#Public
fs mkmount mail/Public service.mail.Public -rw
fs listacl mail/Public
fs setacl  mail/Public mailgroup rl      # dovecot no puede modificarla (RO)
fs setacl  mail/Public system:anyuser  ""
fs setacl  mail/Public system:authuser ""

mkdir -p mail/Public/.newsletter             # Mailbox pu'blica de
                                             # nombre newsletter
touch mail/Public/.newsletter/dovecot-shared # Su presencia hace que
                                             # los flags \Seen
                                             # _no_ se compartan por los
                                             # lectores.
touch mail/Public/dovecot-shared             # Hace que se copie en cada
                                             # mailbox nueva que creemos.



#Sieve scripts para sieve_before:
vos create dklab1.casafx.dyndns.org. a service.mail.dov_sieve -maxquota 1000
fs mkmount \
   /afs/casafx.dyndns.org/service/mail/sieve service.mail.dov_sieve -rw
fs setacl /afs/casafx.dyndns.org/service/mail/sieve mailgroup all
```



```
tree -a
```

Nota: todas las quotas anteriores pueden ser modificadas más tarde, al vuelo, con "fs setquota", véase: http://docs.openafs.org/AdminGuide/ch05s12.html

```
#AFS cron backups:
bos create -type cron -instance respaldovolmail \
           -server dklab1.casafx.dyndns.org. \
          -cmd "/usr/bin/vos backupsys -prefix service.mail -localauth" "05:00"
bos status -server dklab1.casafx.dyndns.org. -long
bos create -type cron -instance respaldovolmail \
           -server dklab2.casafx.dyndns.org. \
          -cmd "/usr/bin/vos backupsys -prefix service.mail -localauth" "05:00"
bos status -server dklab2.casafx.dyndns.org. -long

#procmail MDA opcional:
cd /afs/casafx.dyndns.org/user

mkdir u/um/umea/Maildir
fs setacl u/um/umea/Maildir umea all
fs setacl u/um/umea/Maildir mailgroup rwklid
fs setacl u/um/umea/Maildir system:anyuser ""
fs setacl u/um/umea/Maildir system:authuser ""
fs setacl u/um/umea mailgroup l # Imprescindible

mkdir a/ae/aemu/Maildir
fs setacl a/ae/aemu/Maildir aemu all
fs setacl a/ae/aemu/Maildir mailgroup rwklid
fs setacl a/ae/aemu/Maildir system:anyuser ""
fs setacl a/ae/aemu/Maildir system:authuser ""
fs setacl a/ae/aemu mailgroup l # Imprescindible
```

Sieve, adicionalmente necesita

```
vim /afs/casafx.dyndns.org/service/mail/sieve/00-global.sieve
```

```
require "fileinto";
 if header :contains ["X-Spam-Flag"] ["Yes"] {
   fileinto "Spam";
   stop;
#Fuente: http://wiki.dovecot.org/LDA/Sieve#Example_scripts
}
```

"Be sure to manually pre-compile the scripts specified by sieve_before and sieve_after using the sievec tool, as explained below. For security reasons, global script directories are not supposed to be writable by the user. Therefore, the plugin cannot store the binary when the script is first compiled (lo compila en cada uso en memoria: ineficiente). Is necessary for scripts in sieve_global_path, sieve_before and sieve_after No parece fácil usar sieve-test, ver wiki".

```
#SIEVE_DIR="dir si tienen includes a otra ruta los scripts" \
sievec /afs/casafx.dyndns.org/service/mail/sieve/


ls /afs/casafx.dyndns.org/service/mail/sieve/
```

Quota, adicionalmente necesita

```
vim /usr/local/bin/quota-warning.sh
```

```
#!/bin/sh

PERCENT=$1
FROM="do.not.reply@‘echo $USER|awk -F@ ’{print $NF}’‘"
#FROM="postmaster@‘echo $USER|awk -F@ ’{print $NF}’‘"

cat << EOF | /usr/sbin/sendmail -f $FROM "$USER"
From: ${FROM}
To: ${USER}
Subject: Your email quota is ${PERCENT}% full
Content-Type: text/plain; charset=’UTF-8’


 This message is automatically created
 by mail delivery software.

 The size of your mailbox has exceeded
 a warning threshold that is
 set by the system administrator.
 You *must* delete mails or empty some folders
 or you may loose emails in the future.

EOF
exit 0
```

```
chmod +x /usr/local/bin/quota-warning.sh
```

Public mailboxes, funcionamiento mínimo

El maildirmake es importante: dovecot (proceso imap bajo afs id mailgroup) sólo puede leer y listar, no tiene permisos para escribir allí; así, hemos de precrear, desde una cuenta de administrador, las mailboxes maildir del namespace Public/:

```
kinit root/admin && aklog
maildirmake.dovecot /afs/casafx.dyndns.org/service/mail/Public/.newsletter
mutt -e 'set mbox_type=Maildir; set copy=yes'\
     -e 'set record=/afs/casafx.dyndns.org/service/mail/Public/.newsletter'\
     -e 'set sendmail="/bin/true"'\
     -e 'my_hdr From: newsletter team <noreply@casafx.dyndns.org>' -n \
     -s "Testing Newsletter system"\
     -i /usr/share/doc/base-files/README.FHS\
     -a /usr/share/doc/debian/FAQ/debian-faq.en.pdf.gz\
     -- newsletter@casafx.dyndns.org
tree -a /afs/casafx.dyndns.org/service/mail/Public/.newsletter
mutt -R -f /afs/casafx.dyndns.org/service/mail/Public/.newsletter

env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'aklog;
    ls -A /afs/casafx.dyndns.org/user/u/um/umea/.PublicMailboxIndexes'
```

Tests

Haremos algunas comprobaciones parciales de distitos subsistemas; en la sección sobre los clientes se hará una prueba completa.

Nota:

http://wiki.dovecot.org/PostLoginScripting da indicaciones para inspeccionar variables

durante el proceso etc

Fichero de configuración Comprobaremos la corrección sintáctica y cómo han quedado reflejados nuestros cambios:

```
su vmail -m -c 'head -n 1 /etc/dovecot/dovecot.conf'


dovecot -a   # check syntax, y muestra el estado de todas las opciones
dovecot -n   # muestra opciones no por defecto


invoke-rc.d dovecot start


ps auxw|egrep '(dovecot|imap|managesieve)'
```

Probando si deliver (dovecot MDA) puede acceder a afs Para ello necesita un token AFS para imap.dklab1:

```
env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'pagsh'
```

En esta nueva shell, conseguimos token afs para imap.dklab1:

```
aklog
```

y ahora mandamos dos correos lanzando manualmente a deliver:

```
cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
             -d aemu@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda

 Testing deliver (dovecot MDA) on AFS.
EOF

cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
             -d umea@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda

 Testing deliver (dovecot MDA) on AFS.
EOF

 # To avoid looping, add -c dovecot-nowarning.conf (the same as dovecot.conf minus
```

```
cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
             -d aemu@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda + sieve filtering.
X-Spam-Flag: Yes

 Testing deliver (dovecot MDA) on AFS and with an spam tagged mail
 in order to make global sieve script act, so that mail is stored
 in a different mailbox.
EOF

cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
             -d umea@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda + sieve filtering.
X-Spam-Flag: Yes

 Testing deliver (dovecot MDA) on AFS and with an spam tagged mail
 in order to make global sieve script act, so that mail is stored
 in a different mailbox.
EOF

exit
```

En syslog:

```
grep deliver /var/log/syslog
```

```
dovecot: deliver(aemu@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'INBOX'
dovecot: deliver(umea@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'INBOX'
dovecot: deliver(aemu@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'Spam'
dovecot: deliver(umea@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'Spam'
```

Comportamiento parecido tendrían los programas imap o managesieve.

SASL-GSSAPI en IMAP con dovecot-auth e imtest; diálogo IMAP Utilizaremos el cliente imap "imtest"del paquete cyrus-clients-2.2:

```
apt-get install cyrus-clients-2.2
login umea
```

```
klist
imtest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
       -f /dev/null dklab1.casafx.dyndns.org
klist
exit
```

Añadiendo -t "" realizará la autenticación sobre TLS.

Si cambiamos "-f /dev/null" por "-f /dev/stdin", podremos desarrollar un diálogo IMAP. Algunas características de nuestro despliegue necesitan ser aclaradas de esta manera, por ejemplo ahora podremos demostrar las conclusiones de nuestra discusión sobre el almacenamiento de las subscripciones y flags \Seen del espacio "Public/", o el seguimiento efectivo de las quotas. El rfc (http://tools.ietf.org/html/rfc3501) describe los comandos que se han de utilizar, una guía rápida de ejemplos puede encontrarse en http://bobpeers.com/technical/telnet_imap.

El formato de los comandos IMAP es, básicamente:

```
<etiqueta> <comando> <listaargumentos>
```

... donde la <etiqueta>que identificará nuestro diálogo será, por ejemplo, el símbolo ">" (es necesario escribir una etiqueta porque en IMAP se contemplan varios diálogos en una misma conexión, así que con la etiqueta se puede asociar la respuesta a la pregunta

que la lanzase).

Por su lado <comando>podría ser, para nuestros propósitos:

- LSUB (listar sólo subscripciones)
- LIST (listar todo el árbol de directorios y mailboxes)
- SUBSCRIBE (petición de subscripción a algún mailbox)
- UNSUBSCRIBE (petición desubscripción)
- SELEC (EXAMINE sería la versión read-only) selecciona el mailbox sobre el que se efectuaría comandos lectura/escritura como FETCH/STORE.
- FETCH permite con gran granularidad pedir partes del mensaje y sus metadatos.
- STORE permite cambios en los mensajes al modificar sus etiquetas.
- CLOSE al recibir un comando CLOSE, se deselecciona el SELECT anterior y se hacen efectivas las posibles eliminaciones de mensajes.
- GETQUOTAROOT (información de quota)
- LOGOUT para cerrar la comunicación a nivel de aplicación etc.

```
login umea
```

```
# Recordatorio: el si'mbolo ">" que aparece tras imtest no es su indicador
# de prompt sino parte de los comandos IMAP, luego es necesario escribirlo.
imtest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
       -f /dev/stdin dklab1.casafx.dyndns.org
> LSUB "" *
> LIST "" *
> SUBSCRIBE Spam
> LSUB "" *
> UNSUBSCRIBE Spam
> LSUB "" *
> SUBSCRIBE Spam
```

Nos hemos subscrito a Spam en el espacio de nombres raiz "/" y, por tanto su fichero de subscripciones debe localizarse en (utilícese, como root, otra consola distinta):

```
cat /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/subscriptions
```

```
...
Spam
```

```
> SUBSCRIBE Public/newsletter
```

De nuevo, si no existía se creará un fichero subscriptions bajo el directorio que representa ahora al espacio "Public/", y se añadirá el nombre del mailbox relativo a éste.

Si subscriptions = no (no es nuestro caso), se gestionaban en el espacio "/" y por tanto, como el anterior, en el fichero:

```
cat /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/subscriptions
```

```
...
Spam
Public/newsletter
```

Pero si subscriptions = yes, se gestionan aquí en "Public/" y por tanto en la ruta:

```
cat /afs/casafx.dyndns.org/user/a/ae/aemu/.PublicMailboxIndexes/subscriptions
```

```
newsletter
```

Nota: en este caso también se probó haciendo previamente read-only el 'subscriptions':

```
chmod a-w \
 /afs/casafx.dyndns.org/user/a/ae/aemu/.PublicMailboxIndexes/subscriptions
```

...entonces debería dar un error el comando subscribe puesto que imap perdería el permiso para escribir en el 'subscriptions'. Asombrosamente, contradiciendo el manual AFS, imap consigue escribir y adicionalmente volverle a cambiar el el bit 'w'posix a activado. No sabemos el por qué, pero a falta de más investigación, frusta la solución que se dio para fijar unas subscripciones que el usuario no pueda modificar (si no tiene permisos afs 'wa').

```
> STATUS INBOX (messages recent unseen)
```

El comando status nos permite pedir algo de información pero como sabemos es el comando FETCH tras un SELECT el que permite pedir información concreta de cada

mensaje. Además STORE permitirá modificar las etiquetas:

```
> SELECT INBOX
> FETCH 1:* flags
> FETCH 1 full
> FETCH 1 (rfc822.header rfc822.text)
> STORE 1 +flags mi_etiqueta_porque_no_empieza_por_backslash mietiqueta2
> FETCH 1 flags
> CLOSE
> SELECT Public/newsletter
> FETCH 1 (rfc822.header)
> FETCH 1 flags
```

En el caso del espacio de nombres "Public", queríamos que fuese de sólo lectura pero que los flags pudieran ser modificados, y además serlo por cada usuario (así que el flag \Seen reflejase realmente si el mensaje había sido visto por cada usuario). Veamos entonces que sucede así:

```
tree -as /afs/casafx.dyndns.org/user/a/ae/aemu/.PublicMailboxIndexes/
```

```
> STORE 1 +flags \Seen
```

El fichero dovecot.index.log debería incrementar su tamaño, indicando la transacción. (No vemos, sin embargo el dovecot.index: ésto puede ser estudiado con mayor profundiad en http://wiki.dovecot.org/IndexFiles, donde relaciona el uso de mmap_disable = yes y la versión 1.2 de dovecot con comportamientos especiales que parecen ser responsables).

```
tree -as /afs/casafx.dyndns.org/user/a/ae/aemu/.PublicMailboxIndexes/
```

Respecto a la eliminación de mensajes de un mailbox en "Public/", el flag aparentemente podrá establecerse, pero al enviar CLOSE, imap no podrá llevar a cabo la eliminación del correo porque no tiene permisos en esa parte del árbol AFS para el espacio "Public/", tal como era nuestra intención:

```
> STORE 1 flags \Deleted
> CLOSE
```

Syslog avisará de ello:

```
dovecot: IMAP(aemu@CASAFX.DYNDNS.ORG):
 utime(/afs/casafx.dyndns.org/service/mail/Public/.newsletter/tmp)
         failed: Permission denied
dovecot: IMAP(aemu@CASAFX.DYNDNS.ORG):
 unlink(/afs/casafx.dyndns.org/service/mail/Public/.newsletter/cur/\
 1320859903.14173_2.dklab1:2,)
         failed: Permission denied
```

Si la volvemos a seleccionar, es cierto que el flag continúa ahí, pero sin más consecuencia y podemos borrarlo o no.

```
> SELECT Public/newsletter
> FETCH 1 flags
> STORE 1 -flags \Deleted
> CLOSE
```

Por último, respecto a la quota "user quota", sabiendo que afectaba al espacio "/" y no a "Public/", debiéramos obtener los mismos bytes-usados y bytes-quota para cualquier mailbox afectada por la quoata "user quota", y nada para el mailbox del espacio "Public/".

```
> GETQUOTAROOT INBOX
  * QUOTA "user quota" (STORAGE 81 5000)

> GETQUOTAROOT Public/newsletter
  * QUOTAROOT "newsletter"

> LOGOUT
exit
```

SASL-GSSAPI en MANAGESIEVE con dovecot-auth y sivtest (bug) La herramienta sivtest también está incluída en cyrus-clients:

```
apt-get install cyrus-clients-2.2


login umea
```

```
klist


sivtest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
        -f /dev/null dklab1.casafx.dyndns.org
```

La salida, que inesperadamente termina en un error, es algo así:

```
C: AUTHENTICATE "GSSAPI" {836+}
   ...
S: ...


<queda bloqueado algunos minutos>


S: BYE "Disconnected for inactivity."
base64 decoding error -> supongo que porque no le ha llegado nada
Authentication failed. generic failure
Security strength factor: 0
Connection closed.
```

El error es de sivtest, de hecho el código de cyrus-clients ya ha tenido algún problema al usarse con dovecot-auth, véase:

http://bugs.debian.org/cgi-bin/bugreport.cgi?bug=604410

Añadiendo -t "" realiza la autenticación sobre TLS, pero el problema persiste. No obstante, la buena noticia es que el servicio funciona. Ésto tendremos oportunidad de comprobarlo más tarde al probar los clientes para el usuario (sieve-connect etc). En este momento, al menos, sí podemos verificar que fue pedido y concedido el ticket de servicio:

```
klist


exit
```

SASL-GSSAPI para SMTP-AUTH con dovecot-auth y smtptest Utilizamos a smtptest, un programa para probar SASL-GSSAPI en SMTP-AUTH que viene también con el paquete cyrus-clients:

```
apt-get install cyrus-clients-2.2
```

```
login umea
```

```
klist
smtptest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
         -f /dev/null dklab1.casafx.dyndns.org
```

```
...
S: 235 Authentication succeeded
Authenticated.
...
```

Añadiendo -t "" realizaría la autenticación SASL-GSSAPI pero sobre una capa TLS. Si todo ha ido bien, salimos de la shell de umea:

```
klist


exit
```

MDA opcional con Procmail

Queremos un mecanismo adicional para la entrega de correo, de forma que éste sea almacenado en el espacio home del usuario en AFS, aprovechando su carácter ubicuo. Este sistema opcional vamos a implementarlo con agentes MDA encargados típicamente de este estilo de entrega. Quizás procmail sea el más ampliamente utilizado (80 % popularidad en http://popcon.debian.org/ a pesar de dejar de desarrollarse en 2001, si bien debian va creando revisiones para corregir fallos de seguridad); no menos importante es que procmail soporta el formato Maildir en AFS. Una alternativa conocida pero sensiblemente menos pupular sería maildrop, si bien desconocemos su desempeño con AFS. http://qa.debian.org/popcon.php?package=procmail

La forma usual de usar procmail por parte del MTA, es pasándole el correo a través de stdin. Por su lado procmail le aplicará una serie de reglas de filtrado recogidas en /etc/procmail, ~/.procmailrc o cualquier otro a través del flag "-p". Cuando se llama a

procmail sin el flag "-p" y sin que pueda encontrar un ~/.procmailrc (caso de nuestra cuenta vmail, que no tiene home), usará /etc/procmailrc.

Como se ha reseñado, procmail es ampliamente desplegado y por ello la configuración por defecto de exim4 lo tiene en cuenta para la entrega de los usuarios locales, de forma que el router "procmail" de exim4 comprueba si está instalado y si existe configuración alguna, /etc/procmailrc o ~/procmailrc. Si se cumplen dichas condiciones, llama a procmail para que se haga cargo de la entrega.

No queremos que el uso de procmail para nuestra entrega de cuentas centralizadas interfiera con el uso para la entrega de cuentas locales que acabamos de mencionar. Así pues, teniendo en cuenta los dos párrafos anteriores una estrategia podría ser configurar las reglas para cuentas centralizadas en /etc/procmailrc y configurar las reglas para cuentas locales en cada ~/.procmailrc (y en este caso, exim4 usaría procmail sólo si el usuario local ha creado su propio ~/.procmailrc o heredó alguno desde /etc/skel etc). Para que este esquema funcione sólo nos falta eliminar el string-expansion que busca /etc/procmailrc en el router "procmail" para cuentas locales.

Nota: si omitimos este paso, la única diferencia será que el correo de un usuario local como "user" se encontrará por defecto en ~/Maildir en lugar de /var/mail/user, ya que se les aplicará la regla para cuentas centralizadas que declararemos en /etc/procmailrc y que hace básicamente éso.

Nos interesa pues el router "procmail", cuya directiva require_files todavía busca /etc/procmailrc:

```
view /etc/exim4/exim4.conf.template
```

```
...
procmail:
 debug_print = "R: procmail for $local_part@$domain"
 driver = accept
 domains = +local_domains
 check_local_user
 transport = procmail_pipe
 # emulate OR with "if exists"-expansion
 require_files = ${local_part}:\
                 ${if exists{/etc/procmailrc}\
                   {/etc/procmailrc}{${home}/.procmailrc}}:\
                 +/usr/bin/procmail
 no_verify
 no_expn
 ...
```

La directiva require_files debe ser transformada así:

```
vim /etc/exim4/exim4.conf.template
```

```
...
procmail:
...
 transport = procmail_pipe
####fx:
#-# emulate OR with "if exists"-expansion
#-require_files = ${local_part}:\
#-                ${if exists{/etc/procmailrc}\
#-                  {/etc/procmailrc}{${home}/.procmailrc}}:\
#-                +/usr/bin/procmail
 require_files = ${local_part}:${home}/.procmailrc:+/usr/bin/procmail
####endfx
 no_verify
 no_expn
 ...
```

```
invoke-rc.d exim4 stop; sleep 5; invoke-rc.d exim4 start
```

Ya podemos instalar procmail y editar /etc/procmailrc. Su contenido se crea desde cero, procuraremos que sea totalmente autoexplicativo:

```
apt-get install procmail
vim /etc/procmailrc
```

```
ls -l /etc/procmailrc # legible por cualquiera, tal cual vmail.
```

Veamos cómo a pesar de la presencia de /usr/bin/procmail y /etc/procmailrc, un correo a una cuenta local sin ~/.procmailrc, utiliza el router "local_user" y no el "procmail" gracias a la modificación que le hicimos a éste último:

```
exim4 -bt user@dklab1.casafx.dyndns.org
```

```
...
user@dklab1.casafx.dyndns.org
 router = local_user, transport = mail_spool # local_user, no procmail
```

Test Utilizaremos procmail y otra herramienta del proyecto (formail) para simular una entrega de correo. Previamente conseguiremos las credenciales necesarias (token AFS):

```
cp ~/mailtest/mail.txt /tmp
env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'pagsh'
```

En la nueva shell:

```
aklog
HOMEUMEA=`getent passwd umea | awk -F: '{print $6}'`
#/afs/casafx.dyndns.org/user/u/um/umea
cat /tmp/mail.txt | /usr/bin/formail -q- -s procmail \
                    -tm HOME=${HOMEUMEA} VERBOSE=on /etc/procmailrc
```

```
procmail: [18328] Sat Nov 12 02:27:12 2011
procmail: Rcfile: "/etc/procmailrc"
procmail: Assigning "MAILDIR=/afs/casafx.dyndns.org/user/u/um/umea"
procmail: Assigning "DEFAULT=/dev/null"
procmail: Match on ! "^X-Spam-Flag: [Yy][Ee][Ss]"
procmail: Assigning
 "LASTFOLDER=/afs/casafx.dyndns.org/user/u/um/umea/Maildir/new/
  1321061232.18328_0.dklab1"
From aemu@casafx.dyndns.org Sat Nov 12 02:27:11 2011
 Subject: hello
  Folder:
  /afs/casafx.dyndns.org/user/u/um/umea/Maildir/new/1321061232 141
```

```
cat /tmp/mail.txt | /usr/bin/formail -i "X-Spam-Flag: YES" -q- \
              -s procmail -tm HOME=${HOMEUMEA} VERBOSE=on /etc/procmailrc
```

...ahora el correo tiene el flag para spam, y va a /dev/null:

```
...
Folder: /dev/null
```

```
exit
```

1.3. DKLAB2

1.3.1. Instalación del software principal (Exim4, Dovecot)

Instalábamos inicialmente la parte principal del software:

Exim4

```
apt-get install exim4-daemon-heavy pfqueue spfqtool libmail-spf-perl
dpkg-reconfigure -plow exim4-config
```

```
 -"General type of mail configuration:"
internet site; mail is sent and received directly using SMTP
 -"System mail name:"
dklab2.casafx.dyndns.org
 -"IP-addresses to listen on for incoming SMTP connections:"
0.0.0.0;0.0.0.0.587;::1;::1.587
 -"Other destinations for which mail is accepted:"
casafx.dyndns.org
 -"Domains to relay mail for:"
<ninguno>
 -"Machines to relay mail for:"
<ninguna ip>
 -"Keep number of DNS-queries minimal (Dial-on-Demand)?"
No.
 -"Delivery method for local mail:"
mbox format in /var/mail/
 -"Split configuration into small files?"
No.
```

```
invoke-rc.d exim4 stop


mkdir ~/mailtest
cp /etc/init.d/exim4 ~/mailtest/etc-init.d-exim4
```

Dovecot

```
apt-get install dovecot-common dovecot-imapd mutt sieve-connect
invoke-rc.d dovecot stop; pkill ssl-build-param
```

```
...
Creating generic self-signed certificate: /etc/ssl/certs/dovecot.pem
...
```

```
invoke-rc.d dovecot stop
pkill ssl-build-param
```

Nota: tras la instalación, la ausencia de los parámetros DH provoca que se llame a ssl-build-param para crearlos; si nuestra máquina/emulador es lenta, la creación de los parámetros DH puede tardar mucho tiempo, acaparando intensivamente el tiempo del procesador; si ésto nos resulta molesto en este momento, podemos crear unos parámetros inseguros ahora para evitar la llamada a ssl-build-param, y lidiar con ello más tarde (además y como se verá, en el archivo de configuración se puede indicar cada cuánto renovar esos parámetros DH).

```
cat <<EOF | base64 -d > /var/lib/dovecot/ssl-parameters.dat
AAIAAEgAAAAwRgJBAK6j4/F9kt0TMUBNTMMdj7FnoFwkpla7zda7iUWX6KqxQ1dMfu1zJshTREDz
aJgWr5xSkXxxvRQL5Z3JvRg5QTMCAQIABAAAigAAADCBhwKBgQDQCaM146ukwsw4ZiRNO1R3hzcT
vGoGns1C5fiBGWr9hfl8cYfsj9cExER1Ev+p7aNbk/YBMjwU/HCGH3Q5MZJ0aUtf0ONUos9hKymz
khZWHmTeshfTxw1CxXSPhP2by0Eiz2AnDJ6Zqc0Unz8rT9/Vja7sIpFoompMCGcxdUOIqwIBAgAA
AAA=
EOF
```

1.3.2. Soporte TLS

TLS era fundamental para proveer confidencialidad en todas las comunicaciones (SMTP, IMAP, MANAGESIEVE).

Transferiremos por un canal seguro los pares creados en dklab1. En el caso de exim4, su certificado puede ser (gracias al atributo SAN del modelo PMI), compartido con el de dklab1, y puede ser usado para DKIM.

```
scp dklab1:/usr/local/lib/easy-rsa/2.0/keys/exim4.casafx.dyndns.org.crt \
    dklab1:/usr/local/lib/easy-rsa/2.0/keys/exim4.casafx.dyndns.org.key \
    /etc/exim4/
scp dklab1:/usr/local/lib/easy-rsa/2.0/keys/dovecot-2.casafx.dyndns.org.crt \
    dklab1:/usr/local/lib/easy-rsa/2.0/keys/dovecot-2.casafx.dyndns.org.key \
    /etc/dovecot/

openssl x509    -text -in /etc/exim4/exim4.casafx.dyndns.org.crt \
        | grep -A1 'Subject Alt'

cd /etc/exim4
chown root:Debian-exim exim4.casafx.dyndns.org.key exim4.casafx.dyndns.org.crt
chmod 640 exim4.casafx.dyndns.org.key exim4.casafx.dyndns.org.crt
#
cd /etc/dovecot
chown root:root dovecot-2.casafx.dyndns.org.key dovecot-2.casafx.dyndns.org.crt
chmod 600 dovecot-2.casafx.dyndns.org.key dovecot-2.casafx.dyndns.org.crt
#

cd ~
```

Tras configurar TLS en exim4, podemos dejar que se creen los parámetros DH a la primera conexión (modo de operar de gnutls, con quien se enlaza exim4, pero que puede ser muy costoso en tiempo), o podemos precomputar una caché /var/spool/exim4/gnutls-params usando:

/usr/share/exim4/exim4_refresh_gnutls-params

El resultado en formato PEM sería algo como:

```
openssl dh -text -in /var/spool/exim4/gnutls-params | ccze -A | less -R
cat /var/spool/exim4/gnutls-params
```

```
-----BEGIN DH PARAMETERS-----
MIIBCAKCAQEArYVHPGLKJ8ugLSZhvJXKI7L/AJI+8RvtKkCt9WCntDIwG8ZdUQFP
YNZC5dk4yISA3aw34DS6ihFIhQUm/eLt6/mO34/gNdoWTqRdk1K2+TpwEcHSZ8ir
ynyqqiQGWxoHe9YJyLzeAY+FlYh55JEnx2yiPRQxQCnQYR7e5PxCm0SDqkCVcHwh
VELpKbgsdlM43NZ/RXesBR/o95H21TMbz42LzaV7tqoT9r/iDlrXCHM3XeY4HjbZ
cRp13LaCqORMOsEccVugTE/1RFUIla/CnDSuwKbIBDksEMvbusI61wre5yrd1tIY
3X5z7sMtkaAKfhBqc65svjVRS8FbOy12SwIBAg==
-----END DH PARAMETERS-----
```

En el caso de dovecot, sus parámetros DH se crean por /usr/lib/dovecot/ssl-build-param (efectivamente el script que se aconsejó parar tras instalar dovecot), concretamente cuando no existe el fichero /var/lib/dovecot/ssl-parameters.dat o cada lo que diga la variable ssl_parameters_regenerate en /etc/dovecot/dovecot.conf. Un valor de 0 a esa variable significa deshabilitar su renovación.

1.3.3. Soporte para DKIM en DNS (ya hecho)

Si todo fue bien en dklab1 al incorporar la llave pública en DNS, debería haber sido transferida automáticamente al resto de servidores dns, y no debería ser necesario reiniciar el servicio o incorporarlo nuevamente aquí.

```
dig @dklab2.casafx.dyndns.org exim4dkim._domainkey.casafx.dyndns.org. TXT +short
```

1.3.4. Soporte para centralización de metadatos: LDAP (ya hecho)

Esquema qmail-ldap en database cn=config de dklab1 y dklab2

Como se expuso en dklab1, las bases de datos cn=config no se replican entre dklab1 y dklab2 pero sí dc=casafx,dc=dyndns,dc=org, por lo que la carga del esquema qmail-ldap ya se llevó a cabo inmediatamente después de hacerlo en dklab1, antes de usarlo en dc=casafx,dc=dyndns,dc=org.

```
slapcat -b cn=config | grep "qmail,"
```

Uso del esqumema qmail-ldap

Gracias al mecanismo de replicación, los atributos que se cargaron desde dklab1 en ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org lo están también en dklab2.

```
ldapsearch -x -LLL -H ldap://dklab2.casafx.dyndns.org \
           -b ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org -s sub \
           mailAlternateAddress
```

1.3.5. Soporte para autenticación KERBEROS y almacenamiento en AFS

El material criptográfico ya fue creado, pero en lo que respecta a dklab2, aún tenemos que exportarlo. Eraimportante el orden, concretamente queremos que el primer principal exportado sea el de imap.

```
kadmin.local
>addprinc -policy service -randkey smtp/dklab2.casafx.dyndns.org
>addprinc -policy service -randkey imap/dklab2.casafx.dyndns.org
>addprinc -policy service -randkey sieve/dklab2.casafx.dyndns.org
>ktadd -k /etc/keytab.d/dovecot.keytab \
       -norandkey imap/dklab2.casafx.dyndns.org
>ktadd -k /etc/keytab.d/dovecot.keytab \
       -norandkey sieve/dklab2.casafx.dyndns.org
>ktadd -k /etc/keytab.d/dovecot.keytab \
       -norandkey smtp/dklab2.casafx.dyndns.org
>quit
```

Uso de los principal: servicios kerberizados

Al utilizar dovecot-auth también para SMTP-AUTH, todos los principal iban al dovecot.keytab que utiliza éste y, al ejecutarse como root, requería estos permisos:

```
klist -k /etc/keytab.d/dovecot.keytab


chown root:root /etc/keytab.d/dovecot.keytab
chmod go-rw /etc/keytab.d/dovecot.keytab
su root -m -c 'klist -t -K -k /etc/keytab.d/dovecot.keytab'
```

Uso de los principal: cliente kerberos

Éste era, recordamos, el esquema que seguir para el rol de cliente kerberos y, así, AFS:

- el script /etc/init.d/dovecot creará con ayuda de k5start una PAG usando el principal imap/dklab2.casafx.dyndns.org almacenado en el dovecot.keytab, por tanto el token AFS otorga la identidad AFS "imap.dklab2" a los procesos imap y managesieve que lanza.
- el script /etc/init.d/exim4 creará con ayuda de k5start una PAG usando el principal imap/dklab2.casafx.dyndns.org almacenado en dovecot.keytab, por tanto el token AFS otorga la identidad AFS "imap.dklab2" a los procesos deliver y procmail que lanza.
- las mailboxes en AFS tenían acl permisivas para el grupo AFS "mailgroup", vamos en dklab2 a crear la identidad afs "imap.dklab2" y a incluirla en ese grupo, de forma que al grupo AFS "mailgroup" pertenezcan "imap.dklab1" e "imap.dklab2".

Se aplican las mismas anotaciones que se comentaron para dklab1:

- Ninguna de estas decisiones para el rol de cliente kerberos y AFS afecta a las que se tomaron para desarrollar el rol de servicio kerberizado.
- K5start renovará puntualmente, como siempre, todas las credenciales.
- Puesto que el principal imap/dklab2.casafx.dyndns.org es el primero exportado en dovecot.keytab, podemos pasar -U, en lugar de -u <principal>, a k5start. (Compurébese con

  ```
  klist -k /etc/keytab.d/dovecot.keytab
  ```

).
- KRB5CCNAME=MEMORY: no será usado: man k5start recomienda indicarlo usando esa variable (y omitiendo -k), pero no parece tener efecto y siguen guardándose como un fichero bajo "/tmp", desconocemos por qué. Mientras no se investigue ésto, optamos por mantener dos tickets TGT (sin uso alguno tras conseguir el token AFS) en /tmp pasando las opciones

`-k <file> -o <uid>`

de k5start. Puede comprobarse con:

-

```
env KRB5CCNAME=MEMORY: /usr/bin/k5start -t -U -f /etc/keytab.d/dovecot.keytab\
        -K 10 -l 24h -- cat /proc/self/environ \
        | xargs --null --max-args=1 echo|grep KRB
#...devuelve KRB5CCNAME=/tmp/krb5cc_0_9xwRPj
env KRB5CCNAME=MEMORY: /usr/bin/k5start -t -U -f /etc/keytab.d/dovecot.keytab\
    -K 10 -l 24h -- klist | grep Ticket
#...devuelve Ticket cache: FILE:/tmp/krb5cc_0_HKbC3m
#http://newsgroups.derkeiler.com/Archive/Comp/comp.protocols.kerberos/\
#                                                2010-10/msg00063.html
```

Ejecución de las modificaciones recién aclaradas

K5start en los scripts de inicio; el resultados será algo críptico porque tenemos que hacer un trabajo de integración con las funciones "start" preexistentes en tales scripts.

```
vim /etc/init.d/dovecot
```

```
...
do_start()
{
####fx:
    local DAEMON='/usr/bin/k5start'
    local CONF="${PIDFILE} -b -t -U -f /etc/keytab.d/dovecot.keytab -K 10 -l 24h
                -k /tmp/.krb5cc_`id -u vmail`_dovecot -o vmail
                -- dovecot -F -c ${CONF}"
#${PIDFILE} (en verdad -c ${PIDFILE} porque le precedera' -c al ser
# evaluada la variable CONF) hace que escriba el pid de dovecot en ese
# fichero.
#-b hace que k5start se ejecute en background
#-t es la responsable de que se pida token afs en una nueva PAG, que
#   heredara' dovecot y que sera' distinta a la PAG y token de exim4.
#-o vmail hara' que el TGT kerberos sea accesible so'lo para el uid vmail.
#-k ${KRB5CCNAME} implica que no puede sobreescribir los tickets
#   pedidos por el k5start de exim4, pues la cache' tiene distinto
#   nombre.
#NOTA: KRB5CCNAME=MEMORY: no dio resultado, de ahi' el uso
#      de -k <file> -o <uid>
#AFS es necesario para dovecot:
#    -al lanzar imap, con uid=vmail.
#    -al lanzar managesieve, cou uid=vmail.
####endfx
...
```

```
vim /etc/init.d/exim4
```

```
...
start_exim()
{
####fx:
local K5START__="/usr/bin/k5start -b -c ${PIDFILE}
                 -t -U -f /etc/keytab.d/dovecot.keytab -K 10 -l 24h
                 -k /tmp/.krb5cc_‘id -u vmail‘_exim4 -o vmail
                 -- "
      start_daemon -p  "$PIDFILE"  ${K5START__} \
        "$DAEMON" -bdf "-q${QFLAGS}${QUEUEINTERVAL}" \
        ${COMMONOPTIONS} \
        ${QUEUERUNNEROPTIONS} \
        ${SMTPLISTENEROPTIONS}
#-    start_daemon -p "$PIDFILE" \
#-      "$DAEMON" -bd "-q${QFLAGS}${QUEUEINTERVAL}" \
#-      ${COMMONOPTIONS} \
#-      ${QUEUERUNNEROPTIONS} \
#-      ${SMTPLISTENEROPTIONS}
#NOTA: KRB5CCNAME=MEMORY: no dio resultado, de ahi' el uso de
#      -k <file> -o <uid>
#AFS es necesario para exim4:
#    -en la entrega, al lanzar a deliver, con uid=vmail
#    -en la entrega opcional con procmail, cou uid=vmail
####endfx
...
```

Cuenta posix con que lanzar los procesos hijos (deliver, procmail; imap, managesieve):

```
getent group 800   # No debiera existir usuario o grupo
getent passwd 800  # con el id que hemos elegido (800)
addgroup --system --gid 800 vmail
adduser --system --no-create-home --disabled-login --uid 800 --gid 800 vmail
#--system hace que no tenga shell (/bin/false).
#adduser vmail mail no es necesario (ni siquiera para dovecot), aunque posible.
```

```
  Adding system user ‘vmail’ (UID 800) ...
  Adding new user ‘vmail’ (UID 800) with group ‘vmail’ ...
  Not creating home directory ‘/home/vmail’.
```

Creación de la identidad AFS "imap.dklab2". Previamente usamos el principal de root/admin:

```
kinit root/admin; aklog
```

Y ya, sabiendo que el nombre válido y mapeable desde el principal kerberos, era imap.dklab2:

```
pts listentries -users
pts createuser imap.dklab2 -id 30001 # imap/dklab2.casafx.dyndns.org
pts listentries -users


pts listentries -groups
pts adduser imap.dklab2 -group mailgroup #imap/dklab2.casafx.dyndns.org:mailgroup
pts membership mailgroup                 # users que esta’n en ese grupo?
pts membership imap.dklab2               # grupos en que esta’ ese user?
```

1.3.6. Soporte antivirus con Clamav

```
apt-get install clamav-daemon clamav-freshclam
```

```
 -"Virus database update method:"
daemon
 -"Local database mirror site:"
db.local.clamav.net (u otro apropiado con criterio geogra'fico).
 -"HTTP proxy information (leave blank for none):"
<nada>
 -"Number of freshclam updates per day:"
24 (si bien durante las pruebas puede ser menos molesto un valor como 1)
 -"Should clamd be notified after updates?"
Yes
```

El socket unix de clamd es ya legible/escribible por cualquiera:

```
ls -l /var/run/clamav/clamd.ctl
```

```
srw-rw-rw- 1 clamav clamav 0 2011-10-28 12:32 /var/run/clamav/clamd.ctl
```

```
ls -ld /var/spool/exim4/scan
```

```
drwxr-x--- 2 Debian-exim Debian-exim
```

En ese directorio debía acceder rw clamd, el cual se ejecuta como la entidad clamav; la solución era añadir clamav al grupo Debian-exim, permitir ese esquema en la configuración de clamav (así viene por defecto) y dar permisos "w" de escritura para el grupo a ese directorio.

```
adduser clamav Debian-exim
grep ^AllowSupplementaryGroups /etc/clamav/clamd.conf
```

```
  AllowSupplementaryGroups true
```

```
mkdir -p /var/spool/exim4/scan; chmod g+w /var/spool/exim4/scan


invoke-rc.d clamav-daemon restart
```

Test

```
echo 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*' \
 | su Debian-exim -m -c 'clamdscan -v -'
```

```
stream: Eicar-Test-Signature FOUND
----------- SCAN SUMMARY -----------
 Infected files: 1
```

1.3.7. Soporte antispam con Spamassassin

```
apt-get install spamassassin libnet-ldap-perl spamc
vim /etc/spamassassin/local.cf
```

```
...
####fx:
include /etc/spamassassin/local-fx-01.cf
####endfx
```

```
vim /etc/spamassassin/local-fx-01.cf
```

(La sentencia definiendo user_scores aparece dividida en 4 partes delimitadas por \ y cambio de línea. Realmente debe aparecer en una sóla línea, sin \ y cambio de línea adicionales:)

```
 ####fx:
 user_scores_dsn                 ldap://dklab2.casafx.dyndns.org/\
ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?\
spamassassinUserPrefs?sub?\
(&(uid=__USERNAME__)(objectClass=qmailUser))
#...en una so'la li'nea, recordamos.

 #Desafortunadamente, no parece que spamd pueda usar registros SRV para
 #descubrir los servidores LDAP, ni permite declarar varias
 #variables user_scores_dsn ni otras manipulaciones. Asi', esta instancia de
 #spamd usa al servidor dklab2.casafx.dyndns.org (y a la db
 #dc=casafx,dc=dyndns,dc=org).
 #Consu'ltese /usr/share/doc/spamassassin/README.ldap.gz
 #http://cpansearch.perl.org/src/FELICITY/Mail-SpamAssassin-3.0.0/ldap/README

 clear_headers
 add_header all Flag _YESNO_
 add_header spam Result _SCORE_/_REQD_ (_TESTS_)
 #
 # Nota: Por defecto, escucha en ip 127.0.0.1 tcp 783.
 ####endfx
```

```
vim /etc/default/spamassassin
```

```
...
# Change to one to enable spamd
####fx:
#-ENABLED=0
ENABLED=1
####endfx
...
####fx:
#-OPTIONS="--create-prefs --max-children 5 --helper-home-dir"
OPTIONS="-x --ldap-config -u nobody --max-children 5"
#-x             disable user-config files
#--ldap-config  enable ldap-config
#-u nobody      la'nzate como nobody en lugar de como root, pues no necesitas
#               acceder a preferencias en el home del usuario.
####endfx
```

Test

Para comprobar que funcionaba, utilizábamos un correo ejemplo de spam:

```
spamassassin -t < /usr/share/doc/spamassassin/examples/sample-spam.txt | less
```

```
...
X-Spam-Checker-Version: SpamAssassin 3.3.1 (2010-03-16) on
        dklab1.casafx.dyndns.org
X-Spam-Flag: Yes
X-Spam-Result: 1000.0/5.0 (GTUBE,NO_RECEIVED,NO_RELAYS)
```

...ha detectado el correo ejemplo de spam; ha incluído la cabecera X-Spam-Flag: Yes
Consulta de preferencias de usuario desde ldap:

```
spamd -D -x --ldap-config -u nobody --max-children 5 2>&1 | grep ldap
```

...desde otra consola:

```
cat /usr/share/doc/spamassassin/examples/sample-spam.txt | spamc -u umea
```

```
...
dbg: ldap: URL is ldap://dklab2.casafx.dyndns.org/\
  ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?\
  spamassassinUserPrefs?sub?(&(uid=__USERNAME__)(objectClass=qmailUser))
dbg: ldap: host=dklab2.casafx.dyndns.org, port=389,\
  base='ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',\
  attr=spamassassinUserPrefs, scope=sub,\
  filter='(&(uid=umea)(objectClass=qmailUser))'
dbg: ldap: retrieving prefs for umea: required_score 5.00
dbg: ldap: retrieving prefs for umea: use_bayes 1
dbg: info: user has changed
```

... si todo va bien, con CTRL-C podemos cerrar la instancia spamd de prueba.

```
invoke-rc.d spamassassin start
```

Más sobre spamassassin

Consúltese las notas que se dieron al desplegar dklab1, en general un resumen de /usr/share/doc/spamassassin/README.Debian.gz, README.ldap.gz.

1.3.8. MTA

Sistema de configuración de exim4

Recordamos el modo de configurar exim4 tal como se expuso al desplegar en dklab1:
"invoke-rc.d exim4 start"construía el fichero /var/lib/exim4/config.autogenerated definitivo a partir del /etc/exim4/exim4.conf.template donde se realizaban las modificaciones principalmente. Así pues editábamos /etc/exim4/exim4.conf.template para configurar exim4. Adicionalmente /etc/default/exim4 (editable directamente o con update-exim4defaults) y /etc/exim4/update-exim4.conf.conf (editable directamente o con "dpkg-reconfigure -plow exim4") controlaban algunas variables particulares. Finalmente, ordenábamos que se apliquen los cambios reiniciando exim4 con: "invoke-rc.d exim4 stop && invoke-rc.d exim4 start".

Algunos ejemplos de string-expansion con lookups ldap/ldapm

```
cat <<EOF > ~/mailtest/mail.txt
From: aemu@casafx.dyndns.org
To: umea@casafx.dyndns.org
Cc:
Bcc:
Subject: hello
Reply-To:

This is the body of an rfc2822 message.
EOF
```

Volvemos a mencionar las precauciones que se dieron:

- exim4 "cachea" los resultados de las queries a cierto nivel (per message, suponemos)
- no úsense espacios tras filtro ldap y el símbolo "}", es decir, no úsese algo como "<filtro>) }}".
- Tras un escape de línea "\", el espacio en blanco hasta el primer caracter de la siguiente línea no es usado en absoluto, permitiéndonos crear márgenes de sangría para clarificar las expansiones.

Ejemplo con variable reelevante:

```
exim4 -bP +local_domains
```

Ejemplos útiles en /etc/exim4/local-main-00-gnl (véase luego):

```
exim4 -be 'ou=users,ou=accounts,dc=${sg{casafx.dyndns.org}{\\.}{,dc=}}'
exim4 -be '${if exists{/etc/ssl/certs/casafx-ca.crt}
            {/etc/ssl/certs/casafx-ca.crt}{/dev/null}}'
```

Ejemplos útiles en /etc/exim4/local-main-01-gnl (véase luego):

```
BE='-t -bem /root/mailtest/mail.txt -bfl umea -bfd casafx.dyndns.org'
exim4 $BE '${lookup ldapm{dereference=never time=15
     ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?uid?sub?}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
homeDirectory?sub?(uid=umea)}}'

exim4 $BE '${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
uidNumber?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
mailQuotaSize?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
mailQuotaCount?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
mailSizeMax?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
deliveryMode?sub?(uid=aemu)}}'
```

```
exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
mailReplyText?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
uid?sub?(mailAlternateAddress=postmaster@casafx.dyndns.org)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
mailForwardingAddress?sub?(uid=umea)}}'

exim4 $BE '${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
deliveryProgramPath?sub?(uid=umea)}}'

exim4 $BE '${lookup
ldap{ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
homeDirectory?sub?(uid=umea)}}'
```

Ejemplos útiles en /etc/exim4/local-router-250-lda:

- En router ldap_maxsize:

```
exim4 $BE '${lookup ldap{dereference=never time=15 \
ldap:///ou=users,ou=accounts,dc=${sg{casafx.dyndns.org}{\\.}{,dc=}}?\
mailSizeMax?sub?(uid=umea)}}'
```

- En router ldap_mailalternate:

```
exim4 $BE '${map{<\n${lookup ldapm{dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
uid?sub?(mailAlternateAddress=postmaster@casafx.dyndns.org)}}}
{${item}@casafx.dyndns.org}}'
```

- En router ldap_dovecot_mda y otras:

```
exim4 $BE '${lookup ldap{ dereference=never time=15 ldap:///ou=users,ou=accounts,d

exim4 $BE '${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
uid?sub?(&(uid=umea)(uid=umea))}}'

#... que unidas:
exim4 $BE '${if and{{!eq{nolocal}{${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
deliveryMode?sub?(&(uid=umea)(deliveryMode=nolocal))}}}}{eq{umea}
{${lookup ldap{ dereference=never time=15
ldap:///ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org?
uid?sub?(&(uid=umea)(uid=umea))}}}}}{yes}{no}}'
```

/etc/exim4/exim4.conf.template

```
vim /etc/exim4/exim4.conf.template
```

```
...
#                    MAIN CONFIGURATION SETTINGS                        #
#########################################################################
####fx:
.include /etc/exim4/local-main-00-gnl
####endfx
...
# Deny if the message contains malware. Before enabling this check, you
# must install a virus scanner and set the av_scanner option in the
# main configuration.
####fx:
.include /etc/exim4/local-acl-config-check-data-00-clamav
####endfx
...
### end acl/40_exim4-config_check_data
#########################################################
####fx:
.include /etc/exim4/local-acl-check-dkim
####endfx
...
### end router/200_exim4-config_primary
#########################################################
####fx:
.include /etc/exim4/local-router-210-spamassasing
.include /etc/exim4/local-router-250-ldap
####endfx
```

```
...
### end transport/35_exim4-config_address_directory
#########################################################
####fx:
.include /etc/exim4/local-transports-40-ldapusers
.include /etc/exim4/local-transports-50-spamassasing
####endfx
...
### end retry/00_exim4-config_header
#########################################################
####fx:
.include /etc/exim4/local-retry-20-ldap
####endfx
...
### end auth/30_exim4-config_examples
#########################################################
####fx:
.include /etc/exim4/local-auth-40-ldapusers-dovecotsasl
####endfx
```

/etc/exim4/local-main-00-gnl

```
vim /etc/exim4/local-main-00-gnl
```

```
####fx:
MAIN_LDAP_ENABLE = true
#No consulta SRV RR, luego hay que definir expli'citamente los servidores ldap:
MAIN_LDAP_DEFAULT_SERVERS = dklab2.casafx.dyndns.org::389:dklab1.casafx.dyndns.org
MAIN_LDAP_VERSION = 3
MAIN_LDAP_BASEDN = ou=users,ou=accounts,dc=${sg{$domain}{\\.}{,dc=}}
MAIN_LDAP_TIMEOUT = 15
#MAIN_LDAP_BINDDN =
#MAIN_LDAP_BINDPW =

.include /etc/exim4/local-main-01-ldap
#
MAIN_TLS_ENABLE = true
MAIN_TLS_CERTIFICATE = /etc/exim4/exim4.casafx.dyndns.org.crt
MAIN_TLS_PRIVATEKEY  = /etc/exim4/exim4.casafx.dyndns.org.key
MAIN_TLS_VERIFY_CERTIFICATES = ${if \
                                exists{/etc/ssl/certs/casafx-ca.crt}\
                                {/etc/ssl/certs/casafx-ca.crt}\
                                {/dev/null}}
#
av_scanner = clamd:/var/run/clamav/clamd.ctl
#...el soporte para el filtro antivirus necesitara' tambie'n modificar
#   "acl_check_data:"
.ifndef CHECK_RCPT_IP_DNSBLS
CHECK_RCPT_IP_DNSBLS = cbl.abuseat.org:dnsbl.njabl.org:sbl.spamhaus.org
.endif
```

```
#...esta' a warn, se puede cambiar a deny, bu'squese entonces
#         .ifdef CHECK_RCPT_IP_DNSBLS en el exim4.conf.template
.ifndef CHECK_RCPT_SPF
CHECK_RCPT_SPF = true
.endif
#
#auth_advertise_hosts = * por defecto, y se aplica si has definido
#                         al menos un driver para auth.
#
# Para usar DKIM y firmar correo saliente:
DKIM_DOMAIN = ${lc:${domain:$h_from:}}
DKIM_FILE = /etc/exim4/exim4.casafx.dyndns.org.key
DKIM_PRIVATE_KEY = ${if exists{DKIM_FILE}{DKIM_FILE}{0}}
DKIM_SELECTOR = exim4dkim
DKIM_CANON = relaxed
#... todas son, si definidas, usadas en el transport "remote_smtp"
#    definido en /etc/exim4/exim4.conf.template.
# Para utilizar DKIM verificando firmas en correo entrante:
acl_smtp_dkim = acl_check_dkim
# ...definiremos la acl de nombre "acl_check_dkim" ma's tarde.
####endfx
```

/etc/exim4/local-main-01-ldap

```
vim /etc/exim4/local-main-01-ldap
```

```
####fx:
# Este fichero es inclui'do desde /etc/exim4/local-main-00-gnl
# y no desde el /etc/exim4/exim4.conf.template.
.ifdef MAIN_LDAP_ENABLE

.ifndef MAIN_LDAP_DEFAULT_SERVERS
MAIN_LDAP_DEFAULT_SERVERS = dklab1.casafx.dyndns.org::389:\
                            dklab2.casafx.dyndns.org::389
.endif
ldap_default_servers = MAIN_LDAP_DEFAULT_SERVERS

.ifndef MAIN_LDAP_VERSION
MAIN_LDAP_VERSION = 3
.endif
ldap_version = MAIN_LDAP_VERSION

.ifndef MAIN_LDAP_BASEDN

MAIN_LDAP_BASEDN = ou=users,ou=accounts,dc=${sg{${lc:$domain}}{\\.}{,dc=}}
.endif
LDAP_BASEDN = MAIN_LDAP_BASEDN
```

```
#.ifndef MAIN_LDAP_BINDDN
#MAIN_LDAP_BINDDN = cn=admin,dc=example,dc=com
#MAIN_LDAP_BINDPW = CHANGE
#.endif
#LDAP_BINDDN = MAIN_LDAP_BINDDN
#LDAP_BINDPW = MAIN_LDAP_BINDPW

.ifndef MAIN_LDAP_TIMEOUT
MAIN_LDAP_TIMEOUT = 15
.endif
LDAP_TIMEOUT = MAIN_LDAP_TIMEOUT

# Determines how aliases are handled during a search. This option is available
# only with OpenLDAP 2.0.
# NOTE: DEREF can be one of the following values: never, searching, finding,
#       always. If not specified, aliases are never dereferenced.
#
.ifndef MAIN_LDAP_DEREF
MAIN_LDAP_DEREF = never
.endif
LDAP_DEREF = MAIN_LDAP_DEREF
```

```
LDAP_UID = uidNumber
# The LDAP_HOMEDIR attribute MUST exist, and we MUST be able to chdir
# to it. It is used as $home during transport.
#
LDAP_HOMEDIR = homeDirectory
# If homeDirectory is not an absolute path, define the root of the
# relative paths in LDAP_MAILROOT.
# LDAP_MAILROOT = /
# The LDAP_MAILDIR attribute is OPTIONAL, and specifies the location of
# target maildir. If specified as relative path, $home is automatically
# prepended to it. If not specified, $home/Maildir will be used.
# LDAP_MAILDIR = Maildir

LDAP_QUOTA_SIZE = mailQuotaSize
LDAP_QUOTA_COUNT = mailQuotaCount

LDAP_MAXSIZE = mailSizeMax
# Multi field entries of these keywords:
# - (no entry): Default delivery, put message into maildir (localdelivery),
#   plus forward and program delivery if specified.
# - noforward: Do not forward (ignores forwarding entry).
# - nolocal: Do not put message into maildir.
# - noprogram: Do no do program deliveries.
# - reply: Send a vacation message with text from LDAP_REPLYTEXT
LDAP_MODE = deliveryMode

LDAP_REPLYTEXT = mailReplyText
```

```
LDAP_MAILALTERNATE = mailAlternateAddress
LDAP_FORWARDS = mailForwardingAddress
LDAP_PROGRAM = deliveryProgramPath
LDAP_MAIL = uid

LDAP_DEFAULT_CONTROLS = dereference=LDAP_DEREF time=LDAP_TIMEOUT

LDAP_DEFAULT_FILTER = (LDAP_MAIL=${local_part})

LDAP_DEFAULT_MAXSIZE = ${lookup ldap{\
  LDAP_DEFAULT_CONTROLS \
  ldap:///LDAP_BASEDN?LDAP_MAXSIZE?sub?LDAP_DEFAULT_FILTER\
}}

# This macro is a short-hand for LDAP_HOMEDIR query.
LDAP_DEFAULT_HOMEDIR = ${lookup ldap{\
  LDAP_DEFAULT_CONTROLS \
  ldap:///LDAP_BASEDN?LDAP_HOMEDIR?sub?LDAP_DEFAULT_FILTER\
}}
.endif
####endfx
```

/etc/exim4/local-router-250-ldap

(Éstos son los routers que diseñamos nosotros. La justificación se dio al desplegar en dklab1).

```
vim /etc/exim4/local-router-250-ldap
```

```
####fx:
.ifdef MAIN_LDAP_ENABLE
# - Si el correo es demasiado grande, debie'ramos avisar de ello y no
# almacenarlo: usamos "driver = redirect"; "data" contiene :fail: ;
# "condition" es el peso del correo y el aviso esta' tambie'n en "data";
# no usamos "unseen" para que no se desdoble y continu'e.
```

```
ldap_maxsize:
  debug_print = "R: ldap_maxsize for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  condition = ${if \
                  and{\
                      {>{LDAP_DEFAULT_MAXSIZE}{0}}\
                      {>{$message_size}{LDAP_DEFAULT_MAXSIZE}}\
                    \}
                  {yes}{no}\
                }
  data = :fail:\n\Your message is too big.\n\
    Your message was rejected because the user ${local_part}@${domain}\n\
    does not accept message larger than LDAP_DEFAULT_MAXSIZE.
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
```

```
# - Si el correo es para alguien que esta' de vacaciones, debemos avisar
# adema's de almacenarlo: usamos driver accept; "condition" es que el
# deliveryMode en ldap sea "reply"; "unseen" para que siga y se almacene.
ldap_vacation:
  debug_print = "R: ldap_replytext for $local_part@$domain"
  driver = accept
  no_verify
  no_expn
  unseen
  condition = ${if \
                  and{\
                      {!match{$h_precedence:}{(?i)junk|bulk|list}}\
                      {match{${lookup ldap{\
                                LDAP_DEFAULT_CONTROLS \
                                ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
                                (&LDAP_DEFAULT_FILTER(LDAP_MODE=reply))}}\
                             }{reply}}\
                  }\
                  {yes}{no}\
               }
  transport = ldap_reply
```

```
# - Puede que el correo corresponda a una cuenta ldap a trave's de un alias:
# uso "driver = redirect"; la condicio'n es simplemente que "data"
# contenga la resolucio'n/es del alias al nombre de la cuenta; no "unseen".
ldap_mailalternate:
  debug_print = "R: ldap_mailalternate for ${local_part}@${domain}"
  driver = redirect
  no_verify
  no_expn
  #unseen
  check_ancestor
  data = ${map{<\n${lookup ldapm{\
                                LDAP_DEFAULT_CONTROLS \
                                ldap:///LDAP_BASEDN?LDAP_MAIL?sub?(LDAP_MAILALTERNAT
                               }}\
                }{${item}@${domain}}}
```

```
# - Puede que el correo a esa cuenta deba ser tambie'n reenviado a otra:
# usamos "driver = redirect"; la "condition" es que el forwarding este'
# expli'citamente permitido segu'n el metadato deliveryMode en ldap y que
# "data" contenga la direccio'n/es de forwarding; "unseen" hara' que se
# entregue tambie'n aqui' en virtud del siguiente router, si bien para
# permitir que so'lo se haga forwarding y no entrega, "unseen" lo
# ponemos condicional a la existencia de nolocal como valor del atributo
# mailDelivery en ldap.
```

```
ldap_forwards:
  debug_print = "R: ldap_forwards for ${local_part}@${domain}"
  driver = redirect
  check_ancestor
  #unseen

  #Evitamos que el correo para una cuenta centralizada pase por los routers
  #para correo local, haciendo el "unseen" condicional a que
  #efectivamente el correo se vaya a someter a la entrega por defecto
  #para usuarios centralizados.
  unseen = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=nolocal))\
  }}{nolocal} {no}{yes}}

  data = ${lookup ldapm{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_FORWARDS?sub?LDAP_DEFAULT_FILTER\
  }}
  condition = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=noforward))\
  }}{noforward} {no}{yes}}
```

```
# - Ahora viene el caso principal, el que nos permite almacenar el correo:
# El correo es de una cuenta centralizada, y debe ser aceptado para
# hacer el delivery por el transport por defecto correspondiente, sin
# menoscabo de otros posibles mecanismos seleccionables ma's tarde: usamos
# driver accept; la "condition" es que el usuario sea de una cuenta en
# ldap y el deliveryMode lo permita (no anuncie nolocal); "unseen" lo
# ponemos condicional a que el siguiente router vaya a ser usado
# (que deliveryMode no incluya noprogram).
ldap_dovecot_mda:
  debug_print = "R: ldap_dovecot_mda for ${local_part}@${domain}"
  driver = accept
  unseen = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=noprogram))\
  }}}{noprogram} {no}{yes}}

  condition = ${if and{\
                        {!eq{nolocal}\
                           {${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                            ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
                            (&LDAP_DEFAULT_FILTER(LDAP_MODE=nolocal))}}}\
                        }\
                        {eq{$local_part}\
                           {${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                            ldap:///LDAP_BASEDN?LDAP_MAIL?sub?\
                     (&LDAP_DEFAULT_FILTER(LDAP_MAIL=${local_part}))}}}}\
                       }\
                     {yes}{no}}
```

```
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
  transport = dovecot_mda
# - El correo puede ser entregado por otros mecanismos declarados en ldap
# en el atributo deliveryProgramPath, si bien e'ste es el u'ltimo
# tratamiento para un correo enviado a las cuentas centralizadas: uso
# "driver = redirect"; "condition" es que el deliveryMode no incluya
# noprogram; asi' como que "data" contenga el path del programa que hara'
# la entrega adicional; no "unseen".
ldap_program:
  debug_print = "R: ldap_program for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  #unseen
  data = ${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_PROGRAM?sub?LDAP_DEFAULT_FILTER\
  }}
  condition = ${if match{${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_MODE?sub?\
    (&LDAP_DEFAULT_FILTER(LDAP_MODE=noprogram))\
  }}{noprogram} {no}{yes}}
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
  pipe_transport = ldap_pipe
  reply_transport = address_reply
```

```
# - Finalmente, si un correo para una cuenta centralizada no ha sido
# entregado ya, se hara' fallar y se enviara' un correo informativo. Tal
# como se hizo para el filtro por taman~o: usamos "driver = redirect";
# "data" contiene :fail: ; "condition" como se ha descrito; el aviso esta'
# tambie'n en "data"; no "unseen".
ldap_fail:
  debug_print = "R: ldap_fail for ${local_part}@${domain}"
  driver = redirect
  allow_fail
  # En la versio'n 4.72 de exim4 no existe bool_lax, luego para que el
  # and pueda interpretar las consultas a ldap, tengo que usar
  # bool+strlen y bool+eq con 0
  # (porque necesito construir una funcio'n not{bool} que no parece
  # existir. Tampoco funciona algo "!bool")
```

```
condition = ${if \
               and{\
                   {bool{\
                    ${strlen:\
                      ${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                        ldap:///LDAP_BASEDN}}\
                     }\
                    }\
                   }\
                   {eq{\
                       ${strlen:\
                         ${lookup ldap{ LDAP_DEFAULT_CONTROLS \
                           ldap:///LDAP_BASEDN?LDAP_MAIL?sub?\
                           (LDAP_MAIL=${local_part})}}\
                        }\
                      }\
                      {0\
                      }\
                   }\
                  }{yes}{no}}
```

```
  data = :fail:\n\Delivery Status Notification (Failure)\n\
    Delivery to the following recipient failed permanently:\n\
    ${local_part}@${domain}\n\
    The error returned was:\n\
    The email account that you tried to reach does not exist.
  #local_part_suffix = -*
  #local_part_suffix_optional
  retry_use_local_part
.endif
####endfx
```

/etc/exim4/local-transports-40-ldapusers

```
vim /etc/exim4/local-transports-40-ldapusers
```

```
####fx:
.ifdef MAIN_LDAP_ENABLE
dovecot_mda:
  debug_print = "T: dovecot_mda for $local_part@$domain"
  driver = pipe
  command = /usr/lib/dovecot/deliver -d ${local_part}@${domain}  \
            -f ${sender_address}
  message_prefix =
  message_suffix =
  delivery_date_add
  envelope_to_add
  return_path_add
  log_output
  user = vmail
  temp_errors = 64 : 69 : 70: 71 : 72 : 73 : 74 : 75 : 78
```

```
ldap_pipe:
  debug_print = "T: ldap_pipe for $local_part@$domain"
  driver = pipe
  return_output
  log_output
  home_directory = ${if !eq{LDAP_DEFAULT_HOMEDIR}{} \
                        {${if eq{${substr_0_1:LDAP_DEFAULT_HOMEDIR}}{/} \
                              {LDAP_DEFAULT_HOMEDIR} \
                              {LDAP_MAILROOT/LDAP_DEFAULT_HOMEDIR} }} \
                        {LDAP_MAILROOT} }
  user = ${lookup ldap{LDAP_DEFAULT_CONTROLS \
           ldap:///LDAP_BASEDN?LDAP_UID?sub?LDAP_DEFAULT_FILTER}}
  group = vmail
  return_path_add
  delivery_date_add
  envelope_to_add
  path = "/bin:/usr/bin:/usr/local/bin"
```

```
ldap_reply:
  debug_print = "T: ldap_reply for ${local_part}@${domain}"
  driver = autoreply
  from = "${local_part}@${domain}"
  to = ${sender_address}
  subject = "Autoreply from ${local_part}@${domain}"
  text = ${lookup ldap{\
    LDAP_DEFAULT_CONTROLS \
    ldap:///LDAP_BASEDN?LDAP_REPLYTEXT?sub?LDAP_DEFAULT_FILTER\
  }}

.endif
####endfx
```

/etc/exim4/local-retry-20-ldap

```
vim /etc/exim4/local-retry-20-ldap
```

```
####fx:
#Si no usamos el sistema de quotas de exim4, no hay nada que configurar:
#.ifdef MAIN_LDAP_ENABLE
#*                        quota
#.endif
####endfx
```

/etc/exim4/local-auth-40-ldapusers-dovecotsasl

```
vim /etc/exim4/local-auth-40-ldapusers-dovecotsasl
```

```
####fx:
#http://wiki.dovecot.org/HowTo/EximAndDovecotSASL
dovecot_gssapi:
    driver = dovecot
    public_name = GSSAPI
    server_socket = /var/run/dovecot/auth-client
    server_advertise_condition = yes
    # Cuidado, la siguiente opcio'n "might break several headers in
    # mails sent by authenticated smtp".
    server_set_id=GSSAPI-${quote:$auth1}
    # E'stas NO funcionan con driver=dovecot, porque se especifican
    # en dovecot.conf
    # server_mech = GSSAPI
    # server_service = smtp
    # server_hostname = ${primary_hostname} ...dklab2.casafx.dyndns.org
    # server_realm = ${domain}              ...CASAFX.DYNDNS.ORG
```

```
####
#http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch33.html
#gssapi_sasl_server:
# driver = cyrus_sasl
# public_name = GSSAPI
# server_set_id = $auth1
# server_advertise_condition = yes
## Otras:
# server_mech = GSSAPI
# server_service = smtp
## En server_hostname, puedo poner dklab2.casafx.dyndns.org o:
# server_hostname = $primary_hostname
## En server_realm, CASAFX.DYNDNS.ORG o:
# server_realm = ${domain}

# Dejo los siguientes (login y plain) para el sistema local
####endfx
```

/etc/exim4/local-acl-config-check-data-00-clamav

```
vim /etc/exim4/local-acl-config-check-data-00-clamav
```

```
####fx:
 deny
    malware = *
    message = This message was detected as possible malware ($malware_name).
####endfx
```

/etc/exim4/local-acl-check-dkim

```
vim /etc/exim4/local-acl-check-dkim
```

```
####fx:
 acl_check_dkim:
        # Check the DKIM signature for gmail (dkim_signers y dkim_status
    # son condiciones)
        deny message            = GMail/Paypal/Ebay sender with \
                                  invalid or missing DKIM signature
        sender_domains  = gmail.com:paypal.com:ebay.com
                dkim_signers    = gmail.com:paypal.com:ebay.com
                dkim_status     = none:invalid:fail
        #...en este momento, parece razonable no verificar DKIM ma's que en
        #ciertos dominios. Gmail por volumen y Ebay/Paypal por informacio'n
        #sensibile parecen buenos candidatos y es conocido que usan DKIM,
        #de ahi' nuestra eleccio'n para este ejemplo.

        accept
####endfx
```

/etc/exim4/local-router-210-spamassasing

```
vim /etc/exim4/local-router-210-spamassasing
```

```
####fx:
spamcheck_router:
  debug_print = "R Spamassassin"
  driver = accept
  # Este router, desde el punto de vista de exim4, manda el correo a otra
  # instancia de exim4 (en un modo especial "batched" para protocolo
  # "spam-scanned", ve'ase su transport). A consecuencia de e'sto, el
  # rastro de los tests de tipo "exim4 -bt <address>" se parari'a
  # sistema'ticamente aqui'.
  # La solucio'n es evitar que este router sea tenido en cuenta en
  # estos tests, usando no_address_test:
  no_address_test
  no_verify
  #no_expn
  #domains = +local_domains : +relay_to_domains
  #          si restringimos a correo nuestro
  #
  # When to scan a message (independientemente del sistema
  # antispam usado luego)
  #   -   message size within limits
  #   -   it isn't already flagged as spam
  #   -   it isn't already scanned
  condition = ${if and { {<{$message_size}{100K}}\
                         {!def:header_X-Spam-Flag:} \
                         {!eq {$received_protocol}{spam-scanned}}} \
                         {1}{0}}
  transport = spamcheck
####endfx
```

/etc/exim4/local-transports-50-spamassasing

```
vim /etc/exim4/local-transports-50-spamassasing
```

```
####fx:
spamcheck:
    driver = pipe
    debug_print = "T: spamassassin_pipe for ${local_part}@${domain}"
    command = /usr/sbin/exim4 -oMr spam-scanned -bS
    # -bS para batched
    use_bsmtp = true
    transport_filter = /usr/bin/spamc -u ${local_part}
    home_directory = "/dev/shm"
    current_directory = "/dev/shm"
    #...nota: /dev/shm sera' sustitui'do por /run/shm a corto plazo:
    #   http://wiki.debian.org/ReleaseGoals/RunDirectory
    # must use a privileged user to set $received_protocol on the way
    # back in! #mail
    user = Debian-exim
    group = mail
    log_output = true
    return_fail_output = true
    return_path_add = false
    message_prefix =
    message_suffix =
    #ignore_status = true
   #En servidores sobrecargados que rebotan el mail por 421 SMTP timeouts
####endfx
```

Nota funcionalidad de sufijos

Como se comentó, en la configuración expuesta anteriormente, aparecen intencionadamente las opciones "local_part_suffix = -*" y "local_part_suffix_optional", pero deshabilitadas. El cometido de ambas es permitir que un correo con la parte local "umea-<sufijo cualquiera>" pueda ser interpretado como para umea de la misma forma que ocurriría si la parte local fuese simplemente "umea". Este sistema de sufijos se ideó para permitir automáticamente la entrega en diferentes mailboxes, sin necesidad de escribir filtros dedicados en algún sistema como sieve o procmail. Aunque parece una funcionalidad interesante, no se puede desplegar correctamente (sí a nivel de exim4 -y probablemente procmail- pero no a nivel de nuestro principal sistema de entrega, dovecot deliver). Los detalles se expusieron cuando dklab1.

Tests

Comprobaciones parciales de distitos subsistemas; en la sección sobre los clientes se hacía una prueba más completa.

Sintaxis; inicio de exim4 con la nueva configuración Corrección sintáctica:

```
update-exim4.conf -v
```

... si todo está bien, no debiera decir nada; en otro caso indicará dónde y en qué consiste el error. Tras solucionarlo, es también conveniente asegurarse que /var/log/exim4/paniclog esté vacío:

```
cat /var/log/exim4/paniclog
:> /var/log/exim4/paniclog


invoke-rc.d exim4 start
```

MX DNS RR Comprobaremos que están ahí:

```
dig @dklab2.casafx.dyndns.org. casafx.dyndns.org MX +short
```

```
    0 dklab1.casafx.dyndns.org.
    0 dklab2.casafx.dyndns.org.
```

SASL-GSSAPI para SMTP-AUTH con dovecot-auth y smtptest Puesto que hemos configurado exim4 para delegar la autenticación al componente dovecot-auth, esta

prueba se realizará cuando hallamos instalado y configurado Dovecot.

Sender Policy Framework Como se expuso, la base de la validación de SPF es que la IP de origen en la conexión SMTP esté admitida como origen de un correo. Esa validación se declara en un registro DNS (tipo TXT o el nuevo SPF dedicado) asociado al dominio del remitente.

```
dig @dklab2.casafx.dyndns.org. casafx.dyndns.org TXT +short
```

```
...
"v=spf1 +mx/24 ?all"
```

```
dig @dklab2.casafx.dyndns.org. casafx.dyndns.org SPF +short
```

```
...
"v=spf1 +mx/24 ?all"
```

Utilizando el intérprete spfqtool, interpretando el TXT RR (aún no soporta interpretar el SPF RR).

Si el remitente es umea@casafx.dyndns.org y la IP de origen es 10.168.1.2:

```
spfqtool -i 10.168.1.2 -s umea@casafx.dyndns.org \
         -h dklab2.casafx.dyndns.org
```

```
SPF short result:   pass
SPF verbose result: policy result: [pass] from rule [mx/24]
RFC2822 header:     Received-SPF: pass (dklab2.casafx.dyndns.org:
                    domain of umea@casafx.dyndns.org
                    designates 10.168.1.2 as permitted
                    sender) receiver=dklab2.casafx.dyndns.org;
                    client_ip=10.168.1.2;
                    envelope-from=umea@casafx.dyndns.org;
```

Pero si la IP de origen no casa con MX/24:

```
spfqtool -i 1.2.3.4 -s umea@casafx.dyndns.org \
         -h dklab2.casafx.dyndns.org
```

```
SPF short result:   neutral
SPF verbose result: policy result: [neutral] from rule [?all]
RFC2822 header:     Received-SPF: neutral (nobody: domain of
                    umea@casafx.dyndns.org is neutral
                    about designating
                    1.2.3.4 as permitted sender)
```

Si nuestra regla spf acabase en "-all" en lugar de "?all", el resultado sería "fail" y no "neutral":

```
SPF short result:   fail
SPF verbose result: policy result: [fail] from rule [-all]
RFC2822 header:     Received-SPF: fail (dklab2.casafx.dyndns.org:
                    domain of umea@casafx.dyndns.org does not
                    designate 1.2.3.4 as permitted sender)
                    receiver=dklab2.casafx.dyndns.org;
                    client_ip=1.2.3.4;
                    envelope-from=umea@casafx.dyndns.org;
```

Nota: El MTA puede añadir una cabecera con los resultados de SPF, por ejemplo (en este caso anuncia una verificación fallida):

```
Received-SPF: neutral (google.com: 79.155.184.237 is neither permitted
              nor denied by best guess record for domain of
              umea@casafx.dyndns.org) client-ip=79.155.184.237;
```

Nota Sender-ID: como se comentó, los servicios de correo de Microsoft no utilizan SPF sino un sistema similar llamado SenderID, y la interoperabilidad pasaba por registrar nuestros SPF RR en su web:
https://support.msn.com/eform.aspx?productKey=senderid&page=\
support_senderid_options_form_byemail&ct=eformts&scrx=1

DKIM Como se expuso, DKIM es un sistema más complejo que SPF: no utiliza la IP de origen, sino una firma digital, encontrándose la llave pública para verificarla, y no una mención a direcciones IP, en DNS. Conjuntamente, el sistema MTA que manda el correo debería entonces firmarlo con la correspondiente llave privada.

```
dig @dklab2.casafx.dyndns.org exim4dkim._domainkey.casafx.dyndns.org. TXT +short
```

```
    "v=DKIM1\; s=email\; g=*\; t=y\; k=rsa\; p=MIGfMA0GCSqG..."
```

```
dig @dklab2.casafx.dyndns.org _domainkey.casafx.dyndns.org. TXT +short
```

```
    "t=y\; o=~\;"
```

```
dig @dklab2.casafx.dyndns.org _adsp._domainkey.casafx.dyndns.org. TXT +short
```

```
    "dkim=unknown"
```

Como se aclaró:

- dkim=unknown mismo significado que o=~ pero en registro ADSP, el equivalente a
- o=- en ADSP sería dkim=all, pero por precaución utilizamos unknown al menos durante un período de pruebas.

Aprovechando las capacidades de depuración de exim4, podíamos diseñar pruebas:

Correo entrante (verificación DKIM) Cuando un correo entrante es verificado, aparece en el log de exim4 algo como:

```
DKIM: d=gmail.com s=gamma c=relaxed/relaxed a=rsa-sha256 [verification succeeded]
```

Utilizábamos la posibilidad de exim4 para recibir una sesión SMTP desde stdin, simulábamos entonces la llegada de un correo desde gmail.com con una cabecera DKIM-Signature inventada:

```
cat <<EOF > ~/mailtest/receive_dkim.sh
 (sleep 2; \
 printf "EHLO smtp.gmail.com\r\n" | tee /dev/stderr; sleep 1; \
 printf "MAIL FROM: cocnarf@gmail.com\r\n" | tee /dev/stderr; sleep 1; \
 printf "RCPT TO:aemu@casafx.dyndns.org\r\n" | tee /dev/stderr; sleep 2; \
 printf "DATA\r\n" | tee /dev/stderr; sleep 1; \
 printf "Subject: dkim\r\n" | tee /dev/stderr; \
 printf "From: cocnarf@gmail.com\r\n" | tee /dev/stderr; \
 printf "To: aemu@casafx.dyndns.org\r\n" | tee /dev/stderr; \
 printf "DKIM-Signature: v=1; a=rsa-sha256; q=dns/txt;
c=relaxed/relaxed; d=casafx.dyndns.org;
s=exim4dkim;h=Content-Type:MIME-Version:Message-ID:Subject:To:From:Date;
bh=mxBrUUox23UseRX2N7nW3GvifUa4W01XXlWCe9ZcMtQ=;b=jcbip6cxlU5ykdDkMStiOW
kW50REOOtyamDMGF4qmvFZAHFkY0G8a+g12aCEx/YJsM2mKmWkfMCxFnYpK0JDWGOasJdzo1
LFSKb/NYDECFNkHmqmjfg0DbH2FlNQ50Cg6ZRznw1YLGw0sotWQiwsdGkSuqA8hWeoRGReeF
E+f4M=;\r\n" | tee /dev/stderr; \
 printf "\r\n" | tee /dev/stderr; \
 printf "\r\n" | tee /dev/stderr; \
 printf "Hola cara de bola\r\n" | tee /dev/stderr; \
 printf ".\r\n" | tee /dev/stderr; \
 printf "QUIT\r\n" | tee /dev/stderr ) \
 | exim4 -bh `dig smtp.gmail.com A +short | tail -n 1`
                # smtp.gmail.com, es decir 209.85.229.108
EOF
```

```
sh ~/mailtest/receive_dkim.sh 2>&1 | grep DKIM
```

... y veremos aparecer información del proceso de verificación que lleva a cabo exim4. Puesto que la cabecera DKIM-Signature es inventada, acaba fallando, pero el despliegue en sí parece funcionar:

```
LOG: 1RSkKI-0001nW-Iq DKIM: d=casafx.dyndns.org s=exim4dkim
     c=relaxed/relaxed a=rsa-sha256 [verification failed - signature
     did not verify (headers probably modified in transit)]
```

Correo saliente (firmado DKIM) Este script hará SMPT-AUTH y mandará un correo a una cuenta no gestionada por nosotros, como por ejemplo bit-bucket@test.smtp.org (como su nombre sugiere, es una cuenta libremente disponible para este tipo de pruebas):

```
cat <<EOF > ~/mailtest/send_dkim.sh
(sleep 2; \
printf "MAIL FROM: umea@casafx.dyndns.org\r\n" | tee /dev/stderr; sleep 1; \
printf "RCPT TO:bit-bucket@test.smtp.org\r\n" | tee /dev/stderr; sleep 2; \
printf "DATA\r\n" | tee /dev/stderr; sleep 1; \
printf "Subject: dkim\r\n" | tee /dev/stderr; \
printf "From: umea@casafx.dyndns.org\r\n" | tee /dev/stderr; \
printf "To: bit-bucket@test.smtp.org\r\n" | tee /dev/stderr; \
printf "\r\n" | tee /dev/stderr; \
printf "Testing dkim signing.\r\n" | tee /dev/stderr; \
printf ".\r\n" | tee /dev/stderr; \
printf "QUIT\r\n" | tee /dev/stderr ) \
| smtptest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
  -f /dev/stdin dklab2.casafx.dyndns.org
EOF
```

```
cp ~/mailtest/send_dkim.sh /tmp
```

Con el scrip anterior vamos a mandar un correo (con el objetivo de que sea firmado por el código DKIM de exim4), pero nuestra configuración actual hace obligatoria la autenticación y, sabemos, aún no tenemos disponible al proceso encargado de ella, dovecot-auth. Podríamos deshabilitar la necesidad de hacer SMTP-AUTH ahora, pero vamos a tomar otro camino más sencillo e interesante. Vamos a utilizar temporalmente (sólo para esta prueba) la alternativa de exim4 para poder autenticar con SASL-GSSAPI, es decir, su enlace con la librería de cyrus-sasl en lugar de su interfaz a dovecot-auth. Ésto nos va a permitir dar un ejemplo de configuración de esta alternativa, además de apenas modificar

los ficheros de configuración. Para ello escribirmos la configuración de uso con cyrus-sasl bajo /tmp/ y utilizando sed haremos que el ".include" relativo a la autenticación apunte temporalmente a él en lugar de al fichero definitivo bajo /etc/exim4/.

```
cat <<EOF > /tmp/exim4-local-auth-40-ldapusers-cyrussasl
#http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch33.html
gssapi_sasl_server:
 driver = cyrus_sasl
 public_name = GSSAPI
 server_set_id = \${auth1}
 server_advertise_condition = yes
# Otras:
 server_mech = GSSAPI
 server_service = smtp
# En server_hostname, puedo poner dklab2.casafx.dyndns.org o:
 server_hostname = \${primary_hostname}
# En server_realm, CASAFX.DYNDNS.ORG o:
 server_realm = \${domain}
EOF

sed -i 's/\.include.*local-auth.*dovecotsasl'/\
'\.include \/tmp\/exim4-local-auth-40-ldapusers-cyrussasl/g' \
/etc/exim4/exim4.conf.template
```

Lanzábamos una instancia de exim4 con soporte de depuración y sin soporte AFS (pues no lo necesitamos a la vez que impide ver los mensajes de depuración en la consola). Para ello en lugar de utilizar el script init modificado, utilizamos el original que salvaguardamos antes de aquellas modificaciones.

Paramos la instancia actual.

```
invoke-rc.d exim4 stop
```

Indicamos que se use el código de depuración en la siguiente ejecución:

```
update-exim4defaults --commonoptions '-d+all'
```

exim4 directamente (en lugar de dovecot-auth) debe poder leer el keytab por lo que lo hacemos accesible al uid Debian-exim y lo apuntamos con la variable KRB5_KTNAME:

```
chown Debian-exim /etc/keytab.d/dovecot.keytab
KRB5_KTNAME=/etc/keytab.d/dovecot.keytab \
~/mailtest/etc-init.d-exim4 start 2>&1 | grep -i dkim
```

Con lo anterior, una instancia de exim4 está cargada y mandando mensajes en su terminal. Entonces, en otra consola distinta, hacíamos al usuario umea mandar el correo a través del script anterior:

```
login umea
sh /tmp/send_dkim.sh
exit
```

Si volvíamos a la consola con exim4, veíamos que exim4 firma el correo:

```
  ...
  dkim-signature:v=1;{SP}a=rsa-sha256;{SP}q=dns/txt;{SP}c=relaxed/relaxed;


{SP}d=casafx.dyndns.org;{SP}s=exim4dkim;{SP}h=To:From:Subject:Message-Id
:Date;{SP}bh=JWj+...
  ...
```

Todo correcto, firma el correo saliente. Pulsamos Control-c en la consola con exim4 para pararlo. Ya entonces, para volver el sistema al estado inicial:

```
chown root /etc/keytab.d/dovecot.keytab
ls -l  /etc/keytab.d/dovecot.keytab
```

```
-rw------- 1 root root 1086 2011-12-20 17:10 /etc/keytab.d/dovecot.keytab
```

```
sed -i 's/\.include.*local-auth.*cyrussasl'/\
'\.include \/etc\/exim4\/local-auth-40-ldapusers-dovecotsasl/g' \
/etc/exim4/exim4.conf.template


update-exim4defaults --force --commonoptions ''
invoke-rc.d exim4 start
```

Enrutamiento del correo saliente

Nota introductoria sobre los tests de enrutamiento Volvemos a repetir los pormenores de exim4 para entender los tests de enrutamiento posteriores:

En terminología de exim4, "dominios locales" son todas aquellos domain-part (lo que sigue a la "@" en una dirección de correo) para los que se aceptará el correo y será entregado, en general, aquí (correo entrante). Estos dominios son, por defecto, localhost y el <FQDN>de la máquina. Nosotros añadimos otro, "casafx.dyndns.org", de hecho la pregunta "Other destinations for which mail is accepted:" al instalar exim4 nos pedía adelantar ésto y así lo introducimos.

```
exim4 -bP +local_domains
```

```
    @:localhost:casafx.dyndns.org
```

...el valor de la variable lo toma del contenido de la macro LOCAL_DOMAINS, que es inicializada a partir del /var/lib/exim4/config.autogenerate.

```
grep LOCAL_DOMAINS= /var/lib/exim4/config.autogenerated
```

```
    @:localhost:casafx.dyndns.org
```

Nota: "@" en ese domainlist corresponde al FQDN según la documentación.

```
exim4 -bP primary_hostname
```

El contenido de esta variable, +local_domains, da sentido a toda la secuencia de routers por las que pasa un correo para decidir su ruta y destino. Veamos; en orden descendente los routers serían:

- Un primer grupo de routers se encargan de comprobar que el domain-part no pertenece a +local_domains y es, entonces, un correo saliente. Son routers predefinidos

por el paquete de exim4 en debian, que anteceden a los nuestros, y que pretendemos testear en este apartado.

- El segundo grupo de routers son los que hemos diseñado en este despliegue, y encamina los correos destinados a una cuenta con domain-part casafx.dyndns.org. Además si uno de estos correos no llega a coincidir completamente con ningún router, es rebotado (y no sigue al resto de routers del tercer grupo).
- El tercer y último grupo de routers, comprueban que el domain-part pertenezca a +local_domains, entonces hacen una entrega local. Por tanto, en la práctica, se ocupan de correos con domain-part "localhost" y el <FQDN>. Como los del primer grupo, son routers predefinidos por el paquete de exim4 en debian.

Nota: si se intenta enviar un correo a un destinatario no cualificado (sin indicar @<domain-part>) exim4 lo reescribe para sufijar con @${qualify_domain}, siendo qualify_domain una variable de exim4. Si rastreamos la configuración, esa variable recibe como valor el macro ETC_MAILNAME, que es inicializado con el contenido del fichero /etc/mailname. La pregunta "System mail name:" al instalar exim4 se refería a este contenido, y convenientemente respondimos el FQDN de la máquina. Conclusión: un destinatario sin cualificar se cualifica con @dklab2.casafx.dyndns.org en esta máquina. ([[http://wiki.debian.org/EtcMailName][re

```
cat /etc/mailname

#...fue incorporado a la configuracio'n, veremos, como ETC_MAILNAME:
```

```
   dklab2.casafx.dyndns.org
```

```
egrep '(^ETC_MAILNAME|quali.*ETC_MAILNAME)' /var/lib/exim4/config.autogenerated
```

```
ETC_MAILNAME=dklab2.casafx.dyndns.org

qualify_domain = ETC_MAILNAME
```

Tests del correo saliente Routers del primer grupo:

```
view /etc/exim4/exim4.conf.template
```

```
...
begin routers


domain_literal:
...
```

Routers correo saliente y su condición de aplicación	
domain_literal	si tras la @ viene la dirección IP del servidor SMTP remoto
hubbed_hosts	si para ciertos dominios no usa dns sino un fichero especial
dnslookup_relay_to_domains	usa dns; es igual que el siguiente, sólo que éste no lidia
dnslookup	con una situación especial

Suponiendo que cern.ch no pertenece a nuestros +local_domains:

```
exim4 -bt recruitment.service@cern.ch
```

```
recruitment.service@cern.ch
  router = dnslookup, transport = remote_smtp
  host cernmxgwlb2.cern.ch [137.138.144.183] MX=10
```

Todos los detalles añadiendo -v -d+all

```
exim4 -v -d+all -bt recruitment.service@cern.ch 2>&1 | less
```

Nota: explicamos que, específicamente, un correo desde dklab2 a dklab1 fallará:

```
exim4 -v -f umea@casafx.dyndns.org -bt umea@dklab1.casafx.dyndns.org
```

```
...
R: dnslookup for aemu@dklab1.casafx.dyndns.org
aemu@dklab1.casafx.dyndns.org is undeliverable: Unrouteable address
```

La razón es que el router dnslookup ignora la resolución de dominios para ciertos segmentos de red predefinidos:

```
grep -B10 -A10 ignore_target_hosts /etc/exim4/exim4.conf.template|less
```

```
dnslookup:
  ...
  ignore_target_hosts = 0.0.0.0 : 127.0.0.0/8 : 192.168.0.0/16 :\
                        172.16.0.0/12 : 10.0.0.0/8 : 169.254.0.0/16 :\
                        255.255.255.255
  no_more
  ...
```

... corresponde al administrador hacer o no modificaciones a esa lista.

Enrutamiento del correo a cuentas centralizadas Routers del segundo grupo, los que hemos diseñado en este despliegue:

```
...
begin routers


domain_literal:
...
.include /etc/exim4/local-router-250-ldap
...
```

Para poder conocer con total exactitud el comportamiento de nuestro código (string-expansion etc) le pasaremos a exim4 los flags -v y, sobre todo, -d+all, activando el modo depuración. Siempre es conveniente pasar -f para que tenga disponible un remitente.

```
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt aemu@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt umea@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt postmaster@casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt noexiste@casafx.dyndns.org 2>&1 | less
```

Enrutamiento del correo a cuentas locales en dklab2 Routers del tercer grupo: cuentas cuyo domain-part es localhost, el FQDN o, equivalentemente a éste por el sistema de cualificación que explicamos, nada.

```
...
begin routers


domain_literal:
...
.include /etc/exim4/local-router-250-ldap
...
real_local:
...
```

Routers para el correo a cuentas locales y su condición de aplicación	
real_local	si es dominio local _y_ usa sistema prefijos ${local_part}-real; no importa
system_aliases	si te encuentra en /etc/aliases
userforward	si encuentra ~/.forward
procmail	si encuentra ~/.procmail
maildrop	si encuentra ~/.mailfilter
lowuid_aliases	caso especial de aliases para entidades no persona
local_user	es un dominio local, y es un usuario local "check_local_user"
mail4root	el caso anterior especializado en el usuario “root”.

... local_user es el router cuyo transport almacena el correo en /var/mail/<local-part>, y es el destino habitual de los correos a cuentas locales.

```
getent passwd root user
```

```
root:x:0:0:root:/root:/bin/bash
user:x:1000:1000:Debian User,,,:/home/user:/bin/bash
```

Con domain-part dklab2.casafx.dyndns.org:

```
exim4  -bt user@dklab2.casafx.dyndns.org
exim4  -bt root@dklab2.casafx.dyndns.org
```

...o, si queremos ver todos los detalles, añadimos -v -d+all:

```
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt user@dklab2.casafx.dyndns.org 2>&1 | less
exim4 -d+all -v -f umea@casafx.dyndns.org \
      -bt root@dklab2.casafx.dyndns.org 2>&1 | less
```

Sin domain-part (se cualifican con @dklab2.casafx.dyndns.org):

```
exim4 -bt user
```

Con domain-part localhost:

```
exim4 -bt user@localhost
```

Por último, y no menos importante, un correo a umea (cuenta centralizada, resolvible por NSS a través de ldap) cuando domain-part no es casafx.dyndns.org, no se tratará como un correo a cuenta centralizada sino como local y, entonces, se entrega en /var/mail/umea:

```
exim4 -bt umea
#...para ver todos los detalles, como siempre usamos "-d+all -v" :
exim4 -d+all -v -f umea@casafx.dyndns.org -bt umea 2>&1 | less
```

Gestión de cola; monitorización Debieran mostrar una cola vacía en este momento:

```
mailq
exipick -i | while read L; do exim4 -Mvh $L; exim4 -Mvb $L; done
pfqueue
```

Respecto a la monitorización:

```
tail -f /var/log/exim4/mainlog
exiwhat # pregunta a exim4 en que' esta' ocupado.
geximon # si queri'amos una aplicacio'n gra'fica lo podemos instalar
```

Estadísticas El comando eximstats puede inspeccionar los losgs y presentarnos algunos estadísticos:

```
eximstats /var/log/exim4/mainlog* | less
```

Si queríamos generar html con gráficos, podíamos:

```
apt-get install libgd-gd2-noxpm-perl libgd-text-perl libgd-graph-perl
eximstats -html \
          -charts -chartdir /tmp/eximstats.charts \
          -chartrel /tmp/eximstats.charts \
          /var/log/exim4/mainlog* > /tmp/eximstats.html


w3m /tmp/eximstats.html
```

1.3.9. MDA, IMAP y MANAGESIEVE

Como sabemos, el proyecto Dovecot provee todas las funcionalidades: IMAP server (programa imap), MANAGESIEVE server (programa managesieve) y componente MDA del lado del servidor (programa deliver). Adicionalmente mostrábamos las capacidades de MDA de procmail, también compatible con AFS y mostrando así otro estilo de entrega, esta vez bajo el home del usuario.

Sistema de configuración dovecot

/etc/dovecot/dovecot.conf controla el grueso de la configuración, /etc/dovecot/dovecot-ldap.conf controla específicamente lo relativo al uso de ldap.

/etc/dovecot/dovecot.conf

```
vim /etc/dovecot/dovecot.conf
```

```
####fx:
#http://wiki.dovecot.org/MainConfig

#!include /etc/dovecot/conf.d/*.conf
#!include_try /etc/dovecot/extra.conf
#Nota: Los include no funcionan au'n con el programa delivery

#mail_debug = yes
####endfx
...
```

```
####fx:
# Protocolos de internet:
protocols = imap imaps managesieve # pop3 pop3s
#-protocols = imap imaps
####endfx
...
```

```
####fx:
# Logging: por defecto, el logging se hace a syslog;
####endfx
log_timestamp = "%Y-%m-%d %H:%M:%S "
...
```

```
## SSL settings
##
####fx:
# TLS: http://wiki.dovecot.org/MainConfig#SSL_settings
ssl = yes #no, required. Adicionalmente, por defecto ssl_listen
          #coincide con listen.
ssl_cert_file = /etc/dovecot/dovecot-2.casafx.dyndns.org.crt #pem format
ssl_key_file =  /etc/dovecot/dovecot-2.casafx.dyndns.org.key #pem format
##ssl_verify_client_cert = no # Existe require en lugar de verify
###The CAfile should contain the CA-certificate(s) followed by the matching CRL(s).
##ssl_ca_file = /etc/ssl/certs/casafx-ca.crt
#verbose_ssl = no
ssl_parameters_regenerate = 0 # evita que el proceso ssl-build-params
                              # (por defecto cada semana), regenere DH.
                              # De utilidad si el ordenador/emulador es
                              # lento.
####endfx
...
```

```
#verbose_ssl = no
####fx:
#UID's: dovecot es -1. Lanzado como root (uid 0)
#                  -2. Ciertas acciones del procesamiento del login se hacen
#                      como user dovecot (o lo que diga "login_user"). Nada en
#                      el sistema deberi'a tener permiso para este usuario
#                      excepto lo que pueda crear el servidor. No pertenece a
#                      ma's grupo que al suyo y a su grupo no pertenece nadie
#                      ma's que e'l, ni puede ser usado tampoco para
#                      los dema's roles:
login_user = dovecot
#                  -3. El acceso al correo se hace con otro usuario,
#                      cuyo uid/gid se puede configurar ahora y ser el
#                      valor por defecto si no se especifica otra cosa
#                      por seccio'n auth {} y, alli', por usuario en
#                      el metadato userdb:
mail_uid = vmail
#mail_gid = vmail #Adema's, para uid y gid se pueden declarar li'mites:
first_valid_uid = 800 #8 corresponde a la cuenta local "mail", pero
                     #nosotros usamos vmail 800.
last_valid_uid =  800
first_valid_gid = 800
last_valid_gid =  800
```

```
 #Acorde con la debian policy, el tradicional directorio spool del
 #correo unix (/var/mail) debe tener como grupo a "mail" y permisos de
 #grupo rws, de forma que los MUA puedan manejar los recipientes
 #mbox si su uid pertenece al grupo "mail". Si en un futuro necesitamos
 #dar soporte IMAP o mda para esas cuentas locales, compatiblemente
 #a los uid y gid anteriores podemos indicarle a dovecot el grupo
 #para esa situacio'n concreta:
#http://www.debian.org/doc/debian-policy/ch-customized-programs.html\
#                                          #s-mail-transport-agents
mail_privileged_group = mail
#                  -4. Finalmente, dovecot hace las resoluciones para
#                      userdb/passdb como root, o lo que diga "user" en
#                      seccio'n de auth (ve'ase luego)
####endfx
...
```

```
## Login processes
##
####fx:
login_process_per_connection = yes # more secure vs faster
disable_plaintext_auth = yes
max_mail_processes = 512 #when this limit is reached, new users aren't
                         #allowed to log in
mail_process_size = 256  #in megabytes. Most of the memory goes to
                         #mmap()ing files
####endfx
...
```

```
## Mailbox locations and namespaces
##
####fx:
#Mail location: http://wiki.dovecot.org/MailLocation
 # mail_location define el valor por defecto, pero puede ser redefinido
 # en en la seccio'n auth {} por el metadato mail de la userdb
 # (adicionalmente, puede sobreescribirse si se utilizan imap
 # namespaces, no es el caso -ver wiki- pero sirven para mezclar
 # formatos o compartir algunas mailboxes
 # http://wiki.dovecot.org/SharedMailboxes/Public).
 # Los i'ndices se crean en el directorio maildir si no se dice otra cosa,
 # aunque puede redefinirse otra ruta para ellos, incluso almacenarse en memoria.
 #mail_location = maildir:~/Maildir
 #mail_location = mbox:~/mail:INBOX=/var/mail/%u
 #mail_location = maildir:~/Maildir:INBOX=~/Maildir/.INBOX
 #mail_location = \
 #     maildir:/afs/cell/home/%u/Maildir:INDEX=/var/lib/dovecot-index/%u
mail_location = maildir:/afs/%Ld/service/mail/%1n/%2n/%n/Maildir
#
# AFS support:
#http://nocones.blogspot.com/2007/05/afs-postfix-dovecot-and-maildirs.html
#http://wiki.dovecot.org/MainConfig#Mail_processes
#http://wiki.dovecot.org/MainConfig#Maildir-specific_settings
#http://wiki.dovecot.org/NFS
```

```
 #Keep indexes out of AFS
 #...no puedo, creo, poner el index en memoria ni localmente, pero al fin
 #y al cabo los i'ndices son archivos comunes, solamente
 #se penaliza el desempen~o que se intenta ganar (ve'ase discusio'n en
 #http://wiki.dovecot.org/NFS)

 #Disable link() copies:
maildir_copy_with_hardlinks = no

 #Next, disable mmap and enable dotlocks. I would think you shouldn't need
 #these but this is what I ended up with that worked. Test yourself and see:
mmap_disable = yes
dotlock_use_excl = yes
lock_method = dotlock
#maildir_very_dirty_syncs = yes # Assume Dovecot is the only MUA
                                # accessing Maildir:
 #                                Scan cur/ directory only when its
 #                                mtime changes unexpectedly or when we
 #                                can't find the mail otherwise.
####endfx
...
```

```
# If you need to set multiple mailbox locations or want to change default
# namespace settings, you can do it by defining namespace sections.
####fx:
# Imap Namespaces: http://wiki.dovecot.org/Namespaces
# ( http://www.faqs.org/rfcs/rfc2342.html )
# Aqui' se van a usar para ofrecer mailboxes pu'blicas:
# http://wiki.dovecot.org/SharedMailboxes/Public
# Por defecto las mailboxes residen en un namespace privado que dovecot crea
# automa'ticamente. Para poder tener mailboxes de acceso pu'blicas, es necesario
# primero explicitar el namespace privado (por alguna razo'n no se hace ahora
# automa'ticamente) y a continuacio'n hay que declarar un namespace "Public"
# para las nuevas mailboxes pu'blicas:
namespace private {
  # Simplemente lo configuramos para que no cambie nade respecto a si no
  # estuvie'semos definiendo namespaces adicionales. Asi' pues, la
  # localizacio'n del correo en este espacio es la definida antes en
  # mail_location
  separator = /
  prefix =
  inbox = yes # so'lo un namespace puede afirmar tener la inbox del usuario
}
#... podri'amos definir otros namespace private {}, por ejemplo para tener
#    mailboxes en otro formato (ejem mbox). No ejemplificaremos ese caso.
```

```
#    Vayamos ya al namespace public:
namespace public {
  # Si'mbolo para separar los namespaces en jerarqui'a; debe ser el
  # mismo en todos.
  separator = /
  # Si se debiera listar este namespace si el cliente envi'a el
  # comando imap NAMESPACE:
  hidden = no
  # Si este namespace y sus mailboxes debieran ser listadas ante
  # un comando LIST:
  prefix = Public/
  # Si este namespace y sus mailboxes debieran ser listadas ante comando LIST:
  list = yes # "children" hari'a que listase so'lo si el espacio
             # realmente contiene mailboxes.
```

Si este espacio gestiona las subscripciones a sus mailboxes o ésto se gestiona en el espacio padre (en este caso "/", en el namespace privado) lo diferenciamos con 'subscriptions = yesó 'subscriptions = no'respectivamente. En despliegues donde "Public/" corresponde a una ruta en el sistema de archivos en que el proceso que accede no puede escribir, (ej: deliver u otro sistema escribe al hacer la entrega, pero imap no puede) entonces eligir 'subscriptions = no'permite cambiar la ruta a la del espacio "/", donde presumiblemente sea posible crear y escribir un fichero 'subscriptions'(ej: imap puede escribir en el home del usuario). Pero nuestro caso no es típico, veamos entonces: la ruta para el espacio raiz "/" (/afs/<realm>/service/mail/<dobleNivelUser>) es escribible, pero la del espacio "Public/" también porque aunque la ruta para las mailboxes no lo es (/afs/<realm>/service/Public), la de los ficheros de control de ese espacio los declararemos separadamente bajo el home del usuario en un directorio escribible (/afs/<realm>/user/<dobleNivelUser>/.PublicMailboxIndexes).

Por tanto en nuestro caso ambas posibilidades permiten al usuario gestionar sus subscripciones porque imap puede a su vez gestionar un fichero 'subscriptionsén ambas localizaciones. Nos parece pertinente entonces que las subscripciones a este espacio se gestionen en la ruta de control de este espacio como siempre, así que elegimos:

(La sentencia que define "location.ªparece dividida en 3 partes delimitadas por \ y cambio de línea. Realmente debe aparecer en una sóla línea, sin \ y cambio de línea:)

```
  subscriptions = yes
  # Rutas para las mailboxes y rutas para los archivos de control:
  # nota: hay que ponerlo en una so'la li'nea, sin usar el escape "\"
  location = maildir:/afs/%Ld/service/mail/Public:\
             CONTROL=~/.PublicMailboxIndexes:\
             INDEX=~/.PublicMailboxIndexes
#...recordamos que debiera aparecer en una sola li'nea.
}
#... podri'amos definir otros namespace public {}, por ejemplo para
#    crear una jerarqui'a bajo Public/, por ejemplo Public/SedeA
#    Public/SedeB... tambie'n existen namespace shared {} para
#    compartir entre grupos reducidos. No ejemplificaremos ambos casos,
#    pero lo anterior es suficiente para hacerse una idea y evaluar si
#    son necesarios en otro despliegue.
####endfx
...
```

...así con esas rutas, como se adelantó al hablar de las subscripciones:

- las mailboxes de este espacio estarán bajo /afs/<realm>/service/mail/Public, una carpeta no modificable por los usuarios, imap o deliver.
- Por ese carácter de sólo lectura los índices y archivos de control los hemos movido a un lugar escribible por imap (probablemente mejorará el desempeño si está en otro volumen afs, como el del home), /afs/<realm>/user/<DobleNivelUser>/.PublicMailboxIndexe donde los servicios de correo (grupo afs "mailgroup") sí pueden escribir.
- Nótese que el fichero 'subscriptionsén esa ruta bajo AFS, puede ser convertido en de sólo lectura y, así, fijar unas subscripciones al usuario para el espacio "Public/". Para ello consúltese la relación entre ACL afs y posix en http://docs.openafs.org/UserGuide/ch04s08. y revísese el capítulo que presentaremos más tarde para crear los recursos de almacenamiento en AFS.

```
protocol imap {
####fx:
# IMAP protocol: http://wiki.dovecot.org/MainConfig#IMAP_specific_settings
  listen = *   #*,[::]
  login_executable = /usr/lib/dovecot/imap-login
  mail_executable = /usr/lib/dovecot/imap
  #mail_executable = /usr/lib/dovecot/rawlog /usr/lib/dovecot/imapn
  #mail_executable = /usr/lib/dovecot/gdbhelper /usr/lib/dovecot/imap

  mail_plugins = quota imap_quota
#}
 #protocol pop3 {
 # pop3_uidl_format = %08Xu%08Xv
 # mail_plugins = quota
 #}
####endfx
...
```

```
protocol managesieve {
####fx:
# Managesieve protocol: http://wiki.dovecot.org/ManageSieve
#                       http://tools.ietf.org/html/rfc5804
#Configuramos lo relativo al demonio y, lo relativo al lugar de
#almacenamiento se configurara' en la seccio'n "plugins" puesto
#que es comu'n con el soporte de sieve de la funcionalidad MDA.
#Nota: supongo que managesieve se lanza con uid el de mail_uid
#http://wiki.dovecot.org/ManageSieve/Configuration
  #Daemon:
  listen = *:4190
  mail_executable = /usr/lib/dovecot/managesieve
  login_executable = /usr/lib/dovecot/managesieve-login
  #Storage: ve'ase seccio'n "plugin"
#}
####endfx
...
```

```
## MDA (aka LDA) specific settings
##
####fx:
# Delivery: http://wiki.dovecot.org/LDA
protocol lda {
  #Cabeceras para rebotar mails:
   #From:
   postmaster_address = postmaster@casafx.dyndns.org
   #Para Message-IDs and in Reporting-UA:
   hostname = dklab2.casafx.dyndns.org

  #Soporte para reenvi'os:
  sendmail_path = /usr/sbin/sendmail

  # Socket para consultar metadatos userdb al proceso de auth de dovecot
  # (ver seccio'n auth) al llamar el MDA con -d user@host:
  auth_socket_path = /var/run/dovecot/auth-master
```

```
  #Log: si utilizo syslog, no tengo que plantearme si el usuario vmail
  #puede escribir en el fichero de log de dovecot; para utilizar
  #syslog, vaci'o las variables:
  log_path =
  info_log_path =
  syslog_facility = mail

  #Plugins:
  mail_plugins = quota sieve
  #...la configuracio'n de los plugins se declara posteriormente en la
  #   seccio'n plugin {}
# Nota: adicionalmente se puede usar el deliver directamente (ejem en
#       .forward, ver wiki)
}
####endfx
...
```

```
auth default {
####fx:
# Subservicios auth / metadata lookup:
# http://wiki.dovecot.org/Authentication
  # Las bu'squedas de metadatos van a realizar con la cuenta local:
  user = root

  #http://wiki.dovecot.org/Authentication/Mechanisms
  mechanisms = gssapi #plain login cram-md5...
  auth_default_realm = CASAFX.DYNDNS.ORG
  auth_krb5_keytab = /etc/keytab.d/dovecot.keytab
  auth_gssapi_hostname = dklab2.casafx.dyndns.org
  #auth_verbose
  #auth_debug
  #auth_debug_passwords

  userdb ldap {
    args = /etc/dovecot/dovecot-ldap.conf # Alli' declaramos igualmente
                                          # los metadatos esta'ticos
  }
```

```
#
#No utilizo base de datos de password al usar el mecanismo gssapi.
#You can use multiple databases, so if the password doesn't match in
#the first database, Dovecot checks the next one.
#http://wiki.dovecot.org/PasswordDatabase
#passdb ldap {
#  args = /etc/dovecot/dovecot-ldap.conf
#}
#Prefetch metadata de userdb cuando obtines la de passdb.
#userdb prefetch { }
#

#Exportamos a otros programas los subservicios auth / metadata lookup:
socket listen {

  #Master socket, usado por el dovecot lda (deliver) para hacer lookups:
  master {
    path = /var/run/dovecot/auth-master
    #Los permisos se configuran teniendo en cuenta que a trave's de
    #este socket se resuelven metadatos userdb; por seguridad
    #restringimos a:
    mode = 0600
    user = vmail
  }
```

```
    #Client socket, usado por el MTA (exim4) para SMTP-AUTH
    client {
      path = /var/run/dovecot/auth-client
      #Es seguro exportarlo a cualquiera, en general, en este caso
      #lo fundamental es que sea accesible para la cuenta de exim4:
      mode = 0660
      user = Debian-exim
      group = mail
    }
  }

# No uso modo proxy a otros dovecot en otros hosts donde realmente estuviese
# el correo: http://wiki.dovecot.org/PasswordDatabase/ExtraFields/Proxy
#            http://wiki.dovecot.org/HowTo/ImapProxy
####endfx
...
```

```
####fx:
# Deshabilito, au'n en "auth default {}", la configuracio'n
# preexistente de mecanismos
#-mechanisms = plain
####endfx
...
####fx:
# Deshabilito, au'n en "auth default {}", la configuracio'n preexistente
# de passdb
#-passdb pam {
####endfx
...

    #args = dovecot
####fx:
#cierro llave de la seccio'n passdb pam deshabilitada.
#-}
####endfx
...
```

```
####fx:
# Deshabilito, au'n en "auth default {}", la configuracio'n preexistente
# de userdb
#-userdb passwd {
####endfx
...
    #args =
####fx:
#cierro llave de la seccio'n userdb passwd deshabilitada.
#-}
####endfx
...
}
...
```

```
## Dictionary server settings
##
####fx:
# Subservicio de almacenamiento de pares llave-valor (no lo uso: nada que hacer)
####endfx
dict {
}
```

```
plugin {
####fx:
 # Q u o t a: http://wiki.dovecot.org/Quota/1.1
 # "maildirsize" format y otros tipos de quota:
 #  http://wiki.dovecot.org/Quota/Maildir
 # (imap quota spec http://www.rfc-editor.org/rfc/rfc2087.txt)
 #
 # quota = <backend>:<quota root name>:<backend args>. Ejems:
 #  quota = maildir:user quota
 #  quota2 = maildir:Shared quota:ns=Public/
 #  quota3 = fs:Disk quota
 # quota_rule = <mailbox name>:<limit conf> #siendo los li'mites:
 #storage=nKBytes bytes=nBytes messages=nMensajes backend=tipo ignore
 #  quota_rule = *:storage=1G # '*' es para cualquiera que no tenga
 #                                quota_rule definida.
 #  quota_rule3 = Trash:storage=20%%
 #  quota_rule2 = SPAM:ignore
 # quota_warning[n] = <limit configuration> <command to run>  # O bien:
 # quota_exceeded_message = Quota exceeded, please go to\
 #    http://www.example.com/over_quota_help for instructions \
 #    on how to fix this.
 #
```

```
## Plugin settings
##
####fx:
# Finalmente, configuracio'n de plugins:
####endfx
```

```
# Vamos alla':
 quota = maildir:User quota
#quota_rule lo definiremos por cada usuario en dovecot-ldap.conf
 quota_warning = storage=90%% /usr/local/bin/quota-warning.sh 90
#"the warning is ONLY executed at the exact time when the limit
# is being crossed"
```

```
 # S i e v e: http://wiki.dovecot.org/LDA/Sieve/Dovecot
 #            http://wiki.dovecot.org/ManageSieve
 # (sieve spec      : http://tools.ietf.org/html/rfc5228)
  # sieve_before apunta a un directorio con scripts .sieve ejecutados
  # en orden alfabe'tico previamente al de usuario. Nota: Existe
  # sieve_after.
  sieve_before = /afs/casafx.dyndns.org/service/mail/sieve/
  # Enlace simbo'lico al script activo de usuario (u'nico, pero puede
  # incluir otros)
  sieve = /afs/%Ld/service/mail/%1n/%2n/%n/.dovecot.sieve
  # Do'nde se almacenan los scripts subidos (los del espacio de nombres
  # :personal):
  sieve_dir = /afs/%Ld/service/mail/%1n/%2n/%n/sieve
  # Sieve permite hacer includes a scripts en espacio :personal o en
  #espacio :global accesibles por cualquiera, sieve_global_dir dice
  #do'nde residen:
  #sieve_global_dir = <path>
  # Script que se ejecutari'a si no se encuentran scripts de usuario:
  #sieve_global_path =
  # Todas las extensiones sieve esta'n activas (excepto obsoletas),
  # pero es modificable:
  #sieve_extensions = +imapflags -otraextension

  #http://www.earth.ox.ac.uk/~steve/sieve/procmail2sieve.pl
#}
####endfx
```

/etc/dovecot/dovecot-ldap.conf

```
vim /etc/dovecot/dovecot-ldap.conf
```

```
...
 ####fx:
 #http://wiki.dovecot.org/AuthDatabase/LDAP/Userdb
 #http://wiki.dovecot.org/AuthDatabase/LDAP
 #Comprobe' que au'n no es capaz de usar los DNS RR SRV para ldap
 #cuando no pongo uris:
 #   "dovecot: auth(default): Fatal: LDAP: No uris or hosts set"
uris = ldap://dklab2.casafx.dyndns.org ldap://dklab1.casafx.dyndns.org

#dn = uid=dovecot,dc=example,dc=com
#dnpass = dovecotpass
ldap_version = 3
base = ou=users,ou=accounts,dc=%Dd
# %d es casafx.dyndns.org y %Dd es casafx,dc=dyndns,dc=org
scope = subtree
user_filter = \
(&(objectClass=qmailUser)(objectClass=posixAccount)(uid=%n))

user_attrs = =uid=vmail, =gid=vmail, \
=mail=maildir:/afs/%Ld/service/mail/%1n/%2n/%n/Maildir,\
mailQuotaSize=quota_rule=*:storage=%$,mailQuotaCount=quota_rule=*:\
messages=%$,homeDirectory=home

#pass_attrs = ...
#pass_filter = ...
####endfx
```

Creación de volúmenes, directorios y permisos para el mail storage en AFS (ya hecho)

Ya se hizo desde dklab1:

- Volumen para servicio mail. Volumen para mailbox públicas. Volumen sieve global.
- Directorios .PublicMailboxIndexes y Maildir en los home; permisos.
- AFS-cron backups.

Sieve, adicionalmente necesita (ya hecho)

Ya se hizo desde dklab1.

```
ls /afs/casafx.dyndns.org/service/mail/sieve/
```

Quota, adicionalmente necesita

```
vim /usr/local/bin/quota-warning.sh
```

```
#!/bin/sh

PERCENT=$1
FROM="do.not.reply@‘echo $USER|awk -F@ ’{print $NF}’‘"

cat << EOF | /usr/sbin/sendmail -f $FROM "$USER"
From: ${FROM}
To: ${USER}
Subject: Your email quota is ${PERCENT}% full
Content-Type: text/plain; charset=’UTF-8’


 This message is automatically created
 by mail delivery software.

 The size of your mailbox has exceeded
 a warning threshold that is
 set by the system administrator.
 You *must* delete mails or empty some folders
 or you may loose emails in the future.

EOF
exit 0
```

```
chmod +x /usr/local/bin/quota-warning.sh
```

Public mailboxes, funcionamiento mínimo (ya hecho)

Nada que hacer. Revísese este apartado cuando se expuso dklab1. Lo fundamental fue:

- El maildirmake es importante: dovecot (proceso imap bajo afs id mailgroup) sólo

puede leer y listar, no tiene permisos para escribir allí; así, hemos de precrear, desde una cuenta de administrador, las mailboxes maildir del namespace Public/:

```
maildirmake.dovecot /afs/casafx.dyndns.org/service/mail/Public/.newsletter
```

- No tenemos un mecanismo definido para crear contenido en esas mailboxes, por ello se expuso como ejemplo el uso desde la línea de comandos a mutt para almacenar un correo desde la línea de comandos.

Tests

Comprobaciones parciales de distitos subsistemas; en la sección sobre los clientes se hacía una prueba completa.

Fichero de configuración Comprobaremos la corrección sintáctica y cómo han quedado reflejados nuestros cambios:

```
su vmail -m -c 'head -n 1 /etc/dovecot/dovecot.conf'


dovecot -a   # check syntax, y muestra el estado de todas las opciones
dovecot -n   # muestra opciones no por defecto


invoke-rc.d dovecot start


ps auxw|egrep '(dovecot|imap|managesieve)'
```

Probando si deliver (dovecot MDA) puede acceder a afs Para ello necesita un token AFS para imap.dklab1:

```
env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'pagsh'
```

En esta nueva shell, conseguimos token afs para imap.dklab2:

```
aklog
```

Ahora mandamos dos correos lanzando manualmente a deliver. Primero para aemu, después para umea. Al acabarm echaremos un vistazo a syslog para comprobar que se ha producido correctamente la entrega.

```
cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
                                      -d aemu@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing mda

 Testing deliver (dovecot MDA) on AFS. EOF
```

"To avoid looping, add -c dovecot-nowarning.conf (the same as dovecot.conf minus quota bits)".

Con correos señalados como correo no deseado (X-Spam-Flag: Yes):

```
cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
                                      -d aemu@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda + sieve filtering.
X-Spam-Flag: Yes

 Testing deliver (dovecot MDA) on AFS and with an spam tagged mail
 in order to make global sieve script act, so that mail is stored
 in a different mailbox.
EOF
```

Repetimos para umea:

```
cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
                                      -d umea@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda

 Testing deliver (dovecot MDA) on AFS.
EOF



cat << EOF | /usr/lib/dovecot/deliver -f postmaster@casafx.dyndns.org \
                                      -d umea@casafx.dyndns.org
From: postmaster@casafx.dyndns.org
Subject: Testing lda + sieve filtering.
X-Spam-Flag: Yes

 Testing deliver (dovecot MDA) on AFS and with an spam tagged mail
 in order to make global sieve script act, so that mail is stored
 in a different mailbox.
EOF


exit
```

En syslog:

```
grep deliver /var/log/syslog
```

```
dovecot: deliver(aemu@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'INBOX'
dovecot: deliver(umea@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'INBOX'
dovecot: deliver(aemu@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'Spam'
dovecot: deliver(umea@casafx.dyndns.org): sieve: msgid=unspecified:
         stored mail into mailbox 'Spam'
```

Comportamiento parecido tendrían los programas imap o managesieve para acceder a AFS.

SASL-GSSAPI para IMAP con dovecot-auth e imtest Utilizaremos el cliente imap "imtest"del paquete cyrus-clients-2.2:

```
apt-get install cyrus-clients-2.2
login umea
```

```
klist
imtest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
       -f /dev/null dklab2.casafx.dyndns.org
klist

#... an~adiendo -t "" realizari'a la autenticacio'n sobre TLS.

exit
```

Si cambiamos "-f /dev/null" por "-f /dev/stdin", podremos desarrollar un diálogo IMAP, consúltese el despliegue en dklab1.

SASL-GSSAPI para MANAGESIEVE con dovecot-auth y sivtest (bug) La herramienta sivtest también está incluída en cyrus-clients-2.2:

```
apt-get install cyrus-clients-2.2


login umea
```

```
klist


sivtest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
        -f /dev/null dklab1.casafx.dyndns.org
```

La salida, que inesperadamente termina en un error, es algo así:

```
C: AUTHENTICATE "GSSAPI" {836+}
   ...
S: ...


<queda bloqueado algunos minutos>


S: BYE "Disconnected for inactivity."
base64 decoding error -> supongo que porque no le ha llegado nada
Authentication failed. generic failure
Security strength factor: 0
Connection closed.
```

Añadiendo -t "" realiza la autenticación sobre TLS, pero el problema persiste. El error era de sivtest; no obstante, la prueba con el cliente de usuario sieve-connect demostrará que el servicio funciona.

```
klist


exit
```

SASL-GSSAPI para SMTP-AUTH con dovecot-auth y smtptest Ésta es la primera vez que comprobamos las capacidades SASL-GSSAPI de exim4 a través de su interfaz con dovecot-auth. Postpusimos este paso hasta ahora, cuando dovecot-auth está

configurado.

```
login umea
```

```
klist
smtptest -v -m GSSAPI -u umea -r CASAFX.DYNDNS.ORG \
         -f /dev/null dklab2.casafx.dyndns.org
```

```
...
S: 235 Authentication succeeded
Authenticated.
...
```

Añadiendo -t "" realizara'la autenticacioñ SASL-GSSAPI pero sobre una capa TLS. En cualquier caso, si todo fue bien salimos:

```
klist


exit
```

MDA opcional con Procmail

La directiva require_files del router procmail debía ser transformada:

```
vim /etc/exim4/exim4.conf.template
```

```
...
 procmail:
 ...
  transport = procmail_pipe
 ####fx:
 #-# emulate OR with "if exists"-expansion
 #-require_files = ${local_part}:\
 #-                ${if exists{/etc/procmailrc}\
 #-                  {/etc/procmailrc}{${home}/.procmailrc}}:\
 #-                +/usr/bin/procmail
  require_files = ${local_part}:${home}/.procmailrc:+/usr/bin/procmail
 ####endfx
  no_verify
  no_expn
  ...
```

```
invoke-rc.d exim4 stop; sleep 5; invoke-rc.d exim4 start
```

Ya podíamos instalar procmail:

```
apt-get install procmail
```

... y editar /etc/procmailrc; procurábamos que fuese totalmente autoexplicativo:

```
vim /etc/procmailrc
```

```
####fx:
# Una regla de procmail tiene 3 elementos separados por cambio de li'nea:
# 1)
# ":0" indica el inicio d una receta, se puede sufijar con:
# '<opciones>', u '<opciones>:', u '<opciones>:<lockfile>' donde <opciones>
# corresponde a, entre otras:
#
# W,w -> Wait el estado de salida del filtro o prog; si no es exitoso, pasa
# i,r -> Ignora errores; Raw-mode, no se fija q el mensaje termine en \n
# c  -> Carbon copy, consigue que si era regla de entrega, haga de no-entrega
# f,w -> el programa posterior hace de filtro, w para que espere su salida
# ... man procmail.
#
# 2)
# "* <posix regular expression>" (opcional)
#
# 3)
# "action"
#  que puede ser: /path/file, la ruta del mbox o maildir/ donde almacenarlo
#                 | comando, al q se le pasa el mail, o variable=;
#                           si no se pone nada: stdout
#                 ! user@host.com, para reenviarlo

# Destino por defecto cuando ninguna regla ha casado con un correo:
DEFAULT=/dev/null
```

```
# Vamos ya conla regla que almacena en el "home" del usuario;
# recordemos que exim4 llama a procmail como usuario y grupo vmail
# y, puesto que exim4 es ejecutado en su PAG gracias a k5start, no
# es necesario conseguir aqui' el tokens afs, de forma que lo
# siguiente seri'a posible, pero es innecesario:
#:0 Wic
#| /usr/bin/kinit -k \
#  -t /etc/keytab.d/dovecot.keytab imap/dklab1.casafx.dyndns.org
#:0 Wic
#| /usr/bin/aklog
#:0 Wic
#| /usr/bin/kdestroy
#
#...adicionalmente, el grupo afs "mailgroup" tiene los permisos
#   necesarios para listar el $HOME del usuario tal como para escribir
#   en $HOME/Maildir/, establecido asi' en el apartado de
#   creacio'n de recursos de almacenamiento en AFS de dovecot.


# Nuestra (u'nica) regla procmail:
:0
* !^X-Spam-Flag: [Yy][Ee][Ss]
$HOME/Maildir/
```

```
#ANOTACIONES:
#
#-El forwardslash (si'mbolo "/") que sufija Maildir en la regla, le indica
# a procmail que es una mailbox de formato maildir (el usado en nuestro
# despliegue) y no, por ejemplo, mbox.
#-Puesto que: a) el mailbox por defecto es, acorde con la variable DEFAULT,
# /dev/null, b) la regla anterior no opera sobre correos que hayan sido
# marcados como spam, y c) la regla anterior es la u'ltima definida,
# el Spam no se almacena. Es asi' porque ya lo hace el sistema
# preferido de almacenamiento del correo (el gestionado por deliver).
# No obstante, si queremos conservarlo, precediendo a la regla anterior
# declarari'amos otra como:
# :0
# * ^X-Spam-Flag: [Yy][Ee][Ss]
# $HOME/Maildir/.Spam
#
```

```
#-...incluso a este nivel podri'amos hacer la llamada a los filtros
# (spamassassin, clamav) o implementar cuotas. Es decir, como se
# ha visto desvelado varias veces ya, los subservicios del sistema de correo
# se pueden implementar en varios niveles. En el caso del spam, no lo hicimos
# en tiempo SMTP (exim4) para evitar perder correo en falsos positivos,
# tampoco lo hacemos aqui' porque el spam lo consideramos so'lo en el
# sistema predeterminado de almacenamiento.
#
####endfx
```

```
ls -l /etc/procmailrc # legible por cualquiera, tal cual vmail.
```

Test Utilizaremos procmail y otra herramienta del proyecto (formail) para simular una entrega de correo. Previamente conseguiremos las credenciales necesarias (token AFS):

```
cp ~/mailtest/mail.txt /tmp


env KRB5CCNAME=/tmp/.krb5cc_800_dovecot su vmail -m -c 'pagsh'
```

En la nueva shell:

```
aklog
HOMEUMEA=`getent passwd umea | awk -F: '{print $6}'`
#... /afs/casafx.dyndns.org/user/u/um/umea
cat /tmp/mail.txt | /usr/bin/formail -q- -s procmail \
                    -tm HOME=${HOMEUMEA} VERBOSE=on /etc/procmailrc
# formail:  -q- no quiet
#           -s program
# procmail: -t fail softly
#           -m toma un rc file (y alguna variable) y actu'a como filtro
```

```
procmail: [18328] Sat Nov 12 02:27:12 2011
procmail: Rcfile: "/etc/procmailrc"
procmail: Assigning "MAILDIR=/afs/casafx.dyndns.org/user/u/um/umea"
procmail: Assigning "DEFAULT=/dev/null"
procmail: Match on ! "^X-Spam-Flag: [Yy][Ee][Ss]"
procmail: Assigning \
"LASTFOLDER=/afs/casafx.dyndns.org/user/u/um/umea/Maildir/new/\
 1321061232.18328_0.dklab2"
From aemu@casafx.dyndns.org Sat Nov 12 02:27:11 2011
 Subject: hello
 Folder: \
 /afs/casafx.dyndns.org/user/u/um/umea/Maildir/new/1321061232      141
```

```
cat /tmp/mail.txt | /usr/bin/formail -i "X-Spam-Flag: YES" -q- \
              -s procmail -tm HOME=${HOMEUMEA} VERBOSE=on /etc/procmailrc
```

...ahora el correo tiene el flag para spam, y va a /dev/null

```
...
Folder: /dev/null
```

```
exit
```

1.4. Clientes (MUA/MRA/MSA...) con Mutt y Mozilla Thunderbird

En primer lugar, nos aseguramos de que los servicios puedan ser descubiertos mediante DNS:

```
dig @dklab1.casafx.dyndns.org. _submission._tcp.casafx.dyndns.org SRV +short
dig @dklab1.casafx.dyndns.org. _imap._tcp.casafx.dyndns.org SRV +short
dig @dklab1.casafx.dyndns.org. _sieve._tcp.casafx.dyndns.org SRV +short
```

En lo que sigue, si es necesario explicitar alguna de las máquinas se dirá dklab1 y, de forma parecida, si hay que suponer que estamos probando los clientes en alguna de las máquinas, se supondrá dklab1. Es indiferente, los resultados serán los mismos que si se utiliza dklab2.

1.4.1. Mutt y Sieve-connect (CLI)

```
apt-get install mutt-patched urlview lbdb libnet-ldap-perl sieve-connect \
               libgssapi-perl
```

mutt-patched cliente IMAP/SMTP mutt con funcionalidades adicionales (sidebar...)

lbdb interfaz para búsquedas en ldap u otros

sieve-connect cliente MANAGESIEVE

Desafortunadamente, mutt aún no soporta el descubrimiento del servicio LMTP de envío de correos a través del DNS SRV RR "_submission._tcp". Ésto implica que, por el momento, a lo máximo que podemos aspirar es a configurar aleatoria y automáticamente uno de los servidores exim4 cada vez que lancemos mutt, pero no podemos esperar que mutt cambie de uno a otro ante problemas durante una misma ejecución.

Mutt permite dos tipos de usos:

- Uso interactivo de mutt: "q" salir, los cursores para seleccionar en la lista, "RET" tanto para ver un correo seleccionado en la lista como para paginarlo después, "?" ayuda. Ésto debiera ser suficiente.
- Uso no interactivo de mutt: la página del manual describe las opciones que soporta en línea de comandos, la más importante es -e (eval), que permite cambiar cualquier variable de configuración (en las siguientes pruebas se hará uso de "-e", pero tras ellas mostraremos cómo crear una configuración que evite explicitarlas en cada llamada).

Para nuestro propósito, lo fundamental de ambos modos es que mandaremos los correos de forma no interactiva (pasando flags a mutt), mientras que para saber si el correo enviado se entregó correctamente, inspeccionaremos las mailboxes usando el modo interactivo de mutt.

Mutt soporta el modelo de seguridad openPGP (modelo distribuído Web-Of-Trust) con ayuda de GPG (gnu privacy guard, "gpg –version"). También soporta el sistema alternativo S/MIME (modelo de seguridad basado en PKI) con ayuda de openssl. Queda fuera del alcance de este documento exponer la configuración y uso de estos sistemas. Algunas referencias útiles podrían ser:

- http://pthree.org/2011/09/17/pgpmime-versus-smime/
- http://pthree.org/2011/09/15/setting-up-mutt-with-smime-and-pgpmime/
- Otras referencias alternativas:
  - http://mutt.blackfish.org.uk/cryptography/
  - http://www.imc.org/smime-pgpmime.html
  - http://wiki.mutt.org/?MuttGuide/UseGPG
  - https://kb.wisc.edu/middleware/page.php?id=4091

Otras anotaciones sobre mutt:

- Como se vio en el apartado de pruebas para exim4, mandar un correo podía generar otros tantos con sus correspondientes entregas. Por ejemplo:

```
exim4 -f aemu@casafx.dyndns.org -bt umea@casafx.dyndns.org

# -v -d+all para traza completa
```

... revelaba dos entregas para umea (una con deliver, otra con procmail) y una tercera para aemu (reenvío de todo lo que llegue a umea; con deliver).

... además, a posteriri, en /var/log/exim/mainlog y /var/log/syslog, exim4 y dovecot respectivamente registrarán sus pasos.

Por tanto el modo -bt de exim4 y los logs nos deberían aclarar dónde debemos buscar los correos entregados ante las pruebas que vendrán a continuación, en caso de duda.

- Mutt (enlazado con gnutls y no con openssl) tiene algún problema para verificar el certificado que le presenta el servidor, de forma que nos pide aceptarlo y almacenarlo en su ~/.mutt_certificates. Podemos evitar ésto explicitándole la localización del certificado de nuestra CA con el flag: -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt'

Mail de aemu a umea usando LMTP; acceso IMAP

Lo enviaremos, como se expuso, de forma no interactiva, es decir en lugar de ejecutar mutt y pulsar "m", le pasaremos como flags todos los parámetros necesarios y el cuerpo del mensaje como pipe a su descriptor "stdin". Puede que mutt nos pregunte por la creación de un directorio local, es seguro pulsar "n" para indicarle que no lo haga.

```
login aemu
```

```
klist; tokens
echo "Cuerpo del correo: hola de aemu a umea" \
  | mutt -d 5 \
  -e "set smtp_url='smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
  -e "set smtp_authenticators='gssapi'"\
  -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
  -e "my_hdr From: aemu@casafx.dyndns.org" \
  -s "aemu2umea" -- umea@casafx.dyndns.org

less .muttdebug0  # gracias a  -d 5  mutt dejo' un fichero de log
                  # con lo que envi'a y recibe del servidor etc
```

- El primer flag "-e" (eval) establece manualmente dónde está el servidor LMTP y el método de autentificación preferido.
- El segundo establece el valor de la cabecera "From:"

- El flag "-s" establece el asunto del correo. Le sigue la dirección de destino.
- Si hubiésemos tenido que adjuntar un archivo, usaríamos “-a <path>”tantas veces como hiciese falta.

Puesto que umea tiene configurado hacer reenvío ("forward") de sus correos a aemu, si aemu accede a su cuenta IMAP, ya debiera estar allí su copia:

```
mutt -d 5 -e 'set imap_check_subscribed' \
          -e 'set imap_list_subscribed = no' \
          -e "set sort=reverse-threads" \
          -e\
           'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
          -f imap://aemu@dklab1.casafx.dyndns.org


less .muttdebug0 # gracias a -d 5  mutt deja de nuevo su log,
                 # rotando adema's al anterior al renombralo
                 # .muttdebug1 etc.
```

- El flag “-f”permite especificar el buzón que se desea abrir. Se ha usado como argumento la URL para imap tal como se define en el rfc http://tools.ietf.org/html/rfc5092.
- mutt hace LSUB de las mailboxes (y en el lateral, o dándole a "y", aparecerán automáticamente a las que estés subscrito). La variable responsable de ese importante comportamiento es imap_check_subscribed: http://mutt-ng.berlios.de/manual/imap-check-subscribed.html.
- Por su lado la variable imap_list_subscribed hace que al pulsar "y" de forma que mutt entre en modo navegador de buzones, aparezcan buzones a los que aún no estamos subscritos, teniendo la oportunidad de hacerlo con "s" ("u" para desusbscribirnos). "T" conmuta este comportamiento. Así pues ésta es la forma de utilizar las subscripciones en mutt, sin embargo, no hemos sido capaces de acertar a utilizarlo (que aparezcan listados buzones aún sin subscripción), la documentación es escueta al respecto: http://mutt-ng.berlios.de/manual/imap-list-subscribed.html.
- Está sugerida la posibilidad de mostrar en la barra de estado inferior de mutt el uso de la quota. Desafortunadamente ésto no ha sido implementado aún, luego no podemos inspeccionar el uso de quota desde mutt. http://dev.mutt.org/trac/ticket/2743#comment:1

```
exit
```

Vamos a ver lo que pasó en la cuenta IMAP del destinatario, umea:

```
login umea
```

```
mutt -e 'set imap_check_subscribed' -e 'set imap_list_subscribed = no' \
     -e "set sort=reverse-threads" \
     -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
     -f imap://umea@dklab1.casafx.dyndns.org
```

Además, umea tiene activado el estilo opcional de entrega de correo, dejando que procmail lo almacene en el directorio ~/Maildir de su Home:

```
mutt -f ~/Maildir
exit
```

Como administrador (es decir, en posesión del token para el usuario afs "root.admin"), podemos acceder a todos esos mailboxes en AFS directamente, sin intermediación de IMAP. Es la forma más rápida y adecuada para estas pruebas del sistema.

```
mutt -R -f /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/
mutt -R -f /afs/casafx.dyndns.org/service/mail/u/um/umea/Maildir/
mutt -R -f /afs/casafx.dyndns.org/user/u/um/umea/Maildir/
#... el flag "-R" se usa para acceso read-only, al ser cuentas ajenas.
```

Nota sobre la cabecera "From:" en pruebas posteriores, el envío de un correo generará un correo hacia el remitente (porque trató de enviar un correo infectado, porque superó el límite máximo de tamaño, o cualquier otra razón). Dejar constancia de la dirección del remitente es, en general, el cometido de la cabecera "From:". Debemos asegurarnos de que esté correctamente establecida.

En programas como Mutt, si no se le dice explícitamente, la creará utilizando el nombre asociado al UID del proceso y el FQDN del host, lo cual puede ser erróneo dando lugar a comportamientos aparentemente erráticos: en este caso se formaría aemu@dklab1.casafx.dyndns.org, que puede hacer match en los routers locales de exim4 y producirse una entrega típica (local) de procmail, por la cual se almacenaría un correo en /var/mail/aemu (y si tuviese un ~/.procmailrc de contenido similar a nuestro /etc/procmailrc, se almacenaría en ~/Maildir a pesar de que aemu no tiene activado el sistema de entrega opcional en, precisamente, ~/Maildir).

Nota sobre las cuentas "postmaster": en ocasiones (porque, por ejemplo, haya sido imposible decidir cómo entregar un correo) el sistema genera un correo para la cuenta

postmaster, sin embargo éste no llega a las cuentas de umea y aemu (tal como explicitan los aliases en ldap). Ésto no debe confundirnos: efectivamente postmaster@casafx.dyndns.org es un alias a umea y aemu, pero cuando exim4 genera un correo a postmaster, lo hace a postmaster@dklab1.casafx.dyndns.org (si en dklab1, en general postmaster@<FQDN>), que no es una cuenta centralizada sino local, y por tanto los aliases que se aplican son los de su "/etc/aliases" local, que hace corresponder postmaster con la cuenta local "root" y éste se hace corresponder con "user", luego todo mail a postmaster generado localmente acaba en /var/mail/user.

Mail de aemu a umea pero con marca de spam

```
login aemu
echo 'XJS*C4JDBQADN1.NSBN3*2IDNEN*GTUBE-STANDARD-ANTI-UBE-TEST-EMAIL*C.34X'\
  | mutt -e "set smtp_url = 'smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
         -e "set smtp_authenticators = 'gssapi'" \
         -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
         -e "my_hdr From: aemu@casafx.dyndns.org" \
         -e "set sort=reverse-threads" \
         -s "spam" umea@casafx.dyndns.org
exit
```

```
mutt -R -f /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/.Spam
```

Mail de aemu a umea pero con marca de virus

```
login aemu
```

```
echo 'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'\
  | mutt -e "set smtp_url = 'smtp://aemu@dklab1.casafx.dyndns.org:587/" \
         -e "set smtp_authenticators = 'gssapi'" \
         -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
         -e "my_hdr From: aemu@casafx.dyndns.org" \
         -e "set sort=reverse-threads" \
         -s "virus" umea@casafx.dyndns.org
```

```
...
SMTP session failed: 550 This message was detected as possible
                     malware (Eicar-Test-Signature).
```

```
exit
```

...al ser un filtro en tiempo SMTP, no llega un correo respuesta a la cuenta del remitente, como sí ocurriría al superar el límite máximo de tamaño u otros.

Mail de aemu a inexistente cuenta centralizada

```
login aemu
```

```
echo "no existe"\
  | mutt \
    -e "set smtp_url = 'smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
    -e "set smtp_authenticators = 'gssapi'" \
    -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
    -e "my_hdr From: aemu@casafx.dyndns.org" \
    -e "set sort=reverse-threads" \
    -s "inexistente" noexiste@casafx.dyndns.org
exit
```

Si abrimos la INBOX de aemu, aparecerá el correo respuesta con asunto "Mail delivery

failed: returning message ..."

```
mutt -R -f /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/
```

Resolución de aliases (en nuestro caso también mostraba la detección de duplicados)

```
login aemu
```

```
echo "hola postmaster" \
  | mutt -e "set smtp_url='smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
         -e "set smtp_authenticators='gssapi'" \
         -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
         -e "my_hdr From: aemu@casafx.dyndns.org" \
         -e "set sort=reverse-threads" \
         -s "postmaster2aemu+umea" postmaster@casafx.dyndns.org
exit
```

```
mutt -R -f /afs/casafx.dyndns.org/service/mail/u/um/umea/Maildir/
mutt -R -f /afs/casafx.dyndns.org/user/u/um/umea/Maildir/
mutt -R -f /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/
```

Vacation reply

```
cat <<EOF | ldapmodify -Q
dn: uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
replace: deliveryMode
deliveryMode: reply
EOF

slapcat -s uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org \
        | grep deliveryMode

login aemu
```

```
echo "Si modo de entrega reply en destinatario,
      rebota un correo de aviso al remitente." \
  | mutt -e "set smtp_url = 'smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
         -e "set smtp_authenticators = 'gssapi'" \
         -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
         -e "my_hdr From: aemu@casafx.dyndns.org" \
         -s "\T1\textquestiondown disponible?" umea@casafx.dyndns.org
exit
```

```
mutt -R -f /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/
mutt -R -f /afs/casafx.dyndns.org/service/mail/u/um/umea/Maildir/
mutt -R -f /afs/casafx.dyndns.org/user/u/um/umea/Maildir/

cat <<EOF | ldapmodify -Q
dn: uid=umea,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
delete: deliveryMode
deliveryMode: reply
EOF
```

Prueba del sistema de quotas de Dovecot deliver

Test sistema de quotas. El sistema de quotas, según se declaró en el dovecot.conf, es del tipo maildir ([[http://www.inter7.com/courierimap/README.maildirquota.html][ref]]). Dovecot registra el estado de uso de las mailboxes en un fichero especial (que procura comprobar correcto y mantener actualizado):

```
cat /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/maildirsize
```

La primera fila es, en bytes, la cuota que dovecot resolvió en ldap para este usuario (la S es de "Size"). La quota en ldap está en Kbytes porque en dovecot-ldap se recoge como metadata en formato "storage" (debe corresponder con KB según la documentación); dovecot lo multiplica por 1024 y éso es, pues, lo que vemos en "maildirsize".

La suma del resto de filas es el uso total de estas mailboxes: en concreto si sumamos primeras columnas tenemos el uso en bytes, si sumamos segundas columnas tenemos el uso en número de mensajes (no tenemos cuota -es cero- que actúe por el número de mensajes, luego ese caso no nos interesa).

Para comprobar el aviso ante paso por 90 %, podemos calcular qué cuota debería tener el usuario para que mandando un sólo y ligero email rebasásemos el límite del 90 %. Para ello vamos a calcular ese valor de quota sabiendo el uso actual, y vamos a calcular un tamaño seguro para el correo de prueba. Después actualizamos la quota en ldap y mandamos un autocorreo bajo la cuenta de aemu. Tras unos segundos, comprobamos la cuenta IMAP para ver si ha llegado el aviso de paso por 90 %.

```
OLD_QUOTA=‘(slapcat -s
            uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
            | grep mailQuotaSize | awk ’{printf("%s",$2)}’;
            echo ’*1024’)|bc‘

CURRENT_SIZE=‘cat
     /afs/casafx.dyndns.org/service/mail/a/ae/aemu/Maildir/maildirsize
     | sed 1d |  awk ’{ SUM += $1} END { print SUM }’‘

NEW_QUOTA_PLUS_1KB=‘echo "scale=4; ((1/0.9)*${CURRENT_SIZE}) + 1024"
                   | bc | awk -F. ’{print $1}’‘

THRESHOLD_QUOTA_PLUS_1KB=‘echo "scale=4;
                         ${CURRENT_SIZE}/${NEW_QUOTA_PLUS_1KB}"|bc‘

 #... THRESHOLD para el valor de nueva quota ma’s 1KB, ya no es 0.9 sino algo
 #menor, debido al 1KB en que hemos incrementado el valor de la quota. Entonces,
 #si mandamos un correo que haga pasar de ese algo menor de 0.9 a ma’s de 0.9,
 #estaremos disparando el mensaje de alerta; el taman˜o de correo que consigue
 #e’so seri’a:

TRIGGER_SIZE=‘echo "scale=4; (( 0.9 - ${THRESHOLD_QUOTA_PLUS_1KB} )
                    *${CURRENT_SIZE} )"|bc|awk -F\. ’{print $1}’‘

echo ${CURRENT_SIZE}, ${OLD_QUOTA}, ${NEW_QUOTA_PLUS_1KB},\
     ${THRESHOLD_QUOTA_PLUS_1KB}, ${TRIGGER_SIZE} \(en bytes\).
```

```
 # Introducimos la nueva quota escalada y redondeada a KB (e'sto u'ltimo
 # hace que la quota sea igual a la calculada o menor en, a lo sumo,
 # 999 Bytes.
cat <<EOF | ldapmodify -Q
dn: uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
replace: mailQuotaSize
mailQuotaSize: `echo ${NEW_QUOTA_PLUS_1KB}/1024|bc`
EOF

slapcat -s uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org \
        | grep mailQuotaSize




login aemu TRIGGER_SIZE=${TRIGGER_SIZE}
```

```
yes o | tr \\n ' ' \
  | dd bs=1 count=`echo ${TRIGGER_SIZE}+10|bc` 2>/dev/null \
  | mutt -e "set smtp_url = 'smtp://aemu@dklab1.casafx.dyndns.org:587/" \
         -e "set smtp_authenticators = 'gssapi'" \
         -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
         -e "my_hdr From: aemu@casafx.dyndns.org" \
         -e "set sort=reverse-threads" \
         -s "testing quota" aemu@casafx.dyndns.org

stat /usr/local/bin/quota-warning.sh
```

... el campo "Access time'én la salida de stat debería ser el inmediato pasado (apróx. la salida de date –rfc-3339=seconds), puesto que el script se acaba de ejecutar por el sistema

de cuotas.

Además, si accedemos a la mailbox veremos el correo de aviso:

```
mutt -e 'set imap_check_subscribed; set imap_list_subscribed = no' \
     -e "set sort=reverse-threads" \
     -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
     -f imap://aemu@dklab1.casafx.dyndns.org


exit
```

Si todo fue bien, reestablecemos el valor inicial:

```
cat <<EOF | ldapmodify -Q
dn: uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
replace: mailQuotaSize
mailQuotaSize: `grep ^mailQuotaSize ~/ldif/aemu+mail_attributes.ldif
                | awk '{print $2}'`
EOF


slapcat -s uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org \
        | grep mailQuotaSize
```

Prueba de la ACL sobre el límite máximo de tamaño por email

Nota: sería conveniente entender que es imposible hacer este test con un automail.

```
OLD_MAXSIZE=‘grep ^mailSizeMax ~/ldif/aemu+mail_attributes.ldif
             | awk ’{print $2}’‘
TESTS_MAXSIZE=500
echo ${OLD_MAXSIZE}, ${TESTS_MAXSIZE}

cat <<EOF | ldapmodify -Q
dn: uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
replace: mailSizeMax
mailSizeMax: ${TESTS_MAXSIZE}
EOF

slapcat -s uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org \
        | grep mailSizeMax | awk ’{print $2}’



login umea TESTS_MAXSIZE=${TESTS_MAXSIZE}
```

```
yes . | tr \\n ’ ’ \
      | dd bs=1 count=‘echo ${TESTS_MAXSIZE}+10|bc‘ 2>/dev/null \
  | mutt \
    -e "set smtp_url = ’smtp://umea@dklab1.casafx.dyndns.org:587/’" \
    -e "set smtp_authenticators = ’gssapi’" \
    -e ’set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt’ \
    -e "my_hdr From: umea@casafx.dyndns.org" \
    -s "sizeMax" aemu@casafx.dyndns.org
```

Si umea accede a su cuenta a través de imap o (puesto que tiene activada entrega opcional con procmail en su home) o ~/Maildir:

```
mutt -e 'set imap_check_subscribed; set imap_list_subscribed' \
     -e "set sort=reverse-threads" \
     -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
     -f imap://umea@dklab1.casafx.dyndns.org


mutt -f ~/Maildir


exit
```

Tuvimos cuidado con que el From fuese el correcto, como siempre que se espera recibir un aviso. También, en /var/log/exim4/mainlog aparecería algo como:

```
tail /var/log/exim4/mainlog
```

```
1RR57D-0001So-QO ** aemu@casafx.dyndns.org R=ldap_maxsize: Your
                    message is too big. \nYour message was rejected
                    because the user aemu@casafx.dyndns.org\ndoes
                    not accept message larger than 500.
```

El test anterior tiene como consecuencia que una de las entregas de correo que se ponen en juego no pude llevarse a cabo: al ser devuelto el correo a umea, se aplican todos los routers y con ello el de los reenvíos que la cuenta umea tiene configurados, en concreto interesa el que decidía hacer reenvío a aemu y que, a sabiendas ya de que no llegará por el límite de tamaño, resulta imposible de entregar. Éso hace que el sistema de correo avise a la cuenta postmaster@<FQDN>, que por el sistema de aliases local /etc/aliases acaba en la cuenta del usuario local user:

```
mutt -R -f /var/mail/user # se usa el router local_user
```

Esta situación puede servir para ilustrar el uso de la cola en exim4. El programa "mailq" ejecuta exim4 en un modo especial para mostrarnos si hay algún correo en cola y su estado (en el caso que nos ocupa, el estado es "frozen" por no encontrarse ruta válida para él).

```
mailq
```

Para gestionar la cola podíamos usar el identificador de correo que muestra mailq y ciertas opciones de exim4, como se mencionó, pero era más sencillo instalar algún

programa como pfqueue (en línea de comandos, cómoda interfaz interactiva). Pulsando "d" podíamos borrar el correo de la cola.

```
pfqueue
```

Si todo lo anterior fue bien, podemos ya restaurar el valor de tamaño máximo por correo para la cuenta de aemu:

```
cat <<EOF | ldapmodify -Q
dn: uid=aemu,ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
changetype: modify
replace: mailSizeMax
mailSizeMax: ${OLD_MAXSIZE}
EOF
```

Nota: si el límite está en torno a 1000 Bytes, un correo que ha sido transformado en un aviso detallado de su condición de spam, superaría ese límite y por tanto no llegaría nunca al destinatario (como spam), sino que llegaría al remitente (como correo de aviso de que ha superado el límite de tamaño). Por tanto, debiera ser tomado como mínimo un valor superior a 1000 para el tamaño máximo por correo.

Creación de una configuración de usuario para mutt en ~/.muttrc

```
login aemu
```

```
cat <<EOF > ~/.muttrc
# Do'nde llega el correo:
set spoolfile = imap://aemu@\`dig @dklab1.casafx.dyndns.org. _imap._tcp.casafx.dyn
set folder = ~/Maildir
set mbox_type = Maildir

# Buzones relacionados con la edicio'n de correos: borradores, copias...
# no'tese que si disponemos estos buzones en imap y no localmente,
# podra'n ser compartidos por todos los clientes, siendo e'sto deseable:
set copy = yes
set record = !Sent      # imap://aemu@dklabX.casafx.dyndns.org/Sent
set postponed = !Drafts # imap://aemu@dklabX.casafx.dyndns.org/Drafts
set trash = !Trash      # imap://aemu@dklabX.casafx.dyndns.org/Trash
```

```
# Los buzones sobre los que queremos navegar sera'n los anteriores ma's
# subscritos, de esta forma se an~aden/suprimen dina'micamente, sin
# declararlo aqui', todos los buzones a los que estemos subscritos
# (comando LSUB de IMAP).
# '!' equivale al spoolfile
# '=' equivale al folder
mailboxes ! !Sent !Drafts !Trash =
set imap_check_subscribed = yes

set imap_list_subscribed  = no
#... e'sta permite que al pulsar la "y" y entrar en modo mailboxes browser,
#    liste todas las mailboxes incluso no subscritas, asi' que puedas
#    subscribirte con "s" ("u" para unsubscribe). "T" llama a toggle-subscribed.
#    nota: no hemos acertado a utilizarlo, pero asi' esta' descrito en el manual.
```

```
# Smarthost LMTP
set smtp_url = smtp://aemu@\‘
dig @dklab1.casafx.dyndns.org. _submission._tcp.casafx.dyndns.org SRV +short
 | awk ’{print \$4}’\‘:587

# Para’metros de autentificacio’n y encriptacio’n smtp/imap
set smtp_authenticators=gssapi
set imap_authenticators=gssapi
set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt

# Composicio’n de correos:
set editor="/usr/bin/vim +8"
set edit_headers

# Declara contenido de la cabecera "From:" (y otras) en los correos enviados:
my_hdr From: aemu@casafx.dyndns.org
my_hdr Jabber-ID: aemu@casafx.dyndns.org
#my_hdr X-PGP-Key: http://casafx.dyndns.org/~aemu/pubkey.asc

# Correo recibido, criterio de ordenacio’n:
set sort=reverse-threads
```

```
# Tratamiento de adjuntos:
# "v" permite listarlos y, entonces, "s" permite guardarlos separadamente.
# Pero adema's si un correo lleva adjuntos de ciertos tipos MIME, podemos
# hacer que se procesen como texto simple para poder ser enviados al terminal
# y visualizarlos nada ma's abrir el correo en cuestio'n. Los programas
# responsables de ese procesamiento son los del sistema MIME/mailcap
# en /etc/mailcap. Recomendamos modificar el comportamiento para html asi':
# echo 'text/html; /usr/bin/w3m -I %{charset} -o display_link_number=1'   \
#      '-T text/html -cols 68 '%s'; copiousoutput; description=HTML Text;'\
#      'nametemplate=%s.html' > ~/.mailcap
# Si esta' instalado url-view, Control-b debiera presentarnos un menu'
# con todas las url presentes y poder lanzar un navegador sin necesidad
# de copiar+pegar.
auto_view text/html
auto_view application/pdf
```

```
# Algunos atajos de teclados u'tiles, ba'sicamente para que n,p,i nos
# permitan movernos y seleccionar por el panel lateral de mailboxes,
# y k,j nos permitan movernos sobre la lista de correos. Por su lado,
# 'b' conmuta el panel lateral para hacerlo visible o no.
bind index k previous-entry
bind index j next-entry
bind index p sidebar-prev
bind index n sidebar-next
bind index i sidebar-open
bind pager p sidebar-prev
bind pager n sidebar-next
bind pager i sidebar-open
macro pager b '<enter-command>toogle sidebar_visible<enter><redraw-screen>'

# Las posibilidades de configuracio'n de mutt son ingentes, puede consultarse:
# man muttrc
EOF

exit
```

Prueba del servicio MANAGESIEVE

Mutt no soporta el protocolo MANAGESIEVE. Para ello se utiliza un programa adicional: sieve-connect, un cliente interactivo/guionizable de MANAGESIEVE escrito en perl. Cuando le pedimos transferir un script sieve utilizando MANAGESIEVE, sive-connect comprueba adicionalmente si el script es correcto y, en caso contrario, aborta la operación informándonos del error exacto.

A veces nos interesa realizar esa verificación localmente, sin necesidad de hacer intentos de transferencia al servidor. Mutt no tiene tampoco soporte para la depuración de scripts sieve, pero podríamos utilizar sieve-test (del paquete dovecot-common) como compilador/depurardor de scripts sieve local.

Ejemplo de (inocuo) de script sieve:

```
cat <<EOF > ~/mailtest/sievetest
       if header :contains "subject" "hello" {
       }
EOF
cp ~/mailtest/sievetest /tmp
```

Opcional comprobación local:

```
sieve-test -t -c /tmp/sievetest mailtest/mail.txt
```

... si "final result: success", el script es correcto y pasamos a transferirlo usando a sieve-connect y su modo interactivo.

Nota: cuando desplegamos dovecot, hicimos una prueba de login SASL-GSSAPI para MANAGESIEVE que falló debido a bugs en la herramienta sivtest; entonces aseguramos que el sistema funcionaba y que podría ser comprobado cuando probásemos las herramientas de usuario. Ésta es esa herramienta y, efectivamente, la autenticación se produce satisfactoriamente:

```
login aemu
```

```
sieve-connect --debug --debugsasl \
              -a aemu -u aemu -m GSSAPI -r CASAFX.DYNDNS.ORG \
              -server dklab1.casafx.dyndns.org

#             (-u aemu@casafx.dyndns.org ... tambie'n funcionari'a)
> upload /tmp/sievetest
> ls
> view sievetest
> activate sievetest
> quit
exit
```

La transferencia ha dado lugar a:

```
tree -a /afs/casafx.dyndns.org/service/mail/a/ae/aemu/sieve/
```

```
 +-sievetest.sieve   (lo que no lleva el sufijo ".sieve", no se tiene
 |                    en cuenta, de ahi' que se an~ada sistema'ticamente a
 |                    todos los scripts transferidos).
 ‘ tmp               (no cambia antes y despue's de activar -no compila
                      ahi'-)
```

La activación ha dado lugar a que aparezca el link .dovecot.sieve apuntando al script:

```
readlink /afs/casafx.dyndns.org/service/mail/a/ae/aemu/.dovecot.sieve
```

```
   sieve/sievetest.sieve
```

Todo es congruente con la configuración de localizaciones dada en la sección "plugin" del dovecot.conf.

Vamos a desactivarlo, por ejemplo utilizando el modo no interactivo de sieve-connect:

```
login aemu
```

```
sieve-connect --debug --debugsasl --notlsverify \
              -a aemu -u aemu -m GSSAPI -r CASAFX.DYNDNS.ORG \
              -server dklab1.casafx.dyndns.org \
              --deactivate "sievetest"
exit
```

```
readlink /afs/casafx.dyndns.org/service/mail/a/ae/aemu/.dovecot.sieve
```

```
   <nada>
```

...puesto que el enlace no existe ya.

Soporte de autocompletado de direcciones en ldap para mutt

Las claves son:

- Anunciar una utilidad externa que lanzará mutt para hacer las búsquedas: set query_command="<command_statement>"
- Cada vez que se quiera autocompletado, se lanzará la función query_command a través del atajo de teclado Control-T. Ésto iniciará la búsqueda con la cadena de texto insertada hasta ese momento en cualquier promt de mutt. Una lista de posibles completados aparecerá y podremos elegir.

Evidentemente <command_statement>podría ser un script de shell que, utilizando herramientas conocidas como ldapsearch, realice la búsqueda. Más adelante se dará un ejemplo de implementación de ese script, pues nos queremos centrar en un software específico: lbdb, http://www.spinnaker.de/lbdb/, que ya fue instalado. Lbdb es capaz de hacer búsquedas en muy diferentes fuentes, nosotros lo configuraremos globalmente para que use ldap si bien cualquier usuario puede copiar los ficheros de configuración desde /etc a su home (prefijándolos con ".") y hacer sus modificaciones.

```
cp -a /etc/lbdb.rc /etc/lbdb.rc_orig

cp -a /etc/lbdb_ldap.rc /etc/lbdb_ldap.rc_orig


vim /etc/lbdb_ldap.rc
```

```
####fx:
# LDAP_NICKNAME => ['LDAP_SERVER',
#                   'LDAP_SEARCH_BASE',
#                   'LDAP_SEARCH_FIELDS',
#                   'LDAP_EXPECTED_ANSWERS',
#                   'LDAP_RESULT_EMAIL',
#                   'LDAP_RESULT_REALNAME',
#                   'LDAP_RESULT_COMMENT',
#                   'IGNORANT' (optional),
#                   'LDAP_BIND_DN' (optional),
#                   'LDAP_BIND_PASSWORD (optional)],
# (IGNORANT is an optional argument. If you set it to 1, mutt_ldap_query
# uses wildcards *foo* for searching).
#
##
# Puesto que la direccio'n de correo puede encontrarse en el atributo "mail"
# o en "mailAlternateAddress" si es un alias, o formarse a partir del uid,
# creamos 3 perfiles de bu'squeda en ldap. Pero adema's, como lbdb no soporta
# DNS SRV RR, debemos duplicar los perfiles anteriores: 3 para dklab1 y
# otros 3 para dklab2.
```

```
%ldap_server_db = (
  'dklab1_uid_profile' => ['dklab1.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn uid', 'cn sn uid',
            '${uid}@casafx.dyndns.org', '${cn} ${sn}', '', 1 ],
  'dklab1_mail_profile' => ['dklab1.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn mail', 'cn sn mail',
            '${mail}', '${cn} ${sn}', '', 1 ],
  'dklab1_alias_profile' => ['dklab1.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn mailAlternateAddress', 'cn sn mailAlternateAddress',
            '${mailAlternateAddress}', '${cn} ${sn}', '', 1 ],
  'dklab2_uid_profile' => ['dklab2.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn uid', 'cn sn uid',
            '${uid}@casafx.dyndns.org', '${cn} ${sn}', '', 1 ],
  'dklab2_mail_profile' => ['dklab2.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn mail', 'cn sn mail',
            '${mail}', '${cn} ${sn}', '', 1 ],
  'dklab2_alias_profile' => ['dklab2.casafx.dyndns.org',
            'ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org',
            'sn cn mailAlternateAddress', 'cn sn mailAlternateAddress',
            '${mailAlternateAddress}', '${cn} ${sn}', '', 1 ],
);
####endfx
```

```
vim /etc/lbdb.rc
```

```
####fx:
METHODS=m_ldap
LDAP_NICKS="dklab1_uid_profile dklab1_mail_profile dklab1_alias_profile
            dklab2_uid_profile dklab2_mail_profile dklab2_alias_profile
            "
KEEP_DUPES=no
####endfx
```

```
cp -a /etc/lbdb.rc /etc/lbdb.rc_custom
cp -a /etc/lbdb_ldap.rc /etc/lbdb_ldap.rc_custom
```

Podemos hacer una prueba de nuestra configuración:

```
lbdbq "u"
```

Si todo ha ido bien, declaramos globalmente para mutt el mecanismo con:

```
echo "set query_command='lbdbq %s'" > /etc/Muttrc.d/local-00-query_command.rc
```

... como se indicó, Control-T lanzará el autocompletado en un prompt de mutt. Una buena oportunidad sería al pulsar "m" para redactar un nuevo correo, pues la interfaz interactiva de mutt abrirá un promt "To:" para que le indiquemos el destinatario.

Podemos llevar nuestra configuración a otras máquinas, por ejemplo a dklab2:

```
ssh dklab2.casafx.dyndns.org "apt-get install lbdb libnet-ldap-perl; \
                              echo set query_command=\'lbdbq %s\'
                              | cat >/etc/Muttrc.d/local-00-query_command.rc"
scp /etc/lbdb.rc /etc/lbdb_ldap.rc  dklab2.casafx.dyndns.org:/etc/
```

Script shell basado en ldapsearch Si sólo se quiere consultar en ldap, este script es más rápido que ldbd, y usa registros SRV pues ldapsearch lo hace (lo hace si su /etc/ldap/ldap.conf está configurado para ello, es decir no explicita parámetro "URI"

alguno).

El script es una adaptación (a nuestro despliegue) y mejora (aliases...) del publicado en: http://wiki.mutt.org/?QueryCommand/MuttLDAPsearchSH.

```
echo "set query_command='/usr/local/bin/address_ldap_lookup.sh'" \
     > /etc/Muttrc.d/local-00-query_command.rc
touch /usr/local/bin/address_ldap_lookup.sh
chmod a+x /usr/local/bin/address_ldap_lookup.sh
vim /usr/local/bin/address_ldap_lookup.sh
```

```
#!/bin/sh
# Requires: echo, expr, grep, ldapsearch, sed, test, export (eg. bash)
# Set your PATH accordingly.
#
(
max=100
timeout=60
sort=givenName

####fx:
base='ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org'
filter="(&(|(cn=*$@*)(displayName=*$@*)(givenName=*$@*)(sn=*$@*)(mail=*$@*)
(mailAlternateAddress=*$@*)))"
#-base='ou=Contacts,dc=example,dc=com'
#-filter="(&(objectClass=mozillaAbPersonAlpha)(|(cn=*$@*)(displayName=*$@*)(givenNa
####endfx

status=`ldapsearch -z $max -x -s one -l $timeout -b "$base" "$filter" 1.1 2>&1
 | grep -E '^(# )?(numResponses|numEntries|result|ldapsearch): '`
count="`echo $status | grep numResponses:
 | sed -e 's/.*numResponses: \([0-9.+-]*\).*/\1/'`"
count="`expr ${count:-1} - 1`"
result="`echo $status | grep result:
 | sed -e 's/.*result: \([0-9.+-]*\) \([^ ]*\).*/\1/'`"
result="`expr ${result:--7} + 0`"
rmesg="`echo $status | grep ldapsearch:`"
test -z "$rmesg" && \
rmesg="rcode=$result `echo $status | grep result: | \
sed -e 's/.*result: \([0-9.+-]*\) \([^#:]*\) [#]*[ ]*[A-Za-z]*[:]*.*/\2/'`"
```

```
if test "${result:-1}" -ne 0 -a "${result:-1}" -ne 4
then
  echo "ERROR: $rmesg"
  exit 2
fi

if test ${count:-0} -eq 0
then
  echo "Searching database ... $max entries ... $count matching."
  exit 1
fi

####fx:
#-(
#-ldapsearch -S $sort -LLL -z $max -x -s one -l $timeout -b "$base" -a search "$fil
#-mail displayName mozillaSecondEmail givenName initials surname ; \
#-echo dn: DONE \
#-)
(
ldapsearch -S $sort -LLL -z $max -x -s one -l $timeout \
           -b "$base" -a search "$filter"              \
            cn sn mail mailAlternateAddress ; \
echo dn: DONE \
) 2>/dev/null | while read ATTR VAL
####endfx
```

```
do
  if test -z "$ATTR"; then continue; fi
  attr="`echo $ATTR | sed -e 's/:$//'`"
  case "$attr" in
    dn)
        if test -z "$dn"
        then
          echo "Searching database ... $max entries ... $count matching:"
        else
####fx:
          if test -n "$uid";
           then echo "${uid}@casafx.dyndns.org\t $cn\t $sn"
          fi
          if test -n "$mail"
           then echo "$mail\t $cn\t $sn"
          fi
          if test -n "$mailAlternateAddress"
           then for i in $aliases $mailAlternateAddress;
                    do echo "${i}\t $cn\t $sn"
                done
           unset aliases
          fi
```

```
#-         echo -n "$mail         $displayName"
#-
#-         if test -n "$sn"
#-         then
#-           echo -n "   "
#-           ( ( test -n "$initials" && echo -n $initials ) \
#-           || (test -n "$givenName" && echo -n $givenName) )
#-           echo -n " $sn"
#-         fi
#-
#-         test -n "$mailAlternateAddress" \
#-         && echo -n " <$mailAlternateAddress>"
#-         test -n "$mozillaSecondEmail" && echo -n " <$mozillaSecondEmail>"
#-         echo
####endfx
        fi
        dn="$VAL"
        eval unset $VARS VARS
        ;;
    *)
####fx:
```

```
#-      test -n "$attr" -a -n "$VAL" && setvar $attr "$VAL" && VARS="$VARS $attr"
        if test $attr = mailAlternateAddress
           then aliases="$aliases $mailAlternateAddress"
        fi
        test -n "$attr" -a -n "$VAL" \
        && export ${attr}="$VAL" \
        && VARS="$VARS $attr"
####endfx
        ;;
  esac
done
) | sed '$!N; /^\(.*\)\n\1$/!P; D'
# ... this sed deletes consecutive duplicates from the whole output

exit ${result:-0}
```

Prueba:

```
address_ldap_lookup.sh ast
```

Podemos transferir el script tal cual a otros sistemas, por ejemplo a dklab2:

```
scp /usr/local/bin/address_ldap_lookup.sh \
    dklab2.casafx.dyndns.org:/usr/local/bin/
ssh dklab2.casafx.dyndns.org "
    echo set query_command=\'/usr/local/bin/address_ldap_lookup.sh\'
    | cat >/etc/Muttrc.d/local-00-query_command.rc"
```

1.4.2. Mozilla Thunderbird

Note: este programa tiene una interfaz gráfica, por tanto y tal como se introdujo en testbed.org, el programa debe lanzarse en un entorno con servidor X11, normalmente

apuntado por la variable de entorno DISPLAY.

Instalación; extensiones

Básicamente, debemos instalar thunderbird tal como lo empaqueta debian. En concreto, y por motivos legales sobre la marca, el nombre pasa a ser "icedove". Así:

icedove Mozilla Thunderbird empaquetado por debian. Algunas extensiones también están presentes en el sistema de paquetes de debian, con la ventaja de que se instalarán para todos los usuarios, a diferencia de la instalación desde la interfaz de usuario de icedove:

xul-ext-sieve Thunderbird Sieve add-on: thunderbird([[https://addons.mozilla.org/en-US/thunderbird/addon/sieve/][addon]],[[http://adullact.net/plugins/mediawiki/wiki/milimail
http://sieve.mozdev.org/installation.html

enigmail extensión para funcionalidades criptográficas (soporte openPGP etc)

Opcionalmente podemos instalar también:

iceowl-extension extensión soporte de calendario

xul-ext-nostalgy extensión atajos de teclado

Algunas de las extensiones sólo aparecen en los repositorios "backports" que mencionamos en el capítulo de postgresql; si no se añadieron entonces:

```
if ! grep ^deb.*backports /etc/apt/sources.list; then
echo 'deb http://backports.debian.org/debian-backports squeeze-backports main' \
  >> /etc/apt/sources.list
fi
```

Instalamos, finalmente, el software:

```
apt-get install icedove icedove-l10n-es-es \
                xul-ext-sieve enigmail xul-ext-nostalgy iceowl-extension
```

Depuración de protocolos

Note: to log both IMAP and SMTP traffic in /tmp/icedove.log, make these variables available:
http://www.mozilla.org/projects/nspr/reference/html/prlog.html
http://email.about.com/od/mozillathunderbirdtips/qt/et_mail_log.htm

```
export NSPR_LOG_MODULES=IMAP:4,SMTP:4 # Tambie'n es posible establecer "all:5"
export NSPR_LOG_FILE=/tmp/icedove.log # Si no establecida, se usa stdout/stderr
```

Configuración

La configuración, modificable desde la interfaz gráfica, es almacenada en el fichero "prefs.js" bajo ~/.icedove/<profilename>/. Junto a otros, será creado en el primer arranque. Es texto plano, así que puede inspeccionarse con:

```
ICEDOVE_PROF="`awk -F= '{if ($1 ~ /Path/) {print $2}}'
              ~/.icedove/profiles.ini`"
view ~/.icedove/$ICEDOVE_PROF/prefs.js
```

Desafortunadamente, icedove aún no soporta el descubrimiento de los servicios a través de DNS SRV RR (si bien se planea usarlo en el futuro, véase https://developer.mozilla.org/en/Thunde Así, no nos queda más remedio que explicitar algún servidor como dklab1.casafx.dyndns.org (A RR), en lugar de casafx.dyndns.org (SRV RR).

```
DISPLAY=10.0.2.2:0 \
icedove
```

La primera vez saltará un asistente "Mail Account Setup":

- Your name: aemu casafx user
- Email: aemu@casafx.dyndns.org
- Password: <dejar en blanco>

Click en "Continue".

Entonces aparecerá "Icedove is looking" ... fallará al descubrir los servicios, pues aún no usa DNS RR SRV; debido a ésto, los datos son (dklab1 puede sustituirse por dklab2):

- Username: aemu
- Incoming: dklab1.casafx.dyndns.org IMAP 143 STARTTLS
- Outgoing: dklab1.casafx.dyndns.org SMTP 587 STARTTLS

De todas formas pulsamos sobre Manual Configuration (lo cual sería equivalente a, fuera del asistente en la típica barra de menúes superior izquierda, elegir Edit->Accounts). Vamos allá:

Edit->Accounts (Manual Configuration)

Item aemu@casafx.dyndns.org (en el panel lateral)

- Account name: aemu@casafx.dyndns.org
- Your name: aemu casafx user
- Email address: aemu@casafx.dyndns.org

Resto no modifíquese, en principio. El valor para "Outgoing Server" se establecerá automáticamente tras configurar uno más tarde.

Server settings (en el panel lateral, bajo item aemu@casafx.dyndns.org)

- Server Type Imap Mail Server
  - Server
    - Name: dklab1.casafx.dyndns.org (pues no parece que use aún DNS SRV RR)
    - Port: 143
    - User Name: aemu
  - Security Settings
    - Connection security: STARTTLS
    - [X] Use secure authentication (ésto debería habilitar SASL-GSSAPI)
  - Botón “Advanced”:
    - IMAP server directory: <blank>
    - [ ] Show only subscribed folders
    - [X] Server supports folders that contain sub-folders and messages
    - [X] Use IDLE command if the server supports it
    - These preferences specify the namespaces on your IMAP server
      - Personal: <empty>(Icedove encontrará automáticamente "")
      - Public: <empty>(Icedove encontrará automáticamente "Public/")
      - Other Users: <empty>
    - [X] Allow server to override these namespaces
  - Sieve settings (también en el panel lateral bajo item aemu@casafx.dyndns.org)
    - [X] Yes, manage Sieve scripts for this account
    - Change Settings:
      - Server Name: dklab1.casafx.dyndns.org
      - Port: 4190

        - ◇ Authentication: use login from IMAP Account. Es decir, nombre usuario "aemu", y auth "segura" (pero, desafortunadamente, no soporta SASL-GSSAPI aún, como veremos luego).
        - ◇ Use TLS
    - Outgoing server SMTP (en panel lateral, item inferior). aemu - dklab1.casafx.dyndns.org->Edit
        - Description: <empty>
        - Server Name: dklab1.casafx.dyndns.org
        - Port: 587
        - Security and authentication:
            - ◇ [X] Use name and password
            - ◇ User Name: aemu
            - ◇ [X] Use secure authentication (para que use SASL-GSSAPI)
        - Connection security: STARTTLS

Click en "OK".

Como omitimos en el caso de mutt, openPGP queda fuera del alcance de este documento y no expondremos su configuración para enigmail, es decir del soporte y uso de openPGP con el sistema de correo. Thunderbird soporta S/MIME sin la ayuda de extensiones y tampoco se darán más indicaciones.

Referencias útiles podrían ser:
http://enigmail.mozdev.org/documentation/quickstart-ch2.php.html
http://kb.mozillazine.org/Getting_an_SMIME_certificate

Edit->Preferences

- Composition->Pestaña Addresssing ->Directory Server, Edit Directory->Add
    - Name: dklab1
    - Hostname: dklab1.casafx.dyndns.org
    - Base DN: ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org
    - Port Number: 389
    - Bind DN: <blank>
    - [ ] Use secure conection TLS (no)
    - Pestaña Advanced
        - (No modifíquese en principio. Autentificación plain (parece admitir GSSAPI). Ningún filtro).

Click en "Ok". (Si volvemos a editarlo, su pestaña Offline nos permite ahora probar si funciona mediante la creación de una caché local "ldap.map", bajo ~/.icedove/<profile>/). De vuelta en la ventana de preferencias, sustituímos None por "dklab1" para que éste sea el perfil del Directory Server ldap.

- Display->Fromatting->Fonts, Advanced->Character Encodings
  - Elíjase "Unicode (UTF-8)" al menos para Outgoing Mail.
- Advanced ->Certificates->View->Pestaña Authorities
  - Import /etc/ssl/certs/casafx-ca.crt, acéptese, en principio todo uso [X].

Click en "Ok". Ésto es importante porque icedove no sabrá verificar el certificado que le presenten los servidores, de forma automática no parece encontrar el certificado de la CA almacenado en /etc/ssl/certs/casafx-ca.crt (no sabemos qué implementación de TLS utiliza, probablemente una propia, pero parece que no es openssl por tanto). Congruentemente a la configuración de los servidores, los clientes no están obligados a presentar un certificado que les identifique.

http://kb.mozillazine.org/Thunderbird_:_FAQs_:_Import_CA_Certificate

Click en "Close".

IMAP test

- Menú contextual sobre item aemu@casafx.dyndns.org en panel lateral ->Seleccionamos "Open in new tab". Ésto desencadena la autenticación SASL-GSSAPI así como el listado de carpetas y mailboxes. Como resultado, además de nuestra INBOX, aparece automáticamente la carpeta Spam, así como la carpeta para el espacio imap "Public/" y su mailbox "newsletter". No estamos suscritos a ellas, pero en la configuración se indicó que se mostrasen también los elementos sin subscripción. El estado de las subscripciones se puede administrar con:
- Menú contextual sobre aemu@casafx.dyndns.org ->"Subscribe". Se comprobó que icedove tenía dificultades para actualizar su panel acorde al nuevo estado de las subscripciones sin antes reiniciar.

Otras características como el uso de nuestra quota se pueden consultar con:

- Menú contextual sobre INBOX ->properties.

SMTP test

- Message->New

En From, al empezar a escribir el nombre, icedove intentará autocompletar según ldap, es decir según los atributos mail de los objetos bajo ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=or
Por ejemplo al poner "aem", se genera, vemos en syslog, la búsqueda para (mail=aem*) :

```
slapd[1728]: conn=1029 op=3 SRCH
base="ou=users,ou=accounts,dc=casafx,dc=dyndns,dc=org"
scope=2 deref=0
filter="(|(cn=aem*)(mail=aem*)(sn=aem*))"
```

... y con ayuda de los tres metadatos, compone automáticamente la dirección "cn sn <mail>". Los campos para dirección, asunto y cuerpo del correo deben quedar así:

From:	aemu casafx user <aemu@casafx.dyndns.org>
Subject:	hola
Body:	hola desde icedove

Click en "Send Message".

Nota: la siguiente sentencia en línea de comandos, lanzada hipotéticamente por aemu:

```
icedove -compose "to='aemu@casafx.dyndns.org',
                  subject='thunderbird test',
                  attachment='file:///home/admin/tip.txt',
                  body=`cat ~/icedove/*/prefs.js`
                 "
```

... permitiría lanzar una ventana de composición de correos con esos datos. No parece posible que mandese el correo directamente, sin mostrar salida gráfica alguna como paso intermedio.

http://kb.mozillazine.org/Command_line_arguments_%28Thunderbird%29

MANAGESIEVE test (no soportado aún nuestro esquema de autenticación)

Problemática: "Thunderbird sieve extension" no soporta SASL-GSSAPI (y sí Plain, Login, CRAM-MD5, SCRAM-SHA1). El autor de la extensión, Thomas Schmidt, nos contestó muy amablemente y de forma informal sobre la posibilidad de implementar el soporte para SASL-GSSAPI:

```
"well over the past years there have been several request concerning
 GSSAPI. But Thunderbird's GSSAPI Interface is not scriptable, this
 means it is not accessible from JavaScript or Extensions. So there is
 currently no way to implement it. I suppose this will not change
 unless someone makes Thunderbird's GSSAPI Interface scriptable, or
 reimplementing the GSSAPI wrapper in JavaScript with C-Types. Both
 is doable but means a lot of work."
```

Además, anima a los usuarios a presentar peticiones de funcionalidades no disponibles. Hemos creado una de estas incidencias, a la que se ha sumado algún comentario adicional y puede seguirse en:

https://bugzilla.mozilla.org/show_bug.cgi?id=711875

Si la solución llegase[3], probar la infraestructura sería tan sencillo como:

- Tool->Sieve Message Filter.

Debiera listar los filtros presentes en el servidor. Ésto sería suficiente para entender que la extensión que instalamos funciona. Ejemplos de scripts típicos listos para usar podrían encontrarse en: http://sieve.info/examplescripts

¿Alternativas? No. De http://sieve.info/clients, nos interesan los MUA gráficos multiplataforma, pero los que cumplen esta premisa no aportan mucho más:

- Opera: no soporta managesieve (http://my.opera.com/community/forums/topic.dml?id %3D95
- Mulberry (FLOSS): no está empaquetado para debian, si bien según su documentación tampoco parece que pueda usar SASL-GSSAPI.

... no dedicaremos más tiempo a la búsqueda de clientes MANAGESIEVE gráficos con soporte SASL-GSSAPI: por ahora sólo es posible subir scripts sieve remotamente a través de la utilidad en línea de comando sieve-connect.

1.4.3. Web Clients (Webmail. No soportado aún nuestro esquema de autenticación)

Su problema es que no parecen soportar SASL-GSSAPI como método de autenticación. No obstante, si esta situación cambiase, podríamos mencionar que:

- Clientes SMTP e IMAP como aplicaciones web (por ejemplo, sobre apache2 + libapache2-mod-php5)

[3]En Abril de 2012 Thomas sugiere también la posibilidad de usar un proxy GSSAPI implementado en Perl por Yubao Liu:
http://sourceforge.net/p/managesieve/blog/2012/04/gssapi/

roundcube basado en AJAX

squirrelmail éste no precisa necesariamente soporte javascript o html >v4 en el cliente.

- Extensiones para MANAGESIEVE existen igualmente para ambos:

sieverules para roundcube, http://www.tehinterweb.co.uk/roundcube/plugins/sieverules.tar.gz

avelsieve para squirrelmail, http://email.uoa.gr/projects/squirrelmail/avelsieve.php

Sobre la situación de GSSAPI en ambos, puede leerse algo en:
http://old.nabble.com/help-with-roundcube-pubcookie-plugin-td30342723.html
[4]

1.5. Problemas con otros MTA; smarthost condicional

1.5.1. Justificación y diseño

En ocasiones, el MTA de un dominio no nos permite enviar correo a sus cuentas. Ésto puede tener muy diversas causas, pero dado que nuestro sistema incorpora algunas características que ayudan a autorizar y demostrar nuestra identidad (como DKIM o SPF), y suponiendo que la IP de nuestros equipos en producción será estática y no habrá formado parte de ninguna lista RBL, podemos considerar improbables estos problemas de interoperabilidad. Si, no obstante, se presentaran, el MTA problema podría darnos (al rebotar el correo) alguna razón orientativa o incluso una URL con la solución o la forma de ponerse en contacto.

Expuesto el problema y nuestra confianza en que suele ser solucionable, nos preguntamos ahora si, aún así, merece la pena hacer un despliegue tan complejo como el que se ha acometido, a pesar de existir la posibilidad de que no sea totalmente funcional. Si descubriésemos que sistemáticamente no podemos interoperar con algunos dominios, aunque sólo fuese durante una temporada, tendríamos que plantearnos utilizar y quizás desplegar algo adicional mientras tanto. Ésto sería una incongruencia. Improbable, pero posible.

Por tanto, en este apartado nos dedicamos a estudiar posibles modificaciones adicionales que solventasen, si se diese el caso, el problema de interoperabilidad. Se espera con ello que el tiempo invertido en el sistema de correo que hemos presentado, merezca la pena cualquiera que sea el contexto en que se pretenda poner en producción.

En terminología de sistemas de correo electrónico, un "smarthost" es un MTA ajeno quizás a nuestra organización pero que, cumplidas unas condiciones (como una exitosa autenticación SMTP-AUTH), realiza la transferencia de nuestros correos hacia otros dominios. Es un intermediario.

[4]Nota eximstats: Como se comentó, podemos ver unas sencillas estadísticas sobre la actividad de exim4 dejando que eximstats inspeccione el log. Tras toda esta batería de pruebas, se recomienda comprobarlo con los comandos que se expusieron al hablar de eximstats.

Proveedores de servicios de correo suelen ofrecer esta posibilidad a sus usuarios. Ésta tiene precisamente la ventaja de que la práctica totalidad de los MTA's en internet confían en algunos de estos MTA-smarthost, abriendo así el camino entre nuestro correo saliente y cualquier otro destinatario. Evidentemente, también hay desventajas, la más obbia es dejar nuestro correo en terceras manos.

Nuestra estrategia será tener una reserva de smarthosts que se harán cargo del correo saliente, pero sólo cuando vaya destinado a los dominios que no interoperan con nosotros. Son, así, smarthosts condicionales. Veamos cuál sería el diseño general para configurar estos smarthosts condicionales:

- Los metadatos relativos a los smarthosts estarían en ldap. Puesto que el esquema qmail-ldap no incluía soporte para ésto, diseñamos el nuestro propio y lo añadimos a qmail-ldap antes de cargarlo en cn=config. Así, openldap ya está preparado para reconocer objetos de tipo "smarthost" y todos sus atributos (dominios problemáticos, ruta al MTA-smarthost, dominios ante los que se precisa autenticación, nombre de usuario y contraseña...).

- Respecto a la configuración de exim4, consistiría en crear un router que se evalúe antes que los hasta ahora previstos, detecte los correos a las direcciones problemáticas y consiga la ruta de su MTA-smarthost. El driver que permite ésto es "manualroute".

- El router llamará a su transport. Debemos pues crear también ese transport y hacer que abra una comunicación SMTP/LMTP con el MTA intermediario y averiguar si hace falta hacer SMTP-AUTH. El driver que permite ésto es "smtp".

- Para la autentificación, se crearía un autenticador para exim4 actuando con el rol de cliente. Es decir, cuando los usuarios de nuestro dominio mandaban correos a nuestros MTA, esos exim4 actuaban de autenticador y los clientes pretendían ser autenticados, utilizándose el mecanismo SASL-GSSAPI para permitirles hacer SSO, etc. Ahora, cuando exim4 (de parte de nuestros usuarios) intente mandar un correo conectándose al MTA-smarthost, nuestro exim4 no será el autenticador sino que pretenderá ser el autenticado, y el mecanismo que se use en este caso deberá ser, necesariamente, el que nos pida el MTA-smarthost y que probablemente no será ya SASL-GSSAPI (no tiene sentido ya en ese contexto). Véase http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch33.html, "5. Authentication by an Exim client".

No podemos profundizar más en la configuración del smarthost condicional sin tener en cuenta el tipo de problemas de interoperabilidad que tiene nuestra organización. Por ejemplo, dependiendo del contexto podría resultar adecuado hacer que cada usuario tenga un smarthost asociado, o no: que haya un pool que usen todos. Así, el objetivo que nos hemos planteado es el de introducir una configuración compleja sólo hasta el nivel que permita mostrar cómo se utiliza el esquema ldap propuesto, y cómo configurar los mencionados router/transport/authenticator. En base a ello, el administrador debería poder luego adaptarlo fácilmente a su conveniencia. Así, en nuestro despliegue de ejemplo:

- utilizaremos un pool común de smarthosts localizable en la rama ou=mailextra,dc=casafx,dc=d
  de nuestro directorio ldap. Esta vez no utilizaremos macros en exim4 para declararla,
  pero sigue siendo posible.
- el smarthost condicional estará disponible tanto para las cuentas centralizadas como las locales (pues el router no verificará si la cuenta del remitente es o no centralizada).
- se dan indicaciones para usar dos servicios de smarthost concretos: el proveído por Gmail (Google) y el proveído por Live/Hotmail (Microsoft). Éstos utilizan PLAIN sobre TLS como método de autentificación, y necesitan declarar previamente las posibles direcciones de remitente que se van a utilizar (en caso contrario, reescribe sistemáticamente el contenido de la cabecera "From" a la cuenta con que se hace la autentificación con el MTA-smarthost). Hemos escogido estos smarthost simplemente porque disponemos de cuentas para hacer las pruebas, a la vez que cumplen el ser considerados confiables para la práctica totalidad de MTA's en internet.
- cada objeto "smarthost" se hace responsable de un conjunto de dominios problemáticos, siendo este conjunto disjunto con el de los demás objetos "smarthost".
- No se describe, pero ha de tenerse en cuenta para un despliegue en producción:
  - cómo obtener cuentas en Gmail, Live/Hotmail, o cualquier otro MTA-smarthost fiable.
  - nótese que podríamos tener varias cuentas, por ejemplo en gmail, de forma que no sólo una de ellas se ocuparía de un conjunto disjunto de los dominios problemáticos, en ese caso habría varias para esos dominios pero todas en el mismo smarthost (y por tanto con el mismo tipo de SMTP-AUTH). Ésto no se contempla, aunque probablemente baste con buscar una función de exim4 que aleatorice los pares <loginname>:<password>que devolvería ldap, e incluirla en el autentificador.
  - tampoco se describe cómo eliminar el correo almacenado en el típico buzón de enviados de la cuenta intermediaria. Hay varias posibilidades; la más obbia es buscar la opción que deshabilita copias de correos enviados (Gmail la tiene). Si ésto no interesase exactamente así, se puede lanzar un script que las borre cuando fuese necesario: desde tareas configuradas en el sistema Cron hasta la función "run" de exim4. La posibilidad de automatizar la tarea con un script viene del hecho de que Gmail provee acceso imap. En el caso de Live/Hotmail sólo sabemos que permite acceso pop(el mecanismo de autenticación es PLAIN en ambos casos). Ejem:

    ```
    mutt -f imaps://usuarioG@imap.gmail.com/[Gmail]/Enviados

    mutt -f pops://usuarioH@hotmail.com@pop3.live.com
    ```

  - como se dijo, el objeto ldap "smarthost", no "structural", permite hacer configuraciones per user que no se muestran aquí.

- el administrador querrá configurar (conexión ldapi:// %2Fvar %2Frun %2Fslapd %2Fldapi, ACL's, TLS...) formas para que información sensible como la de autentificación, esté protegida.
    - Anotación: la solución más obbia en nuestro despliegue es interponer una ACL que sólo permita leer los nodos bajo ou=smarthosts,ou=mailextra a la identidad que usa exim4, es decir, al DN al que se mapea el principal Kerberos que usa exim4 para, hasta ahora, conseguir tokens AFS. Nosotros creamos dicha ACL así como, similar a los recursos creados para la replicación ldap, un mapeo y una rama específicas para la identidad de exim4 en ldap. Desafortunadamente el sistema no funcionaba y la razón es que los "ldap lookup" de exim4 no implementan SASL-GSSAPI. Puede verse la discusión al respecto que iniciamos en la lista de correos exim-users aquí: https://lists.exim.org/lurker/thread/20120919.141138.b080bef3.en.html
- eventualmente, a modo de inspección del problema, puede ser interesante una simple configuración sin metadatos centralizados en ldap. Es nuestra experiencia. Puede encontrarse algunas indicaciones en:
    - http://www.linode.com/forums/viewtopic.php?p=12840&sid=88f773bc334c427df4b7

1.5.2. Soporte en ldap para el pool de smarthost de Gmail y Hotmail

Un smarthost para Gmail, otro para Hotmail. Se incluyen en el ldif ejemplos de dominios problemáticos.

```
vim ~/ldif/mailextra.ldif
```

```
dn: ou=mailextra,dc=casafx,dc=dyndns,dc=org
changetype: add
objectClass: organizationalUnit
ou: mailextra

dn: ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org
changetype: add
objectClass: organizationalUnit
ou: smarthosts

# Nota: Necesitamos an~adir el objeto inetOrgPerson (core.schema/rfc2256)
#       porque nuestro objeto smartHost no es STRUCTURAL en aras de la
#       flexibilidad. Respecto a inetOrgPerson, es un objeto esta'ndar
#       y por tanto definido en los esquemas que carga slapd por defecto.
#       Adema's, define como obligatorios los atributos sn y cn que, puesto
#       que no los vamos a usar, pueden tener cualquier valor, por ejemplo "_".
# Nota: MUST ( smarthostID $ smarthostName $ smarthostRoute $ domainMakesUsUseSmart
```

```
dn: smarthostID=usuarioG@gmail.com,ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,d
changetype: add
objectClass: smartHost
objectClass: inetOrgPerson
sn: _
cn: _
smarthostID: usuarioG@gmail.com
domainMakesUsUseSmarthost: gmail.com
domainMakesUsUseSmarthost: ujaen.es
smarthostLoginName: usuarioG@gmail.com
smarthostPass: supasswordG
smarthostPort: 587
smartHostName: gmail-smtp-msa.l.google.com
smarthostRoute: smtp.gmail.com::587 bydns
#Nota: smarthostRoute y smarthostName han sido configurados atendiendo a que:
#dig _submission._tcp.gmail.com SRV +short ...responde smtp.gmail.com
#dig smtp.gmail.com A +short ...responde que es un CNAME a
#gmail-smtp-msa.l.google.com
```

```
dn: smarthostID=usuarioH@hotmail.com,ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns
changetype: add
objectClass: smartHost
objectClass: inetOrgPerson
sn: _
cn: _
smarthostID: usuarioH@hotmail.com
domainMakesUsUseSmarthost: hotmail.com
domainMakesUsUseSmarthost: hotmail.es
domainMakesUsUseSmarthost: hotmail.co.uk
domainMakesUsUseSmarthost: msn.com
domainMakesUsUseSmarthost: live.com
smarthostLoginName: usuarioH@hotmail.com
smarthostPass: supasswordH
smarthostPort: 587
smartHostName: smtp.hot.glbdns.microsoft.com
smarthostRoute: smtp.live.com::587 bydns
#Nota: smarthostRoute y smarthostName han sido configurados atendiendo a que:
#dig _submission._tcp.hotmail.com SRV +short ...no responde nada, pero
#es conocido smtp.live.com, que es un CNAME a smtp.hot.glbdns.microsoft.com
#dig smtp.live.com A +short
```

NOTA: es válido también <host>/MX al declarar rutas, y en principio es válido puesto que el sistema que envía el correo al smarthost dispone de una cola y por tanto, no es necesario usar el protocolo LMTP en puerto 587; sin embargo, estos proveedores dictan esta configuración para usar su servicio. Por su lado "bydns" preguntará por el DNS RR tipo A.

```
kinit -p root/admin
ldapmodify -c -S /tmp/mailextra.err \
           -Qf ~/ldif/mailextra.ldif

KRB5CCNAME=/tmp/.krb5cc_800_exim4 su vmail -m -c \
 "ldapsearch -QLLL -b
  'ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org'
 -s sub -a never '(smarthostID=usuarioG@gmail.com)' cn"
```

```
 cn: _
```

```
exim4 -d+all -be '${lookup ldapm{ dereference=never time=15
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
domainMakesUsUseSmarthost?sub}}'
```

No usa GSSAPI para acceder a ldap[5], así que cambiamos la ACL, con el aviso y las precauciones que se expusieron:

1.5.3. Router para el smarthost condicional

```
vim ~/ldif/read_access_to_mailextra_tree.ldif
```

```
dn: olcDatabase={1}hdb,cn=config
add: olcAccess
olcAccess: {5}to dn.subtree="ou=mailextra,dc=casafx,dc=dyndns,dc=org"
  by * read
```

```
ldapmodify -QY EXTERNAL -H ldapi:/// \
           -f ~/ldif/read_access_to_mailextra_tree.ldif
```

[5] http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch09.html#SECID70

La misma modificación en cn=config ha de hacerse en el resto de servidores ldap. De dklab1 a dklab2 podemos hacerlo automáticamente con:

```
scp ~/ldif/read_access_to_mailextra_tree.ldif \
  dklab2.casafx.dyndns.org:/root/ldif/read_access_to_mailextra_tree.ldif


ssh dklab2.casafx.dyndns.org '
 ldapmodify -QY EXTERNAL -H ldapi:///
            -f /root/ldif/read_access_to_mailextra_tree.ldif'
```

```
vim /etc/exim4/exim4.conf.template
```

```
...
### router/200_exim4-config_primary
#################################
####fx:
.include_if_exists /etc/exim4/example_conditional_smarthost_support-router
####endfx
...
```

```
vim /etc/exim4/example_conditional_smarthost_support-router
```

```
####fx:
unfriendly_domains:
  driver = manualroute
  # manualruoute implica no enrutar necesariamente utilizando registros
  # MX o A segu'n $domain_part, sino con las reglas explicitadas
  # en route_list o route_data (e'sta si lookups)

  transport = remote_conditional_smarthost
  self = pass
  hosts_randomize
  route_data = ${lookup ldap{LDAP_DEFAULT_CONTROLS \
         ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
                                                  smarthostRoute?sub?\
                              (smarthostID=${lookup \
                                    ldap{LDAP_DEFAULT_CONTROLS \
         ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
                                                    smarthostID?sub?\
                                       (domainMakesUsUseSmarthost=${domain})\
                                    }\
                               }\
                            )\
                          }\
                }
  no_more
####endfx
```

Cambios idénticos debieran realizarse sobre el resto de servidores exim4.

Test de acceso con exim4 string-expansion

```
exim4 -be '${sg{${sg{${sg{${lookup
ldapm{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
domainMakesUsUseSmarthost?sub?(objectClass=smartHost)}}}{\n}{:}}}{,}{:}}}{ }{}}'
```

```
 gmail.com:ujaen.es:hotmail.com:hotmail.es:hotmail.co.uk:msn.com:live.com
```

```
exim4 -be '${lookup
ldap{ ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostRoute?sub?(smarthostID=${lookup ldap{
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostID?sub?(domainMakesUsUseSmarthost=gmail.com)}})}}'
```

```
   smtp.gmail.com::587 bydns
```

```
exim4 -be '${lookup ldap{
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostRoute?sub?(smarthostID=${lookup ldap{
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostID?sub?(domainMakesUsUseSmarthost=hotmail.com)}})}}'
```

```
   smtp.live.com::587 bydns
```

1.5.4. Transport para el smarthost condicional

Nota: no es necesario demostrar nuestra identidad a través de DKIM puesto que se usará SMTP-AUTH.

```
vim /etc/exim4/exim4.conf.template
```

```
...
### transport/30_exim4-config_remote_smtp_smarthost
#################################
####fx:
.include_if_exists /etc/exim4/example_conditional_smarthost_support-transport
####endfx
...
```

```
vim /etc/exim4/example_conditional_smarthost_support-transport
```

```
####fx:
# This transport is used for delivering messages over SMTP connections
# to a smarthost. The local host tries to authenticate.
# This transport is used for smarthost and satellite configurations.
remote_conditional_smarthost:
  debug_print = "T: remote_smtp_smarthost for $local_part@$domain"
  driver = smtp
  tls_tempfail_tryclear = false
  #hosts_require_auth
  hosts_try_auth = ${sg{\
                       ${lookup ldap{LDAP_DEFAULT_CONTROLS \
         ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
                               smarthostName?sub?(smarthostID=\
                       ${lookup ldap{LDAP_DEFAULT_CONTROLS \
         ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
                               smarthostID?sub?\
                                                    }\
                                               }\
                                    )\
                                   }\
                        }\
                       }{,}{:}}
```

```
  #NOTA: sustituyo ',' por ':' para lidiar con la multiplicidad, pero puede
  #      haber espacios en blanco tras ':', si bien son permitidos:
  #      "White space surrounding the colons is ignored"
  #      http://www.exim.org/exim-html-4.72/doc/html/spec_html/ch10.html
  #NOTA: que en el original poni'a tras la comprobacio'n, a la
  #      direccio'n IP $host_address:
  #hosts_try_auth = <; ${if exists{CONFDIR/passwd.client} \
  #      {\
  #      ${lookup{$host}nwildlsearch{CONFDIR/passwd.client}{$host_address}}\
  #      }\
  #      {} \
  #    }
####endfx
```

Cambios idénticos debieran realizarse sobre el resto de servidores exim4.

Test de acceso con exim4 string-expansion

```
exim4 -be '${sg{${lookup
ldap{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostName?sub?(smarthostID=${lookup ldap{ldap:///ou=smarthosts,ou=mailextra,dc
smarthostID?sub?(domainMakesUsUseSmarthost=ujaen.es)}})}}}{, }{:}}'
```

```
    gmail-smtp-msa.l.google.com
```

```
exim4 -be '${sg{${lookup
ldap{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostName?sub?(smarthostID=${lookup ldap{ldap:///ou=smarthosts,ou=mailextra,dc
smarthostID?sub?(domainMakesUsUseSmarthost=hotmail.com)}})}}}{, }{:}}'
```

```
    smtp.hot.glbdns.microsoft.com
```

1.5.5. Authenticator para el smarthost condicional

```
vim /etc/exim4/exim4.conf.template
```

```
...
begin authentificators
...
cram_md5:
...
# this returns the matching line from passwd.client and doubles all ^
####fx:
#-PASSWDLINE=${sg{\
#-               ${lookup{$host}nwildlsearch{CONFDIR/passwd.client}{$value}fail}\
#-               }\
#-               {\\N[\\^]\\N}\
#-               {^^}\
#-           }
#${sg{${lookup{target.com}nwildlsearch{/etc/exim4/passwd.client}{$value}fail}}{\N[
.include_if_exists /etc/exim4/example_conditional_smarthost_support-auth_as_a_clien
#El lookup utiliza sg para doblar los ^ en username/password.
#E'sto es asi' porque en el bloque authenticator, los ^ individuales
#en client_send representan (se convertira'n en) \0, pues el formato
#con que se envi'an el usuario y el password en el me'todo de auth
#"plain" los precisa. Los ^^ se envi'an como ^.
####endfx
```

```
plain:
  driver = plaintext
  public_name = PLAIN
.ifndef AUTH_CLIENT_ALLOW_NOTLS_PASSWORDS
  client_send = "<; ${if !eq{$tls_cipher}{}\
                    {^${extract{1}{:}{PASSWDLINE}}\
                     ^${sg{PASSWDLINE}{\\N([^:]+:)(.*)\\N}{\\$2}}\
                   }fail}"
.else
  client_send = "<; ^${extract{1}{:}{PASSWDLINE}}\
                    ^${sg{PASSWDLINE}{\\N([^:]+:)(.*)\\N}{\\$2}}"
.endif
...
```

```
vim /etc/exim4/example_conditional_smarthost_support-auth_as_a_client
```

```
####fx:
# Ba'sicamente, redefinimos el macro PASSWDLINE, y de la forma
# ma's compacta posible.
PASSWDLINE= ${map\
                {smarthostLoginName:smarthostPass}\
                {${extract{$item}{\
                                ${lookup ldap{dereference=never time=15 \
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
smarthostLoginName,smarthostPass?sub?(smarthostID=\
${lookup ldap{dereference=never time=15 \
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?\
smarthostID?sub?(domainMakesUsUseSmarthost=${domain})}})}\
                                        }\
                                    }\
                {${sg{$value}{\\N[\\^]\\N}{^^}}}}}\
            }
####endfx
```

Cambios idénticos debieran realizarse sobre el resto de servidores exim4.

Test de acceso con exim4 string-expansion

```
exim4 -be '${lookup
ldap{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostLoginName?sub?(smarthostID=${lookup ldap{ldap:///ou=smarthosts,ou=mailext
smarthostID?sub?(domainMakesUsUseSmarthost=gmail.com)}})}}'
```

```
    usuarioG@gmail.com
```

```
exim4 -be '${lookup ldap{
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostPass?sub?(smarthostID=${lookup ldap{
ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostID?sub?(domainMakesUsUseSmarthost=gmail.com)}})}}'
```

```
mipass
```

```
exim4 -be '${map{smarthostLoginName:smarthostPass}{${extract{$item}{${lookup
ldap{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostLoginName,smarthostPass?sub?(smarthostID=${lookup
ldap{ldap:///ou=smarthosts,ou=mailextra,dc=casafx,dc=dyndns,dc=org?
smarthostID?sub?(domainMakesUsUseSmarthost=gmail.com)}})}}}
{${sg{$value}{\N[\^]\N}{^^}}}}}}'
```

```
usuarioG@gmail.com:mipass
```

Todo junto:

```
exim4 -be
'^${extract{1}{:}{usuarioG@gmail.com:mipass}}^
${sg{usuarioG@gmail.com:mipass}{\N([^:]+:)(.*)\N}{\$2}}'
```

```
^usuarioG@gmail.com^mipass
```

1.5.6. Tests del smarthost condicional

Comprobación de sintaxis (y, si todo bien, reiniciamos):

```
update-exim4.conf -v
invoke-rc.d exim4 stop && sleep 2 && invoke-rc.d exim4 start
```

Comportamiento del router ante dominios problemáticos:

```
exim4 -bt otrousuarioG@gmail.com
```

```
otrousuarioG@gmail.com
 router = unfriendly_domains, transport = remote_conditional_smarthost
 host gmail-smtp-msa.l.google.com [74.125.39.109]  port=587
 host gmail-smtp-msa.l.google.com [74.125.39.108]  port=587
```

```
exim4 -bt otrousuarioG@hotmail.com
```

```
otrousuarioG@hotmail.com
 router = unfriendly_domains, transport = remote_conditional_smarthost
 host smtp.hot.glbdns.microsoft.com [65.55.96.11]  port=587
```

... el comportamiento con otros dominios deberiá estar intacto:

```
exim4 -bt usuario@student.umu.se
```

Configuración de los remitentes permitidos en las cuentas Gmail y Hotmail. Ésto nos permitía que el MTA-smarthost no reescriba el remitente original al nombre de la cuenta con que se hace la autenticación; por ejemplo, si desde cualquiera de nuestras cuentas se mandase un correo cuyo intermediario ha de ser el MTA-smarthost de Gmail, queremos evitar que "From: <cualquier_cuenta>@casafx.dyndns.org" pase sistemáticamente a "From: usuarioG@gmail.com"). Durante el proceso de configuración se nos pedirá verificación, es decir al remitente original (por ejemplo aemu@casafx.dyndns.org) se le mandará un correo indicándole cómo aceptar o, en caso contrario, abstenerse de hacer nada si considera una posible suplantación de su identidad. Lo importante de este proceso de verificación es que sólo se podrá llevar a cabo cuando nuestras cuentas puedan recibir correo desde fuera. En otro caso el correo de verificación nunca llegaría. Si éste es nuestro caso, haremos login en la aplicación web para el correo de Gmail y Live/Hotmail, en otro caso podemos saltarnos este paso a sabiendas de que la prueba completa que haremos después implica la reescritura de cabecera mencionada). Hacemos, pues, login en la aplicación web:

- Gmail (https://accounts.google.com/ServiceLogin?service=mail):
  - Desplegable superior derecho en forma de rueda dentada:
    - Configuración de cuenta
    - "Cuentas e importación"->"Enviar mensaje como"->"Añadir dirección"
      - umea@casafx.dyndns.org
      - Servidor SMTP (gmail)

        - ◇ Info: correo a tu cuenta.

- Live/Hotmail (https://login.live.com):
    - Desplegable esquina superior derecha->opciones->correo(mini panel izquierdo). Administrar tu cuenta:
        - Enviar y recibir correo electrónico de otras cuentas
        - Agregar otra cuenta desde la que enviar correo:
            - ◇ aemu@casafx.dyndns.org
            - ◇ Enviar correo de verificación. Revisar el correo, hacer click en el enlace... “Has comprobado correctamente aemu@casafx.dyndns.org.“

Vamos ya a hacer una prueba desde la cuenta aemu@casafx.dyndns.org, a la que acabamos de darle permiso para ser utilizada como remitente. Empezamos con un dominio gestionado por el MTA-smarthost de Gmail (en nuestro ejemplo, el propio gmail.com lo es, luego un correo a alguna de sus cuentas va a pasar por su MTA-smarthost):

```
login aemu
```

```
echo "Hola de aemu a usuario en dominio no amigable." \
 |mutt \
  -e "set smtp_url='smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
  -e "set smtp_authenticators='gssapi'" \
  -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
  -e "my_hdr From: aemu@casafx.dyndns.org" \
  -s "smarthost gmail.com" -- otrousuarioG@gmail.com
exit
```

```
tail -f /var/log/exim4/mainlog
```

```
1RY0Qn-0004de-Ub <= aemu@casafx.dyndns.org H=dklab1.casafx.dyndns.org
                    [10.168.1.1] P=esmtpsa X=TLS1.0:DHE_RSA_AES_128_CBC_SHA1:16
                    A=dovecot_gssapi:GSSAPI-"aemu@CASAFX.DYNDNS.ORG" S=665
                    id=20111206191141.GA17826@dklab1.casafx.dyndns.org
 1RY0Qn-0004de-Ub => otrousuarioG@gmail.com R=unfriendly_domains
                    T=remote_conditional_smarthost H=gmail-smtp-msa.l.google.com
                    [74.125.79.108] X=TLS1.0:RSA_ARCFOUR_SHA1:16
                    DN="C=US,ST=California,L=Mountain View,O=Google Inc,CN=smtp.gmail.
 1RY0Qn-0004de-Ub Completed
```

Efectivamente, allí llega un correo con las cabeceras To y, sobre todo, From adecuadas:

From:	aemu casafx user <aemu@casafx.dyndns.org>
To:	otrousuarioG@gmail.com
Subject:	smarthost gmail.com

Para el caso de envío a un dominio que pasa por smtp.live.com como MTA-smarthost:

```
login aemu
```

```
echo "Hola de aemu a usuario en dominio no amigable." \
 |mutt \
  -e "set smtp_url='smtp://aemu@dklab1.casafx.dyndns.org:587/'" \
  -e "set smtp_authenticators='gssapi'" \
  -e 'set ssl_ca_certificates_file=/etc/ssl/certs/casafx-ca.crt' \
  -e "my_hdr From: aemu@casafx.dyndns.org" \
  -s "smarthost live.com" -- otrousuarioH@hotmail.com
exit
```

(hemos usado de nuevo a dklab1, si bien si también se han realizado los cambios sobre el exim4 en dklab2, se puede sustituir smtp://aemu@dklab1... por smtp://aemu@dklab2...)

```
tail -f /var/log/exim4/mainlog
```

```
1RY40t-00053V-OR <= aemu@casafx.dyndns.org H=dklab1.casafx.dyndns.org
                    [10.168.1.1] P=esmtpsa X=TLS1.0:DHE_RSA_AES_128_CBC_SHA1:16
                    A=dovecot_gssapi:GSSAPI-"aemu@CASAFX.DYNDNS.ORG" S=596
                    id=20111206230106.GA19428@dklab1.casafx.dyndns.org
1RY40t-00053V-OR => otrousuarioH@hotmail.com R=unfriendly_domains
                    T=remote_conditional_smarthost H=smtp.hot.glbdns.microsoft.com
                    [65.55.162.200] X=TLS1.0:RSA_ARCFOUR_MD5:16 DN="CN=smtp.live.com"
1RY40t-00053V-OR Completed
```

En el destino, se recibirá el correo con la cabecera From correcta:

From:	aemu casafx user <aemu@casafx.dyndns.org>
To:	otrousuarioH@hotmail.com
Subject:	smarthost live.com

LyX