



¿Qué es el correo electrónico?

El **correo electrónico** (electronic mail, e-mail o email) es un método de almacenamiento y reenvío de mensajes. Básicamente comprende:

- Elaboración
- Envío
- Recepción

Los mensajes son transferidos de manera electrónica sobre un sistema de comunicación.

- alejandro@barcena.com.mx
- hostmaster@it.departamento2.campus5.ciudad3.escuela9.edu.mx

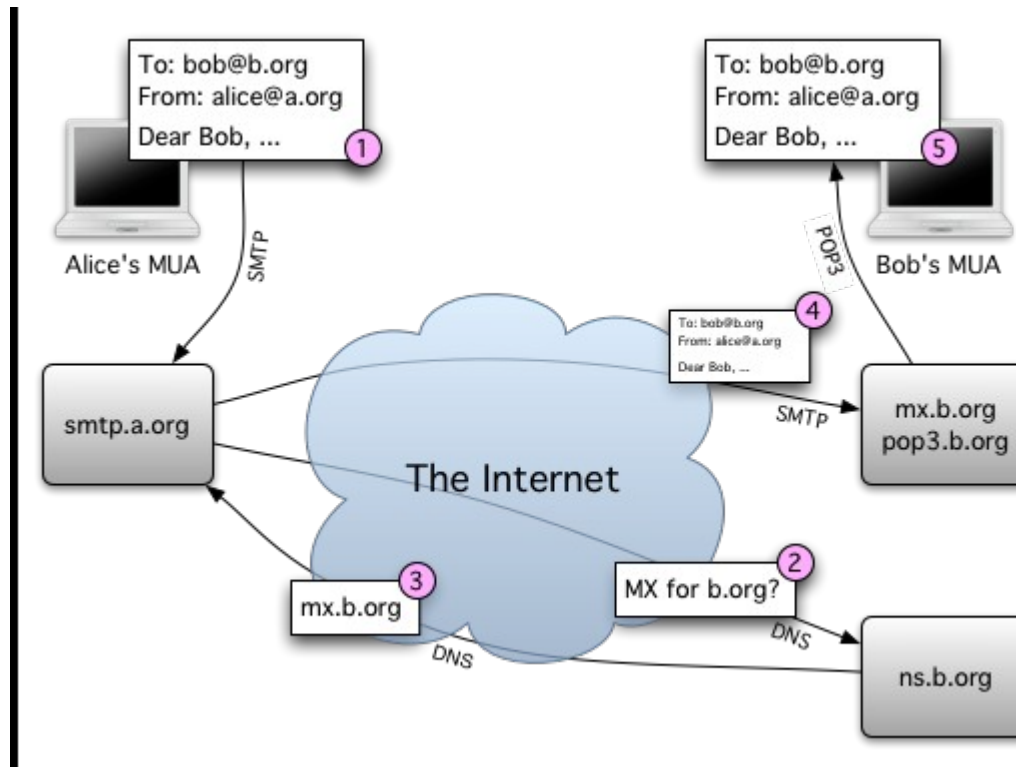
Origen del correo electrónico

El correo electrónico nació mucho antes que el Internet, de hecho es una de las herramientas que jugaron un papel crucial para la creación de Internet. El **MIT** creó el “Sistema compatible de tiempo compartido” (**CTSS**) en **1961**; éste permitía que múltiples usuarios entraran a la **IBM 7094** desde terminales remotas por dial-up, y que guardaran sus archivos en el disco duro. Esta nueva habilidad alentaba a los usuarios a compartir información de una nueva manera. El correo electrónico comenzó en **1965**, como un sistema para que los usuarios de una computadora mainframe de “tiempo compartido” pudieran comunicarse. Aunque la historia exacta sobre el correo electrónico es muy oscura, se puede decir que los primeros sistemas que contaron con esta facilidad fueron los **SDC Q32** y los **CTTS del MIT**.

El correo electrónico pronto se extendió a ser un sistema para redes, permitiendo a los usuarios enviar mensajes entre diferentes equipos en **1966**. Después ARPANET (**1969**) hizo una gran contribución a la evolución del correo electrónico, incrementó su popularidad hasta que se convirtió en su aplicación principal. En **1971** Ray Tomlinson comenzó a separar el nombre del usuario del nombre de la computadora por medio de la @.



¿Cómo funciona el correo electrónico?



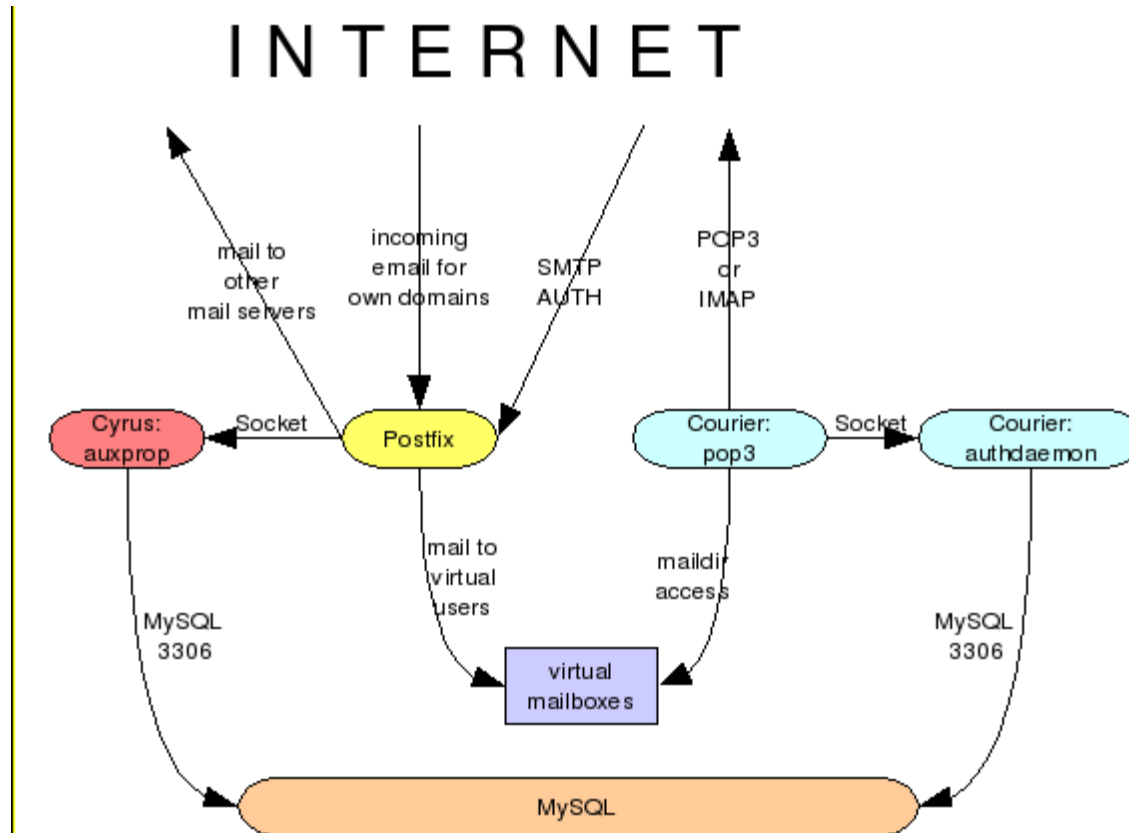


¿Qué pasa si deseo recibir correo para un dominio que no sea local?

- Dominios virtuales. Además del dominio local (/etc/defaultdomain), tu puedes recibir correo para otros dominios, llamados dominios virtuales. No existe un límite para la cantidad de dominios virtuales que quieras administrar.
- Base de datos. No es necesario que las cuentas de correo estén ligadas a los usuarios del sistema, incluso, ni siquiera tienen que estar en un archivo de texto. Postfix soporta búsquedas en los sistemas de bases de datos comunes (como MySQL y PostgreSQL). Esta técnica resulta muy cómoda ya que puedes escribir una pequeña aplicación web para administrar la base de datos, o incluso delegar la administración de las cuentas a tus usuarios.

Los componentes que emplearemos:

- **Postfix:** El **MTA** (Mail Transfer Agent) que recibe los correos vía el SMTP (Simple Mail Transfer Protocol) y los entrega en diferentes partes del disco duro.
- **SASL** (Librería **Cyrus**): Si tus usuarios se conectan fuera de tu red, Postfix no los dejará usarlo. SASL (Simple Authentication and Security Layer) les proveerá el mecanismo de autenticación para que Postfix confíe en ellos.
- **Courier:** Es un servidor de correo, como Postfix, pero sólo utilizaremos su parte de POP3/IMAP.
- **AMaViS:** Un explorador de correo que nos permitirá filtrar el contenido. AmaViS llamará a Clamav y a Spamassassin.
- **Clamav:** Antivirus para Unix.
- **Spamassassin:** Filtro de spam que emplea el análisis de texto basado en Perl.
- **MySQL:** La Base de Datos que guarda la información que controla el comportamiento de Postfix. (Usuarios, dominios, forwardings y contraseñas)





Paso 1: Instalar los paquetes de Debian que se requieren.

Paquetes absolutamente necesarios:

- postfix (Elegir: "Sólo local")
- postfix-mysql
- postfix-doc

Si la Base de Datos estará en la misma máquina:

- mysql-server

Si vamos a ofrecer acceso por medio de POP3/IMAP:

- courier-authdaemon
- courier-authmysql
- courier-pop (para POP3 no encriptado)
- courier-pop-ssl (para POP3 encriptado por SSL)
- courier-imap (para IMAP no encriptado)
- courier-imap-ssl (para IMAP encriptado por SSL)

Si deseamos aceptar correo para usuarios fuera de nuestra red:

- postfix-tls (autenticación encriptada de SMTP)
- libsasl2 (la librería Cyrus SASL)
- libsasl2-modules (el mecanismo para la librería de SASL)
- libsasl2-modules-sql
- openssl (para crear los certificados)

Si queremos analizar el correo para prevenir virus y spam:

- amavisd-new
- spamassassin
- clamav
- clamav-daemon
- zoo
- unzip
- unarj
- lha (¡en non-free!)



Paso 2: Crear la Base de Datos.

Al instalar mysql-server, éste permite la conexión del usuario root sin password, por lo que es lo primero que debemos cambiar:

```
mysqladmin -u root password password-de-mysql
```

Creamos la base de datos, la llamaremos 'proveedor'.

```
mysqladmin -u root -p create proveedor
```

Ahora necesitaremos un usuario con los permisos suficientes para acceder a nuestra base de datos. Para esto abriremos una conexión con MySQL desde el shell:

```
mysql -u root -p
```

Ingresamos el siguiente comando SQL:

```
grant select on proveedor.* to proveedor_admin@localhost identified by 'tu-password';
```

El comando pasado crea un nuevo usuario de la base de datos llamado *proveedor_admin*, quien sólo tiene el privilegio para ejecutar 'SELECT'. (Esto es sólo por cuestiones de seguridad). Para finalizar, necesitamos recargar la información de permisos en MySQL, ésto se hace con el siguiente comando de SQL:

```
flush privileges;
```



Paso 3: Crear las tablas.

Ahora crearemos las tablas que contendrán la información de control para Postfix. Primero le decimos a MySQL cuál base de datos emplearemos:

```
USE proveedor;
```

Dominios

La primera de nuestras tablas sólo contendrá una columna, el nombre del dominio virtual. De aquí Postfix tomará los datos necesarios para saber si aceptamos, o no, correo para ese dominio virtual.

```
CREATE TABLE dominios (  
    dominio varchar(50) NOT NULL,  
    PRIMARY KEY (dominio) ) TYPE=MyISAM;
```

Forwardings

La tabla *forwardings* será utilizada para generar pseudónimos de una dirección de correo a otra. También puede ser empleada para redirecciones generales (incluso para tu dominio local).

```
CREATE TABLE forwardings (  
    origen varchar(80) NOT NULL,  
    destino TEXT NOT NULL,  
    PRIMARY KEY (origen) ) TYPE=MyISAM;
```

Usuarios

Finalmente la tabla *usuarios* que tendrá la información acerca de las cuentas de correo. Cada usuario debe tener un nombre de usuario y una contraseña para acceder a su correo mediante POP3 o IMAP. Como los usuarios tienden a olvidar las cosas (no te imaginas cuántas contraseñas te puedes encontrar debajo de los teclados), he decidido emplear la cuenta de correo como el nombre de usuario. De esta manera, también postfix sabrá cómo almacenar el correo en el disco duro.

```
CREATE TABLE usuarios (  
    email varchar(80) NOT NULL,  
    password varchar(20) NOT NULL,  
    PRIMARY KEY (email) ) TYPE=MyISAM;
```



Paso 4: Trazamos los mapas de definición para la base de datos.

Es necesario indicarle a Postfix que la información de control se encuentra alojada en una base de datos. Por tal motivo crearemos 4 archivos de texto en `/etc/postfix`. Postfix corre, usualmente, en un directorio de chroot (`/var/spool/postfix`), por este motivo no puede acceder a ningún archivo fuera de él. Por lo general, un programa local se comunica con MySQL por medio de un archivo de sockets (`/var/run/mysqld/mysqld.sock`). Como te podrás dar cuenta, este archivo resulta inaccesible para Postfix; así que hay 2 opciones:

1. Tratar de cambiar de lugar el archivo de socket. Lo cuál nos traerá muchos problemas en un futuro.
2. Utilizar la red de TCP. (Así lo conectaremos)

Ventaja: No necesitamos preocuparnos por las restricciones del chroot.

Desventaja: El servidor de MySQL será accesible por la interfase lo, lo cuál en teoría podría ser un problema de seguridad.

mysql-dominios_virtuales.cf

Será un simple trazo de mapa de los dominios virtuales a la cadena *virtual*. Este mapeo es usado para la definición de `virtual_mailbox_domains`. No usar localhost en el archivo, sino Postfix tratará de realizar la conexión por el archivo de sockets.

```
user = proveedor_admin
password = tu-password
dbname = proveedor
table = dominios
select_field = 'virtual'
where_field = dominio
hosts = 127.0.0.1
```



mysql-forwardings_virtuales.cf

Este mapeo leerá la tabla de forwardings para proporcionar la manera en que se redireccionarán las direcciones de correo. Lo usaremos para virtual_alias_maps.

```
user = proveedor_admin  
password = tu-password  
dbname = proveedor  
table = forwardings  
select_field = destino  
where_field = origen  
hosts = 127.0.0.1
```

mysql-buzones_virtuales.cf

La siguiente definición trabajará con los buzones de correo de los usuarios. Le indicará a Postfix el lugar en el disco duro en el cuál deberá almacenar el correo para cada dirección de correo determinada. Lo usaremos para el mapeo de virtual_mailbox_maps.

```
user = proveedor_admin  
password = tu-password  
dbname = proveedor  
table = usuarios  
select_field = CONCAT(SUBSTRING_INDEX(email, '@', -1), '/', SUBSTRING_INDEX(email, '@', 1), '/')  
where_field = email  
hosts = 127.0.0.1
```

mysql-email2email_virtuales.cf

Un capricho de los dominios virtuales es la procedencia del mapeo virtual_alias_maps cuando usamos una cuenta concentradora (*@dominio.com->usuario@dominio.com). Por lo que ésto es necesario para resolver el problema de mapeo.

```
user = proveedor_admin  
password = tu-passwrod  
dbname = proveedor  
table = usuarios  
select_field = email  
where_field = email  
hosts = 127.0.0.1
```



Hay que asegurarnos de que sólo root pueda leer estos archivos, de lo contrario cualquier usuario del sistema podría leer la contraseña, en texto plano, para acceder a la base de datos. Para que Postfix los pueda leer, hay que cambiar el grupo al que pertenecen, y darle permiso de lectura.

```
chgrp postfix /etc/postfix/mysql-*.cf  
chmod u=rw,g=r,o= /etc/postfix/mysql-*.cf
```

Paso 5: Crear el usuario vmail.

Como todas las cuentas serán virtuales, sería absurdo asignarles un UID único a cada una. Así que recomiendo crear un pseudo usuario que será el propietario de todos los buzones de correo.

```
groupadd -g 5000 vmail  
useradd -g vmail -u 5000 vmail -d /home/vmail -m
```

Paso 6: Editar main.cf

El archivo /etc/postfix/main.cf es donde se almacena la configuración principal de Postfix. De momento no edites algo más que lo aquí presentado, ya habrá tiempo después para modificar más opciones.

Opción	Significado
inet_interfaces = all	Por defecto Debian establece esta opción a <i>loopback-only</i> durante la instalación por cuestiones de seguridad.
myhostname = ...	Asegúrate de que apunte a tu dominio completo.
mydestination = ...	Lista aquí tus dominios locales separados por comas. No incluyas los dominios virtuales.
mynetworks = ...	Lista de los rangos de IP a los cuales les daremos servicio de correo.
virtual_alias_domains =	No usaremos esta opción.
virtual_alias_maps = mysql:/etc/postfix/mysql -forwardings_virtuales.cf mysql:/etc/postfix/mysql -	Esta es una tabla de uso general para el redireccionamiento.
email2email_virtuales.cf	



Opción	Significado
virtual_mailbox_domains = mysql:/etc/postfix/mysql -dominios_virtuales.cf	Es la lista de dominios virtuales de la tabla de dominios.
virtual_mailbox_maps = mysql:/etc/postfix/mysql -buzones_virtuales.cf	Este mapeo es muy importante. Le indica a Postfix el lugar donde se encuentra el buzón de cada usuario en el disco duro.
virtual_mailbox_base = /home/vmail	Es la ruta a partir de la cual almacenaremos los buzones de correo.
virtual_uid_maps = static:5000	Le indicamos a Postfix quién debe ser el dueño de los buzones por medio del UID. En este caso el usuario vmail que acabamos de crear.
virtual_gid_maps = static:5000	Lo mismo que lo anterior, sólo que ahora para el grupo (GID).
smtpd_sasl_auth_enable = yes	Habilitamos el mecanismo de autenticación para SMTP.
broken_sasl_auth_clients = yes	Algunos clientes de correo, como el Microsoft Outlook, usan un método muy antiguo (hoy en día desaprobado) para detectar si un servidor de correo entiende la autenticación de SMTP. Hay que mantenerlos contentos =) ¿Quién dijo algo sobre Outlook?
smtpd_recipient_restrictions = permit_mynetworks, permit_sasl_authenticated, reject_unauth_destination	Estas restricciones son revisadas cuando un correo nuevo llega, para ver quién esta autorizado para usar el servidor. permit_mynetworks le dará autorización al rango de Ips. permit_sasl_authenticated lo hará para quien se haya autenticado por medio del mecanismo de SMTP.
smtpd_use_tls = yes	Después negamos el uso al resto de los usuarios, salvo si están enviando un correo que se encuentre en nuestra lista de relay_domains. Encriptar la sesión de autenticación de SMTP usando SSL.



Opción

Significado

smtpd_tls_cert_file =
/etc/postfix/smtpd.cert Ruta al certificado SSL que usará TLS. (Lo crearemos más adelante).

smtpd_tls_key_file =
/etc/postfix/smtpd.key Ruta a la llave privada del certificado SSL.

Una prueba express:

Reiniciamos Postfix (**/etc/init.d/postfix restart**) y ejecutamos **postfix check**. Si no aparecen advertencias... esta parte esta completa. =)

Paso 7: Hacer que Postfix entienda la autenticación de SMTP (Auth-SMTP)

Si permitiéramos que cualquier hijo de vecino usara nuestro servidor de correo, nos convertiríamos inmediatamente en lo que se llama un **open relay**. Los **spammers** no tardarían en encontrarlo y les estaríamos ayudando a seguir inundándonos de correo basura. Es por esto que necesitamos que los usuarios remotos se autentifiquen para poder usar nuestro servidor. Configurar la autenticación por SMTP es muy sencillo, el único problema al que nos enfrentamos es que Debian ejecuta Postfix en un chroot por defecto.

Indicarle a Postfix que use SASL con MySQL:

Creamos el archivo de configuración: /etc/postfix/sasl/smtpd.conf

```
pwcheck_method: auxprop
auxprop_plugin: sql
mech_list: plain login cram-md5 digest-md5
sql_engine: mysql
sql_hostnames: 127.0.0.1
sql_user: proveedor_admin
sql_passwd: tu-password
sql_database: proveedor
sql_select: select password from usuarios where email='%u@%r'
```

Ya sabemos sobre los permisos:

```
chown root:postfix /etc/postfix/sasl/smtpd.conf
chmod u=rw,g=r,o= /etc/postfix/sasl/smtpd.conf
```



Usar TLS para encriptar el tráfico de SMTP:

Un paso importante es encriptar la sesión de SMTP, de lo contrario en nombre de usuario y la contraseña viajarán de un modo muy inseguro si el usuario elige emplear una autenticación de texto plano. Por lo que sugiero encriptar la comunicación usando **TLS** (Transport Layer Security), que explicándolo brevemente, emplea **SSL** (Secure Socket Layer) que encripta la conexión de correo entre el servidor y el usuario.

Primero necesitaremos un certificado de SSL. Si tu no deseas pagarle a tu *trustcenter* favorito, podemos emplear uno firmado por nosotros mismos. (Nota personal: me pregunto... ¿cómo es que pagando por algo se vuelve más seguro?) El único problema es que el cliente de correo no sabe sobre nuestro **CA** (Certificate Authority) y le enviará un mensaje de advertencia a nuestros usuarios. Así que debemos pedirles a nuestros usuarios que lo ignoren, o bien, permitirles que instalen el certificado en sus computadoras. (Outlook no va a permitir que esto sea ignorado, así que deberán instalar el certificado para que esto funcione). Para un certificado válido por 10 años para el dominio smtp.dominio.com:

```
openssl req -new -outform PEM -out /etc/postfix/smtpd.cert -newkey rsa:2048 \  
-nodes -keyout /etc/postfix/smtpd.key -keyform PEM -days 3650 -x509
```

Tendremos que contestar algunas preguntas, prestando una atención especial a *Common Name* que deberá ser el nombre del servidor de correo:

```
Country Name (2 letter code) [AU]MX  
State or Province Name (full name) [Some-State]Coahuila  
Locality Name (eg, city) []Torreon  
Organization Name (eg, company) [Internet Widgits Pty Ltd]Faller de correo electronico  
Organizational Unit Name (eg, section) []UAC  
Common Name (eg, YOUR name) []servidor.uadec.mx  
Email Address []:postmaster@servidor.uadec.mx
```

Obtendremos 2 archivos: **smtpd.key** (la llave privada) y **smtpd.cert** (el certificado).

```
chmod u=rw,g=r,o= /etc/postfix/smtpd.key  
chown root:postfix /etc/postfix/smtpd.key
```



Paso 8: Configurar el servicio de POP3/IMAP

Ya hemos avanzado mucho en la configuración, sin embargo nuestros usuarios estarán muy enojados ya que todavía no puedes acceder a sus buzones de correo... por lo que es tiempo de configurar el servicio de POP3 y/o IMAP. Primero hay que editar el archivo `/etc/courier/authdaemonrc` y establecer la directiva **authmodulelist** a **authmysql**.

```
authmodulelist="authmysql"
```

Ahora definiremos los campos de la tabla de MySQL en `/etc/courier/authmysqlrc`:

```
MYSQL_SERVER localhost
MYSQL_USERNAME proveedor_admin
MYSQL_PASSWORD tu-password
MYSQL_PORT 0
MYSQL_DATABASE proveedor
MYSQL_USER_TABLE usuarios
#MYSQL_CRYPT_PWFIELD (comment this out)
MYSQL_CLEAR_PWFIELD password
MYSQL_UID_FIELD 5000
MYSQL_GID_FIELD 5000
MYSQL_LOGIN_FIELD email
MYSQL_HOME_FIELD "/home/vmail"
#MYSQL_NAME_FIELD (comment this out)
MYSQL_MAILDIR_FIELD CONCAT(SUBSTRING_INDEX(email,'@',-1),'/',SUBSTRING_INDEX(email,'@',1),'/')
```

Hay que tener cuidado de no insertar espacios en blanco después de cada opción, `authmysql` es un poco quisquilloso. Ahora reiniciamos el demonio de autenticación: **`/etc/init.d/courier-authdaemon restart`**

Una prueba express:

Abriremos un telnet para ver si contesta nuestro servidor de POP3.

```
telnet localhost pop3
```

Si nos responde con “+OK Hello there”... nuestros usuarios ya deben estar contentos.



Paso 9: Probar nuestra configuración.

¡Felicidades! Nuestra parte de configuración básica ha terminado. ¿Alguien picha las gordas?
Ahora agregaremos a nuestra base de datos un dominio y un usuario para probar el servicio.

```
INSERT INTO dominios (`dominio`) VALUES ('virtual.test');  
INSERT INTO usuarios (`email`,`password`) VALUES ('usuario@virtual.test','secreto');
```

Establecemos un telnet a nuestro servidor de SMTP:

```
telnet localhost smtp
```

Servidor

```
220 miservidor ESMTP Postfix (Debian/GNU)
```

Tu

```
ehlo uadec.mx
```

```
250-mailtest  
250-PIPELINING  
250-SIZE 10240000  
250-VRFY  
250-ETRN  
250-STARTTLS  
250-AUTH LOGIN PLAIN DIGEST-MD5 CRAM-MD5  
250-AUTH=LOGIN PLAIN DIGEST-MD5 CRAM-MD5  
250 8BITMIME
```

```
mail  
from:<alejandro@barcena.com.mx>
```

```
250 Ok
```

```
rcpt to:<usuario@virtual.test>
```



```
250 0k

                Servidor                                Tu

250 0k

                data

354 End data with <CR><LF>.<CR><LF></LF></CR></LF></CR>

                Este es un correo de prueba.
                .

250 0k: queued as ABC1D1C123

                quit

221 BYE
```



Paso 10: A poblar la base de datos.

Ya que sabemos que el servicio esta funcionando, agregaremos dominios y usuarios.

Para cada dominio nuevo:

Agregarlo a la tabla de dominios.

Para cada usuario nuevo:

Agregarlo a la tabla de usuarios.

Para cada forwarding nuevo:

Agregarlo a la tabla de forwardings. Si es un destinatario múltiple, hay que separar los correos con comas. Por ejemplo:

origen	destino	Efecto
postmaster@mi.dominio	alejandro@mi.dominio	Redirecciona el correo de postmaster a alejandro.
@mi.dominio	m77@mi.dominio	Le envía todo el correo enviado al dominio mi.dominio a m77. Esto no aplica para las cuentas que se encuentren en la tabla de usuarios. Por ejemplo: webmaster@mi.dominio.
@mi.dominio	@otro.dominio	Esto redirecciona todo el dominio. Cada dirección de correo de mi.dominio será dirigida al mismo usuario de otro.dominio. Por lo que m77@mi.dominio será redirigido a m77@otro.dominio .
gulag@mi.dominio	alejandro@mi.dominio ,m77@gmail.com	Todo el correo que le llegue a gulag@mi.dominio será enviado a las 2 cuentas. Cada usuario obtiene una copia.



Paso 11: Analizar el correo para encontrar spam y virus.

Introducción a AmaViS:

Las 2 cosas más molestas, incluso más que las cadenas, al utilizar el correo electrónico son el spam y los virus. Afortunadamente podemos combatir ambos con un software llamado AMaViS (A Mail Virus Scanner). Es una interfase entre Postfix, Spamassassin (famoso por su capacidad de filtrado bayesiana) y cualquier antivirus, como Clamav, que es libre.

Configuración de AMaViS:

Debemos ocuparnos de lo siguiente en `/etc/amavis/amavisd.conf`:

Opción	Significado
<code>\$mydomain = 'midominio.org';</code>	Si no vamos a usar dominios locales, la establecemos como localhost.
<code>@bypass_virus_checks_acl = qw(.);</code>	Si esta opción esta comentada (tiene un # al inicio) entonces la revisión de virus esta habilitada.
<code>@bypass_spam_checks_acl = qw(.);</code>	Si esta opción esta comentada (tiene un # al inicio) entonces la revisión de spam esta habilitada.
<code>@lookup_sql_dsn = (['DBI:mysql:proveedor', 'proveedor_admin', 'tu-password']);</code>	Aquí le indicamos a AMaViS cómo acceder a nuestra base de datos, ya que va a necesitar la tabla de dominios.
<code>\$sql_select_policy = 'SELECT "Y" as local FROM dominioss WHERE CONCAT("@",dominio) IN (%k)';</code>	AMaViS necesita saber qué dominios están hospedados en el servidor. De esta manera determina si el correo va de salida (enviado por nuestro servidor) o viene llegando (enviado por alguien en internet). El correo de salida no será verificado contra virus o spam.
<code>\$final_virus_destiny = D_DISCARD;</code>	El valor por defecto es BOUNCE. Esto rara vez es útil, ya que el correo ya se encuentra en nuestro servidor, y se tendría que enviar un correo de regreso al remitente. Creanme, el remitente real de un correo infectado por virus jamás (99%) es la persona que aparece en el encabezado de correo.



Opción	Significado
<code>\$final_banned_destiny = D_REJECT;</code>	¿Qué hacer con los correos que tienen extensiones vetadas?
<code>\$final_spam_destiny = D_PASS;</code>	¿Qué hacer con los correos clasificados como spam?
<code>\$sa_tag_level_deflt = -1000;</code>	AMaViS califica cada correo con una calificación de spam. Por lo general, varia entre 5 y 10. Pero sólo mostrará la calificación en el encabezado si ésta es superior a este valor.
<code>\$sa_tag2_level_deflt = 5.0;</code>	Si la calificación es superior a este valor, se añadirá un encabezado: "X-Spam-Status: Yes"
<code>\$sa_kill_level_deflt = 10;</code>	Si la calificación es superior a este valor, AMaViS tomará cartas en el asunto. La acción a emprender se define en <code>\$final_spam_destiny</code> .
<code>\$sa_spam_subject_tag = '***SPAM***';</code>	Si deseas alterar el asunto de los correos catalogados como spam.
<code>@av_scanners = (...</code>	En esta sección puedes configurar uno o más antivirus. Algunos muy comunes vienen preconfigurados, sólo hay que quitarles el # para que dejen de ser comentarios. Para empezar, sugiero sólo utilices el de Clamav y comentes el resto. ¿Te preguntas por qué Clamav aparece 2 veces? Bueno, <code>@av_scanners</code> emplea la versión de demonio (clamd) y <code>@av_scanners_backup</code> emplea la de línea de comandos (clamscan). Como no tenemos acceso a internet, lo dejaremos así; no obstante, es muy recomendable que este habilitada para un mejor funcionamiento del analizador de spam.
<code>\$sa_local_tests_only = 0;</code>	

Si deseas emplear clamd, hay que agregar el usuario clamav al grupo amavis.

```
adduser clamav amavis
```

y reiniciamos el servicio de amavis para activar los cambios (**/etc/init.d/amavis restart**).



Paso 12: Indicarle a Postfix que use AMaViS.

Hay que establecer el filtrado global de contenido en **/etc/postfix/main.cf**:

```
content_filter = amavis:[127.0.0.1]:10024  
receive_override_options = no_address_mappings
```

La opción *content_filter* hace que todo el correo sea enviado a un servicio llamado amavis, el cual definiremos en el archivo **/etc/postfix/master.cf**:

```
amavis unix - - - - 2 smtp  
-o smtp_data_done_timeout=1200  
-o smtp_send_xforward_command=yes  
  
127.0.0.1:10025 inet n - - - smtpd  
-o content_filter=  
-o local_recipient_maps=  
-o relay_recipient_maps=  
-o smtpd_restriction_classes=  
-o smtpd_client_restrictions=  
-o smtpd_helo_restrictions=  
-o smtpd_sender_restrictions=  
-o smtpd_recipient_restrictions=permit_mynetworks,reject  
-o mynetworks=127.0.0.0/8  
-o strict_rfc821_envelopes=yes  
-o receive_override_options=no_unknown_recipient_checks,no_header_body_checks
```