Postfix: Guía de Configuración

Por Juan Carlos Inostroza O.(jci@tux.cl)

1.- Que es Postfix

Postfix es un MTA (Mail Transport Agent), escrito originalmente por Wietse Venema, que comenzo siendo una alternativa a Sendmail. Sendmail controla cerca del 70% del movimiento de correo electronico en internet. El problema que Sendmail es demasiado complicado para configurar. Peor aun si se quiere hacer cosas mas alla de una configuracion simple. Postfix es una de las alternativas, como lo son tambien Qmail y Zmailer.

1.1.- Instalando Postfix

Como cualquier programa que corra en GNU/Linux, o se precie de serlo, esta la opcion de instalarlo via codigos fuentes. En sistemas RedHat, preferente mas facil, se puede ahorrar el (tedioso) trabajo de compilar e instalar, ya que archivos en formato RPM ayudan bastante a hacer la vida mas facil.

Para instalar Postfix desde codigo fuente:

- Bajar el ultimo snapshot desde http://www.postfix.org
- Agregar un usuario sin privilegios, ojala llamado postfix.
- Descomprimir los fuentes en algun directorio, con el usuario sin privilegios
- El paso tipico, ./configure, make
- Como superusuario (root), hacer make install

Usando RPMs, es algo mas simple =)

- Bajar el ultimo snapshot en RPM
- Como superusuario, ejecutar rpm -ihv postfix-XXXXXXX.rpm

Para evitar referencias malas de directorios, prefiero guiarme por la estructura de instalacion de programas RedHat.

1.2- Entendiendo unas pocas cosas antes

Antes de comenzar a configurar, se debe entender ciertas cosas en la configuracion de Postfix.

Puede usarse el par parametro = valor en el archivo de configuracion. Asi tambien, $parametro se puede pasar como valor a otra variable. Esto es mas para ahorrar tipeos =).

Por ejemplo

```
parametro1 = valor1
parametro2 = $parametro1
```
(que es equivalente a parametro2 = valor1).

Algunas direcciones IP pueden encubrirse para evitar que el Postfix haga peticiones al DNS. Esto es usar naked ip addresses. Para ingresar una direccion IP desnuda y evitar que Postfix resuelva el nombre, se usan los corchetes cuadrados [ ].

```
parametro1 = [192.168.5.15]
```

2.- Configurando Postfix

En `/etc/postfix` se encuentran los archivos de configuracion. Estos son dos: **main.cf** y **master.cf**. **main.cf** es el archivo principal, donde reside el corazon del funcionamiento de Postfix. El archivo **master.cf** es un tipo de archivo tipo inetd.conf, donde los distintos programas de Postfix ven su forma de funcionar. Advertencia: el archivo **master.cf** es el mas delicado y complicado. No modificar si no se sabe que se esta haciendo.

2.1- Primeras cosas: hostname, domain, networks

Lo primero es ingresar a main.cf el nombre del host (**$myhostname**), el dominio (**$mydomain**). Networks se usa para indicarle a Postfix que maquinas, distinguidas por IP o direccion son consideradas locales y pueden usar al servidor de correo (**MAILSVR** en adelante) para envios. Networks puede ser una colección de Ips o una clase completa.

Por ejemplo, nuestro MAILSVR se llama **foo1.bar.com** y el dominio se llama bar.com, modificamos las siguientes lineas en main.cf:

```
myhostname = foo1.bar.com
mydomain=bar.com
```

Lo siguiente es indicar si se quiere enmascarar las direcciones de correo. Esto es para que usuarios que pertenezcan a distintos subdominios aparezcan

que son enviados desde un mismo dominio (bar.com). Esto es solamente usado si se tiene un dominio con distintas maquinas. El valor por defecto es **$mydomain**

```
myorigin = $myhostname
myorigin = $mydomain
```

El parametro **mydestination** especifica que dominios entregar localmente, en vez de enviarlo a otras maquinas. El valor por defecto es entregarlo al mismo MAILSVR. Puede especificarse ninguno o varios dominios y tablas de lookup con separaciones por espacios o comas.

```
mydestination = $myhostname localhost.$mydomain
mydestination = $myhostname $mydomain
mydestination = $myhostname www.$mydomain ftp.$mydomain
```

Precaucion : siempre agregar $myhostname y localhost.$mydomain para evitar loops de entregas de correo.

Generalmente los mails rebotan. Algunas veces, algunos mails que no rebotan simplemente no son entregados. Para ello, existe un usuario, postmaster que a quien llegan los correos no entregados. Generalmente llegan aquellos correos con un gran debug con errores.

Para saber por que no se entrego correo, la directiva **notify_classes** indica el nivel de error a notificar. Los valores que puede tener son

- **bounce** : envia a postmaster copias de los correos no entregados, pero estas copias son modificadas para proteger la privacidad del mensaje.
- **2bounce** : envia dos copias del mail que rebota
- **policy**: informa a postmaster las peticiones rechazadas por politicas UCE de otros servidores. Llega una copia de la transaccion
- **protocol** : informa a postmaster cualquier error de protocolos, cliente o servidor, o intentos de algun cliente de ejecutar comandos no implementados. Se recibe una copia de la transaccion completa
- **resource** : informa a postmaster de los mail no entregados por algun problema de recursos (errores read/write, queue, etc)
- **software** : informa a postmaster de problemas de software

Cualquiera de estas opciones son combinables.

```
notify_clases = resource, software
```

La directiva **mynetworks** permite que una red se considere local para Postfix. Esto es para distinguir entre maquinas conocidas de las extrañas (fuera de la red). Las maquinas consideradas como locales pueden usar a MAILSVR como un open relay incluso.

Puede configurarse una clase A, B o C, dependiendo de la cantidad de maquinas.

```
mynetworks = 192.168.1.0/28, 127.0.0.0/8
```

El parametro **inet_interfaces** indica que interfaces de red debe escuchar MAILSVR. Los mails enviados a user@direccion_de_red seran entregados localmente, y direccionados a un dominio que este listado en **$mydestination**.

El valor por defecto es all (todas las interfaces). Si se tienen interfaces virtuales, se debe indicar cuales de las interfaces escuchar.

```
inet_interfaces = all
inet_interfaces = virtual.host.name # dominio virtual
inet_interfaces = $myhostname localhost.$mydomain # mailer no virtual
```

La opcion **relay_domains** restringe los dominios donde los clientes usan a MAILSVR para enviar correo (relay) o que destinos va a servir MAILSVR. Por defecto, Postfix relega (relay) correo a: clientes confiables que su direccion esta en **$mynetworks** clientes confiables que esten en **$relay_domains** o algun subdominio clientes no confiables los cuales el destino sea **$relay_domains** o algun subdominio de el.

Postfix ademas acepta correo para:

destinos que esten en $inet_interfaces
destinos que esten en $mydestination
destinos que esten en $virtual_maps

```
relay_domains = $mydestination
```

2.2- Opciones Adicionales

La opcion **queue_directory** especifica el lugar de la cola de Postfix. Es tambien el directorio raiz de los demonios de Postfix (que corren chrooted).

```
queue_directory = /var/spool/postfix
```

command_directory y **daemon_directory** contienen la ruta donde estan los comandos de Postfix y los demonios, respectivamente

```
command_directory=/usr/sbin
daemon_directory=/usr/libexec/postfix
```

mail_owner indica el usuario que es propietario de la cola de Postfix. Especificar un usuario que no comparta un grupo con otras cuentas y que no posea otros archivos o procesos en la misma maquina. O sea, ni nobody ni daemon. Se debe usar un usuario dedicado.

La instalacion de Postfix crea el usuario y el grupo postfix. Seria logico usarlo para mail_owner. =)

```
mail_owner = postfix
```

default_privs indica los privilegios por defecto del agente de entrega de correo para ejecutar un comando o abrir un archivo. NO especificar un usuario con privilegios o el usuario postfix. Generalmente se usa nobody.

```
default_privs = nobody
```

La opcion **mail_spool_directory** indica el directorio donde los mailboxes son almacenados, "alla UNIX".

```
mail_spool_directory = /var/mail
```

2.3- Uso de Tablas Lookup : HASH y REGEXP para filtros

Hay algunas de las opciones de Postfix que requieren saber que son tablas lookup (lookup tables). Son tablas, contenidas en un formato que Postfix define como diccionario.

Estos diccionarios pueden ser de la siguiente forma:

- `regexp:/file/name`
- `pcre:/file/name`
- `hash:/file/name`
- `mysql:/file/name.cf`

Existen mas, pero estos son los mas conocidos.

Pcre y Mysql requieren que Postfix se recompile con este soporte.

Las tablas (maptype) son archivos (mapname) con separacion por comas o espacios (o una dbase) donde se crea una expresion regular y un resultado. Los resultados pueden ser los siguientes:

```
OK : permitida la accion
REJECT : accion rechazada
RELAY : permite relay
ERRORNO razon : un numero de error y una razon del error
```

Algunos de los numeros de errores que se deben devolver son:

```
450 : Unknown address - DNS error
554 : UCE restriction
504 : Non-FQDN sender
```

Regexp y PCRE (Perl Common Regular Expressions) son muy similares en su uso.

Una tabla regexp puede ser, por ejemplo:

```
/^amigo@dominio1.com.*/     OK
/^postmaster@.*$/           RELAY
/[aA][cC]v@subdominio.*$/   REJECT
/hahaha/                    550 Esto es un vil SPAM
```

Una tabla HASH es de la siguiente forma:

```
(patron) (separacion) (accion)
```

y recuerda bastante la generacion de /etc/mail/access de Sendmail.

Una separacion puede ser un blanco (espacio).
Un patron puede venir de una base de datos, tabla NIS, SQL etc. de la siguiente forma:

```
usuario@dominio
nombre.de.dominio
usuario@
numero.ip.de.cliente, numero.ip.de , numero.ip , numero :
```
direcciones de red.

Las acciones son de la siguiente forma
`[45]XX text` : rechaza el mensaje que aparezca en el patron y responde con el codigo y con el texto indicado.
`REJECT` : simplemente rechaza. Un error generico es desplegado
`OK` : acepta.

Un ejemplo de una tabla HASH puede ser:

```
midominio1.com              OK
spammer@spammersunited.com  REJECT
spammer@                    REJECT
192.168.1.99                450 Unresolved
```

Despues de tener la tabla HASH hay que crear el archivo .db . Esto se hace con POSTMAP.

```
# postmap /etc/postfix/access
```

2.4- UCE - Unsolicited Commercial Email a.k.a. SPAM

Para alegria nuestra, Postfix viene configurado para no ser un Open Relay.

Ahora que tenemos instalada la maquina, queremos evitar que la maquina sea usada para hacer SPAM. Las opciones que tenemos son las siguientes:

- Permitir a las maquinas locales poder hacer relay
- No permitir a las maquinas externas hacer relay
- Permitir a los clientes confiables hacer relay
- Denegar a los clientes no confiables a hacer relay

Para lograr estos objetivos, tenemos las siguientes herramientas:

- filtrado de cabeceras
- filtrado de contenido
- restricciones de hostnames/direcciones
- requerir el comando HELO
- restringir el comando HELO
- requerir direcciones de correo RFC821
- restricciones de emisores
- restricciones de destinatarios
- restricciones de ETRN
- restricciones genericas
- parametros UCE

Filtrado de cabeceras

La directiva **header_checks** restringe que cabeceras se permiten en un mensaje. Debe estar acompañada por una tabla lookup, generalmente una expresion regular.

```
header_checks = regexp:/etc/postfix/header_check
```

y creamos el archivo /etc/postfix/header_check con

```
/^to: *amigo@publico\.com$/       REJECT
```

Esto evita que direcciones del tipo miamigo@publico.com, noamigo@publico.com, amigo@publico.com puedan o enviar o recibir o usar el servidor.

Filtrado de contenido

La directiva **body_checks** permite hacer un filtrado del cuerpo del mensaje. Para su uso es necesario una tabla lookup, generalmente (y para facilidad) se usan tablas regexp. Por ejemplo, teniendo un archivo /etc/postfix/body_checks con lo siguiente:

```
/keyword/       REJECT
```

Esto evitara que alguien envie un correo que contenga en alguna parte del BODY del mensaje la palabra "keyword". Por defecto, las tablas regexp NO son sensitivas a las mayusculas.

Si se necesita que sea sensitivo a las mayusculas, se puede agregar "/i".

Restricciones de hostname/direccion

El parametro **smtpd_client_restriction** restringe que clientes aceptar.
Valores posibles :
check_client_access tabla:/file/name: generalmente se usa una tabla de tipo HASH. Contiene la forma de acceso del cliente al conectarse a MAILSVR.
reject_unknown_client : rechaza cualquier conexión desconocida o que no tenga un registro PTR en el DNS.
permit_mynetworks : Permitir las peticiones que provengan de cualquier IP que este en **$mynetworks**
reject_maps_rbl : rechaza la peticion si la direccion de red esta en **$maps_rbl_domains**.
permit
reject
reject_unauth_pipelining

```
smtpd_client_restriction = check_client_access hash:/etc/postfix/access, reject_maps_rbl, permit_mynetworks, reject_unknown_client, reject
```

Requerir el Comando HELO

El parametro **smtpd_helo_required** determina si los clientes DEBEN enviar un HELO (o EHLO) al servidor antes de comenzar una sesion SMTP. Al requerir esto, se detiene gran parte del software usado para UCE.

Posee dos valores, **yes** y **no**. Por defecto, esta el valor **no**.

Restriccion de host para comando HELO

smtpd_helo_restrictions restringe aquellos hosts que pueden enviar el comando HELO.

Las restricciones posibles son:
reject_invalid_hostname : rechaza la peticion cuando HELO es enviado con una mala sintaxis de host. Retorna el error 501 por defecto.
permit_naked_ip_address : permite que la peticion contenga un numero ip sin uso de corchetes en vez de un hostname. Desafortunadamente, muchos programas UCE funcionan con este procedimiento
reject_unknown_hostname : equivalente a reject_unknown_client

reject_non_fqdn_hostname rechaza cuando el hostname en el comando HELO no esta de la forma especificada en el RFC821
check_helo_access maptype:mapname : exactamente igual a **check_client_access**
reject_maps_rbl reject_unknown_client : ya explicado
permit_mynetworks : ya explicado
check_client_access maptype:mapname : ya explicado
permit
reject
reject_unauth_pipelining

```
smtpd_helo_restrictions = permit_mynetworks, reject_invalid_hostname
```

RFC821

El parametro strict_rfc821_envelopes controla que tan tolerante es Postfix con respecto a las direcciones dadas en MAIL FROM o RCPT TO. Desafortunadamente, Sendmail permite mucho de los comportamientos no standard, asi que un software UCE explota este error. Siendo estricto con la forma del RFC821 no solo detiene correo no solicitado, sino que bloquea correo enviado por aplicaciónes mal desarrolladas.

Valores posibles son **yes** y **no**.

Restricciones del emisor

El parametro **smtpd_sender_restriction** restringe las direcciones que el sistema acepta cuando se ingresa el comando MAIL FROM.

Parametros:

reject_unknown_sender_domain : rechaza una peticion cuando la direccion de correo del emisor no posee un registro A o MX en un DNS.
check_sender_access maptype:mapname : exactamente igual a check_helo_access
reject_non_fqdn_sender : ya explicado
permit_naked_ip_address
reject_invalid_hostname
reject_unknown_hostname
reject_non_fqdn_hostname
check_helo_access maptype:mapname
reject_maps_rbl
reject_unknown_client
permit_mynetworks
check_client_access maptype:mapname
permit
reject
reject_unauth_pipelining

Restricciones de destinatarios

El parametro **smtpd_recipient_restrictions** restringe los recipientes o destinatarios que acepta MAILSVR cuando se ejecuta RCPT TO.

Valores:

check_relay_domains : permite la peticion si la direccion del cliente esta

- en **$relay_domains** o en algun subdominio.
- La direccion resuelta (DNS) corresponde a una maquina que pertenece a **$relay_domains**
- cualquier maquina que este en **$mydestination**, **$inet_interfaces** o **$virtual_maps**

permit_auth_destination : ignora el hostname cliente. Permite la peticion si:

- la direccion resuelta (DNS) esta en **$relay_domains** o un subdominio de el y la direccion no contiene rutas especificas (usuario@otrolado@dominio)
- cualquier maquina que este en **$mydestination**, **$inet_interfaces** o **$virtual_maps**

reject_unauth_destination : Ignora el host cliente. Rechaza la peticion si NO concuerda con lo siguiente:

- la direccion resuelta esta en **$relay_domains** o un subdominio, y la direccion no contiene ruteo especifico de usuario (usuario@maquina@dominio)
- Postfix tiene la ultima palabra: cualquier destino que este en **$mydestination**, **$inet_interfaces** o **$virtual_maps**

permit_mx_backup : permite la peticion cuando el sistema de correo local resuelve un host MX. Esto incluye el caso que el sistema de correo local es el destino final. Sin embargo, el MAILSVR no enviara correo con rutas especificas de usuario (usuario@maquina@dominio)

check_recipient_address maptype:mapname : funciona exactamente igual que check_client_access.

reject_unknown_recipient_domain : rechaza la peticion cuando la direccion de correo si no posee un registro A o MX.

reject_non_fqnd_recipient : igual que reject_non_fqnd_sender
reject_unknown_sender_domain

reject_non_fqnd_sender
check_sender_access maptype:mapname : igual que check_client_access
permit_naked_ip_address
reject_invalid_hostname
reject_unknown_hostname
reject_non_fqdn_hostname
check_helo_access maptype:mapname
reject_maps_rbl
reject_unknown_client
permit_mynetworks
check_client_access maptype:mapname
permit
reject
reject_unauth_pipelining

Restriccion ETRN

El parametro **smtpd_etrn_restrictions** restringe los dominios que pueden ejecutar comandos ETRN.

Valores:

check_etrn_access maptype:mapname : igual que check_client_access
permit_naked_ip_address
reject_invalid_hostname
reject_unknown_hostname
check_helo_access maptype:mapname
reject_maps_rbl
reject_unknown_client
permit_mynetworks
check_client_access maptype:mapname
permit
reject
reject_unauth_pipelining

Restricciones genericas

permit : permite la peticion
reject : rechaza la peticion
reject_unauth_pipelining : rechaza la peticion cuando el cliente envia comandos SMTP antes de tiempo sin saber que postfix soporta command pipelining. Esto detiene a programas de correo masivo que usan command pipelining para acelerar entregas.

Restricciones adicionales UCE

maps_rbl_domains : controla el comportamiento de reject_maps_rbl que aparece como parte de una lista de nombres/direcciones de un cliente. Estan desabilitadas por defecto.

```
maps_rbl_domains = rbl.maps.vix.com, dul.maps.vix.com
```

relay_domains : controla el comportamiento de **check_relay_domains**, **reject_unauth_destination** y **permit_auth_destination**.

```
relay_domains = $mydestination
```

3.- control de recursos

Postfix, como todo MTA, corre con recursos limitados. Para evitar sobrecargas de memoria, disco y procesos, existen distintas formas de afinar el funcionamiento de Postfix.

line_length_limit : (por defecto: 2048 bytes) - que tan larga una linea de texto puede ser antes de ser cortada en partes. Las lineas largas son reconstruidas en el momento de la entrega

header_size_limit : (def: 102400 bytes) : cuanto texto puede incluirse en una cabecera. Si una cabecera no cabe en $header_size_limit se sobrecarga hacie el cuerpo del mensaje.

extract_recipient_limit : (def: 10240) : cuantos recipientes podra extraer Postfix de las cabeceras antes de rendirse. Esto reduce el daño de algun programa que intente hacer `sendmail -t`

message_size_limit : (10240000 bytes) : indica el tamaño maximo de un archivo de cola de postfix, incluyendo la informacion del envio (cabeceras, recipientes, etc)

queue_minfree : (sin restriccion) cuantos bytes de espacio libre son necesarios en la cola de sistemas. MAILSVR deja de entregar correo cuando hay espacio insuficiente. No hay un limite por defecto, sin embargo, es buena idea tener un valor de $message_size_limit multiplicado por 10 o 20

bounce_size_limit : (def 50000 bytes) : cuanto de un mensaje no entregado es devuelto a remitente

qmgr_message_recipient_limit (def 10000) : el limite superior de estructuras de direcciones de datos para el administrador de colas. Ademas controla el numero de instancias de otras estructuras de direcciones de datos.

queue_message_active_limit (def 1000) : limite superior de mensajes en la cola de activos.

duplicate_filter_limit : (def 1000) cuantos recipientes recordaran el agente de entregas locales y el demonio de cleanup.

command_time_limit (def 1000 segs) : cuanto tiempo esperara el agente de entregas locales antes de abortar un comando externo

nombre_servicio time_limit (def: $command_line_limit) el limite de tiempo para comandos externos via pipe mailer.

deliver_lock_attemps : (def 5) cuantos intentos de bloquear un archivo antes de rendirse

deliver_lock_delay (def 1 seg) cuanto tiempo esperar entre intentos de bloqueo

stale_lock_time (def 500) que tan antiguo un archivo de bloqueo puede existir antes de ser removido a la fuerza

fork_attemps (def 5 veces) numero de veces para intentar crear un proceso nuevo antes de rendirse

fork_delay (def 1 seg) la espera entre intentos para crear un nuevo proceso

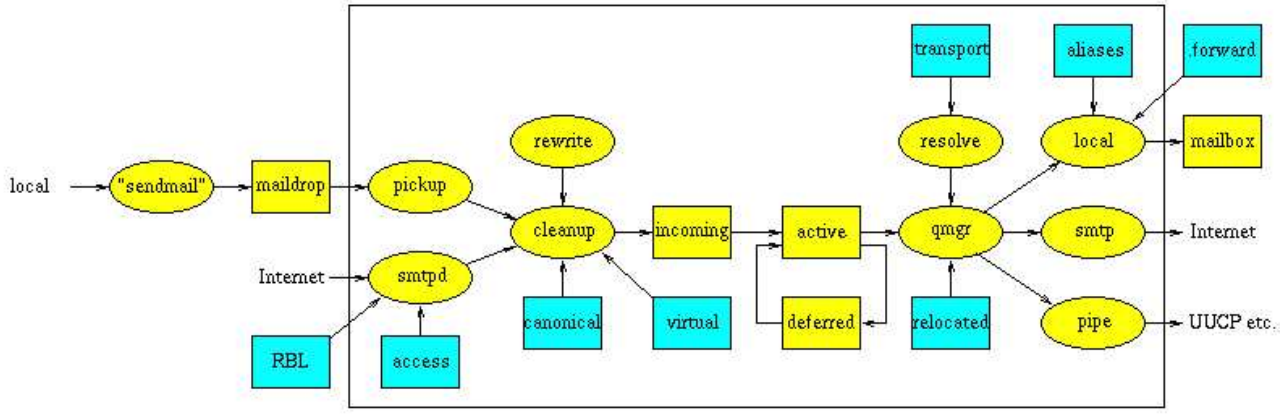
transport_retry_time (def 60 segs) la cantidad de tiempo que el administrador de colas contacte un posible servicio de entregas de Postfix que esta difunto

4.- La ultima frontera : master.cf

Advertencia!
Este archivo es bastante delicado. Advertidos =)

Este archivo contiene la configuracion para los procesos maestros de Postfix. Cada linea indica como un componente debe correr.

Para hacer una explicacion mas racional, voy a incluir el dibujito de como funciona Postfix



Los elipses amarillos son programas de correo.
Las cajas amarillas son colas de correo o archivos.
Las cajas azules son tablas lookup.

Los programas que estan en la caja grande son programas que controla el demonio maestro de postfix, tambien llamado **master**.

El proceso **master** es el proceso residente que ejecuta los demonios de Postfix cuando se necesitan: demonios para enviar o recibir mensajes localmente, etc. Estos demonios son creados por peticion configurados para correr un numero maximo de veces.

Los demonios de Postfix voluntariamente terminan despues de estar idle por una cierta cantidad de tiempo o despues de haber hecho una cierta cantidad de peticiones. La excepcion de esta regla es el administrador de colas, el queue manager (**qmgr**).

El demonio **qmgr** espera la llegada de correo entrante y se preocupa de su entrega via proceso de entrega Postfix. La estrategia de envio y reenvio es delegada a **trivial-rewrite**, tambien ejecutado desde el demonio **master**.

Cuando se inicia una conexión por red a MAILSVR, se inicia un proceso de compatibilidad con Sendmail llamado sendmail. Por defecto, sendmail lee un mensaje de la entrada standard hasta un EOF o hasta que exista una linea con un . (punto) y lo arregla para su envio. Sendmail trata de crear una archivo de cola, pasandolo al directorio **maildrop**. Si ese directorio no tiene permisos de ejecucion para el mundo, el mensaje es pipeado a traves del comando **postdrop**, que se espera se ejecute con privilegios apropiados.

El demonio **pickup** espera la señal que nuevo correo ha llegado al directorio **maildrop** y con este directorio alimenta a **cleanup**.

El demonio **cleanup** procesa el correo entrante, lo inserta en la cola de correo entrante e informa a **qmgr** de su llegada.

Este demonio siempre :

- inserta headers que faltan: **From:, To:, Message-Id: y Date:**

- Extrae las direcciones desde **To:, Cc: y Bcc:** cuando ningun recipiento es especificado en el mensaje
- Transforma el mensaje y las cabeceras al formato usuario@full-dominio que es la forma que otros programas de Postfix entiendan. Esta tarea se delega a **trivial-rewrite**.
- Elimina direcciones duplicadas

El demonio **trivial-rewrite** procesa dos tipos de peticion de cliente:
reescribir : rescribe una direccion a su forma estandar. Por defecto, trivial-rewrite agrega informacion local a direcciones no calificadas, convierte de swap bang path a la forma de dominio y corta cualquier direccion de forma usuario@otrolado@dominio. **resolver** : resurlve una direccion a una tripleta de tipo transporte, netxhop y recipiente. El sentido del resultado es:

- transporte : el agente de entrega. Es el primer campo de master.cf (servicio)
- netxhop : el host a quien enviar el correo. Para entregas locales es un string en blanco
- recipiente: el recipiente del correo que es pasado a netxhop

Este demonio ademas distingue si el correo es local o no local. La forma de afinarlo es atraves de una tabla de transporte.

El demonio **bounce** mantiene los logs de los mensajes que tienen estado de no entregado. Cada archivo de log tiene el nombre de archivo del archivo de la cola de correo que le corresponde y es almacenado en un subdirectorio de la cola, con nombre igual al servicio en master.cf, generalmente **bounce o defer**.

Este demonio procesa dos tipos de peticiones:

- agregar un registro de estado de recipiente en un archivo de log por mensaje
- colocar un mensaje de rebote, junto con una copia del log y del correspondiente mensaje. Si el rebote es exitosamente efectuado, el archivo de log es borrado.

El software hace su mejor esfuerzo para notificar al emisor del correo que hubo un problema. Una notificacion es enviada incluso cuando el archivo de log o el mensaje original no pueden leerse.

Opcionalmente, un cliente puede pedir que un archivo de log por mensaje sea borrado cuando la operación falla. Esto es usado por los clientes que no pueden reintentar una transaccion por ellos mismos y que dependen de la logica de reintento de los mismos clientes de correo. En español, en el software que reenvie el correo de nuevo. =)

El demonio **local** procesa las peticiones de entrega desde **qmgr** para entregar correo a recipientes locales. Cada peticion necesita un archivo de cola, una direccion de emisor, un dominio o host a quien entregar y uno o mas recipientes.
Ademas, **local** actualiza los archivos de la cola y marca los recipientes como terminados o informa a **qmgr** que la entrega debe ser reintentada en otro momento. Los problemas de entrega son enviados a los demonios **bounce o defer**.

El demonio **pipe** procesa peticiones desde qmgr para ejecutar comandos externos. Cada peticion de entrega requiere un archivo de cola, una direccion de emisor, un dominio o host a quien entregar correo y uno o mas destinatarios. Tiene ademas las mismas funciones de actualizacion de la cola que **local**.

El proceso **smtp** procesa las entregas de mensaje desde **qmgr**. Funciona exactamente igual que **local** y **pipe**. La diferencia esta que smtp busca una lista de direcciones de hosts para intercambio de correo. Ordena la lista por preferencia y se conecta a cada una, listada hasta que encuentra un servidor que responda.

Cuando el dominio o el host esta especificado separado por espacios o comas, **smtp** repite el proceso anterior para todos los destinos hasta que encuentra un servidor que responda.

El demonio **showq** reporta estado de la cola de correo. Emula el comando **mailq**.

Volviendo entonces a **master.cf** =)

Este archivo, como dije, contiene la configuracion de los demonios internos de Postfix. Indica el servicio, el tipo de servicio, privado, no-privilegiado, chrooted, wakeups, maxima cantidad de procesos, el comando y los argumentos.

El tipo de servicio puede ser un demonio, una accion, un programa interno o un par de tipo HOST:PORT. Por ejemplo, smtp (demonio), submission (accion), flush (programa) o localhost:2025

Los tipos de servicio indica la forma de transporte o comunicación entre los programas o demonios. Puede ser inet como sockets internos, unix para sockets de tipo UNIX o fifo para indicar pipes.

El campo privado indica si el acceso esta restringido al sistema de correo. Por defecto, son servicios privados. Sockets inet no pueden ser privados.

El campo no-privilegiado indica si el servicio debe correr con privilegios de root o con el setuid/setgid de **$mail_owner**.

El campo chrooted indica si el servicio debe correr chrooted en la cola de correos. Hasta el momento, todos los demonios pueden correr chrooted, excepto por los demonios **pipe** y **local**.

El campo wakeups indica cuanto tiempo de espera debe haber antes de despertar un proceso en segundos. Un ? Indica que los eventos de wakeup deben ser enviados SOLAMENTE a servicios actualmente en uso. Con un 0 se indica que no wakeup. Hasta el momento, solo **pickup**, **qmgr** y **flush** necesitan un tiempo wakeup.

El campo **max_procs** indica el numero maximo de proceso que puede ejecutar el servicio al mismo tiempo. Lo usual es usar un numero finito o

$default_process_limit

El campo comando y argumentos indican el comando y los argumentos a ejecutar. La linea de comandos es RELATIVA al directorio donde residen los programas de Postfix o con la directiva **$program_directory.**

Algunos argumentos utiles:
-v : verbose logging
-D : symbolic debugging

El resto de los argumentos de los programas estan en las man pages de cada uno de los comandos.