Firewalls

Dispositivo hardware o software que filtra el trafico IP/TCP/UDP/ICMP… entre redes decidiendo que paquete pasa, se modifica, se convierte o se descarta cuya misión es proteger una red de otra a la que esta expuesta, es decir garantizar la seguridad de nuestro equipo ante los peligros cibernéticos de la red. P ej. Nuestra red local de Internet.

¡Ojo! Un cortafuegos es inútil si el sistema en el que está instalado es vulnerable a ataques externos. Es necesario desactivar los servicios innecesarios incluso es recomendable que la máquina que tenga el firewall realice únicamente esa función. Mirando xinetd podremos ver los servicios que se inician cuando se requieren. Por ejemplo se debe desactivar cualquier eco, discard, daytime, chargen, ftp, gopher, shell, login, exec, talk, ntalk, pop2, pop3, netstat, systat, tftp, boota, finger, cfinger….

Basta con colocar una # al principio de la línea en inetd.conf y enviar un kill –HUP pidinetd lo que reiniciará el equipo sin apagarlo.

Un firewall a nivel de software está compuesto por un conjunto de reglas en las que se examina el origen y el destino de los paquetes tcp/ip. Un posible pseudocódigo…

¿Cómo crear una política de seguridad?

Describa para que es el servicio
Describa el grupo de personas a las que va dirigido el servicio
Describa a que servicio necesita acceder cada grupo
Describa para cada grupo de servicio como se puede mantener seguro el servicio
Redacte un informe en el que se considere violacion cualquier otro tipo de acceso

Política por defecto ACEPTAR
Todo lo que venga de la red local al firewall ACEPTAR
Todo lo que venga de la ip de casa al puerto TCP 22 ACEPTAR
Todo lo que venga de la ip de casa al puerto TCP 1723 ACEPTAR
Todo lo que venga de pc.rediris.es al puerto UDP 123 ACEPTAR
Todo lo que venga de la red local al exterior ENMASCARAR
Todo lo que venga del puerto TCP 1 al 1024 DENEGAR
Todo lo que venga del puerto UDP 1 al 1024 DENEGAR

Hay dos formas de implementar un firewall:

Política ACEPTAR por defecto: en principio todo lo que entra y sale por el firewall se acepta y solo se denegara lo que se diga explícitamente
Política DENEGAR por defecto: todo esta denegado y solo se permitirá pasar por el firewall aquello que se permita explícitamente.

La primera política es muy fácil de implementar aunque un poco peligrosa ya que podemos descuidar algún puerto que tengamos abierto para que alguien se cuele por ahí.

La segunda el firewall se convierte en un muro pero más difícil de configurar y configurar con cuidado todo lo que se va a permitir.

IMPORTANTE : El orden de cómo se introducen las reglas es fundamental ya que el firewall al recibir un paquete chequea las reglas en el orden en que se han introducido, por lo que si por ejemplo si ponemos reglas muy permisivas al principio es posible que algunas reglas posteriores nunca lleguen a ejecutarse.

IPtables
Sistema de firewall vinculado al kernel de linux. El firewall utilizado anteriormente era el ipchains. Ejecutando el comando iptables añadimos, borramos o creamos reglas.

En versiones redhat se trata como un servicio que se puede parar o iniciar.

Cuando llega un paquete el kernel mira si es para el o para otra máquina y consulta las reglas del firewall para decidir que hacer con el. Para los paquetes que van a la propia maquina se aplican las reglas INPUT y OUTPUT y para filtrar paquetes que van a otras redes o maquinas se aplican reglas FORWARD (que son los tres tipos de reglas de filtrado.

Existen 3 tipos de reglas:
Filtrado : INPUT , OUTPUT, FORWARD
Redirecciones de puertos o cambios en las ips de origen y destino : NAT Enmascarar : petición de un PC de la LAN salir al exterior con la IP publica a Internet. PREROUTING , POSTROUTING
Modificación de paquetes : MANGLE

Existe la herramienta iptraf para depurar y comprobar el funcionamiento de iptables. Podemos comprobar si las conexiones TCPIP se llegan a establecer o no.

- Si la maquina desea conectarse flan SYN
- Si la otra acepta envia SYN/ACK
- Se establece la conexión

Añadir una nueva regla a una cadena ya existente –A
Insertar una regla en una posición determinada en una cadena –I
Reemplazar la regla que ocupa una posición determinada en una cadena –R
Eliminar la regla que ocupa una posición determinada en una cadena –D

Mientras se modifica un firewall se puede configurar para que no deje pasar nada en las primeras líneas.
Proteger la propia máquina conectada directamente a Internet por ejemplo con un MODEM. Se puede crear un script con las siguientes reglas.

```
#!/bin/sh
#Flush de reglas
iptables –F
iptables –X
```

```
iptables –Z
iptables –t nat –F

#politica por defecto
iptables –P INPUT ACCEPT
iptables –P OUTPUT ACCEPT
iptables –P FORWARD ACCEPT
iptables –t nat –P PREROUTING ACCEPT
iptables –t nat –P POSTROUTING ACCEPT

#El localhost lo dejamos como esta
/Sbin/iptables –A INPUT –i –lo –j ACCEPT

#A nuestra ip le permitimos todo
iptables –A INPUT –s 1.2.3.4 –j ACCEPT

#permitimos una ip externa entrar para mantener mysql
iptables –A INPUT –s 2.3.4.5 –p tcp --dport 3306 –j ACCEPT

iptables –a input –s 2.3.4.5 –p tcp –dport 20:21 –j ACCEPT

#El servidor web abierto para todos
iptables –A input –p tcp –dport 80 –j ACCEPT

#El resto se cierra
iptables –A INPUT –p tcp –dport 1:1024 –j DROP
iptables –A INPUT –p udp –dport 1:1024 –j DROP
iptables –A INPUT –p tcp –dport 3306 –j DROP
iptables –A INPUT –p tcp –dport 10000 –j DROP
```

Es recomendable hacer un netstat –an o un nmap a nuestra propia maquina para ver los puertos que hay abiertos. Se puede usar iptables –L –n para verificar las reglas aplicadas.

Ejemplo de un firewall para el típico caso de una red local con salida a internet. En este caso es necesario el enmascaramiento de las ips (NAT) con lo que se haria dos veces NAT en el router y en el firewall.

```
#!/bin/sh
#Flush de reglas
iptables –F
iptables –X
iptables –Z
iptables –t nat –F

#politica por defecto
iptables –P INPUT ACCEPT
iptables –P OUTPUT ACCEPT
iptables –P FORWARD ACCEPT
iptables –t nat –P PREROUTING ACCEPT
iptables –t nat –P POSTROUTING ACCEPT

#Redirigir a una ip por ejemplo para usar radmin
iptables –t nat –A PREROUTING –s 1.2.3.4 –i eth0 –p tcp –dport
4899 –j DNAT –to 192.168.0.5:4899
```

```
#El localhost lo dejamos como esta
/Sbin/iptables –A INPUT –i –lo –j ACCEPT

#A nuestra ip le permitimos todo eth0 va al router y eth1 a la LAN
iptables –A INPUT –s 192.168.0.0/24 –i eth1 –j ACCEPT

#Filtrar el acceso a la red con FORWARD
#Aceptamos accesos a la web
iptables –A FORWARD –s 192.168.0.0/24 –i eth1 –p tcp –dport 80 –j ACCEPT
iptables –A FORWARD –s 192.168.0.0/24 –i eth1 –p tcp –dport 443 –j ACCEPT

#Abrir el Puerto SMPT y POP3
iptables –A INPUT –s 0.0.0.0/0 –p tcp –dport 25 –j ACCEPT
iptables –A INPUT –s 0.0.0.0/0 –p tcp –dport 110 –j ACCEPT

#Permitimos red privada virtual
iptables –A INPUT –s 1.2.3.4 –p tcp –dport 1723 –j ACCEPT

#Aceptar consultas al DNS
iptables –A FORWARD –s 192.168.0.0/24 –i eth1 –p tcp –dport 53 –j ACCEPT
iptables –A FORWARD –s 192.168.0.0/24 –i eth1 –p udp –dport 53 –j ACCEPT

#Se deniega el resto
iptables –A FORWARD –s 192.168.0.0/24 –i eth1 –j DROP

#Enmascarar y activar bit forwarding
iptables –t nat –A POSTROUTING –s 192.168.0.0/24 –o eth0 –j MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward

#El resto se cierra rango bien conocidos 0.0.0.0 significa cualquier red
iptables –A INPUT –s 0.0.0.0/0 –p tcp –dport 1:1024 –j DROP
iptables –A INPUT –s 0.0.0.0/0 –p  udp –dport 1:1024 –j DROP

#cerrar el Puerto de gestion webmin
iptables –A INPUT –s 0.0.0.0/0 -p tcp –dport 10000 –j DROP

# Cerrar la red virtual
iptables –A INPUT –s 0.0.0.0/0 –i eth0 –p tcp –dport 1723 –j DROP
```

No es recomendable que el firewall realice ninguna otra función ya que puede poner en peligro la red. Se recomienda poner el servicio en otra maquina y realizar una red DMZ desmilitarizada.

```
#!/bin/sh
#Flush de reglas
iptables –F
iptables –X
```

```
iptables –Z
iptables –t nat –F

#politica por defecto
iptables –P INPUT ACCEPT
iptables –P OUTPUT ACCEPT
iptables –P FORWARD ACCEPT
iptables –t nat –P PREROUTING ACCEPT
iptables –t nat –P POSTROUTING ACCEPT

#El localhost lo dejamos como esta
/Sbin/iptables –A INPUT –i –lo –j ACCEPT

#A nuestra ip le permitimos todo eth0 va al router y eth1 a la
LAN
iptables –A INPUT –s 192.168.0.0/24 –i eth1 –j ACCEPT

#Enmascarar y activar bit forwarding
iptables –t nat –A POSTROUTING –s 192.168.0.0/24 –o eth0 –j
MASQUERADE
echo 1 > /proc/sys/net/ipv4/ip_forward

#Permitir el acceso desde el exterior al puerto 80 de la DMZ
iptables –A FORWARD –d 217.126.241.16 –p tcp –dport 80 –j ACCEPT
iptables –A FORWARD –d 217.126.241.16/30 –j DROP

#Permitir acceso de la DMZ a mysql en la LAN
iptables –A FORWARD –s 217.126.241.16 –d 192.168.0.5 –p tcp –
dport 3306 –j ACCEPT
#en el sentido contrario lo mismo
iptables –A FORWARD –s 192.168.0.5 –d 217.126.241.16 –p tcp –
sport 3306 –j ACCEPT
#Permitir terminal server de la DMZ desde la LAN
iptables –A FORWARD –s 192.168.0.0/24 –d 217.126.241.16 –p tcp –
sport 1024:65535 –dport 4489 –j ACCEPT
# En el otro sentido igual
iptables –A FORWARD –s 217.126.241.16 –d 192.168.0.0/24 –p tcp –
sport 4489 –dport 1024:65535 –j ACCEPT

#Cerramos el acceso de la DMZ a la LAN
iptables –A FORWARD –s 217.126.241.16 –d 192.168.0.0/24 –j DROP

#Cerramos el acceso de la DMZ al firewall
iptables –A INPUT –s 217.126.241.16 –i eth2 –j DROP


#El resto se cierra rango bien conocidos 0.0.0.0 significa
cualquier red
iptables –A INPUT –s 0.0.0.0/0 –p tcp –dport 1:1024 –j DROP
iptables –A INPUT –s 0.0.0.0/0 –p  udp –dport 1:1024 –j DROP

#cerrar el Puerto de gestion webmin
iptables –A INPUT –s 0.0.0.0/0 -p tcp –dport 10000 –j DROP
```

¿Por que es necesario explicitar la abertura en un sentido y el otro? Por que si se cierra la todo el tráfico de la DMZ a la red local debemos permitir recibir las contestaciones.

Firewall con politica por defecto DROP

```
#!/bin/sh
#Flush de reglas
iptables –F
iptables –X
iptables –Z
iptables –t nat –F

#politica por defecto
iptables –P INPUT DROP
iptables –P OUTPUT DROP
iptables –P FORWARD DROP

# A nuestro firewall tenemos acceso total desde nuestra ip
iptables –A INPUT –s 217.126.241.16 –j ACCEPT
iptables –A OUTPUT –d 217.126.241.16 –j ACCEPT

# para el resto no hay acceso al firewall
iptables –A INPUT –s 0.0.0.0/0 –j DROP

#Ir metiendo reglas para cada servidor
# Servidor web
iptables –A FORWARD –d 1.2.3.4 –p tcp –dport 80 –j ACCEPT
iptables –A FORWARD –s 1.2.3.4 –p tcp –sport 80 –j ACCEPT

iptables –A FORWARD –d 1.2.3.4 –p tcp –dport 25 –j ACCEPT
iptables –A FORWARD –s 1.2.3.4 –p tcp –sport 25 –j ACCEPT

iptables –A FORWARD –d 1.2.3.4 –p tcp –dport 110 –j ACCEPT
iptables –A FORWARD –s 1.2.3.4 –p tcp –sport 110 –j ACCEPT
```

Otro ejemplo

```
#!/bin/sh
#Flush de reglas
iptables –F
iptables –X
iptables –Z
iptables –t nat –F

#politica por defecto
iptables –P INPUT DROP
iptables –P OUTPUT DROP
iptables –P FORWARD DROP

#Empezamos a filtrar. El localhost se deja
iptables –A INPUT –i lo –j ACCEPT
iptables –A OUTPUT –o lo –j ACCEPT

#A nuestra ip le dejamos todo
iptables –A INPUT –s 1.2.3.4 –j ACCEPT
```

```
iptables –A OUTPUT –d 1.2.3.4 -j ACCEPT

# A nuestro firewall tenemos acceso total desde nuestra ip
iptables –A INPUT –p tcp –dport 80 –j ACCEPT
iptables –A OUTPUT –p tcp –sport 80 –j ACCEPT

# A un cliente usar ftp
iptables –A INPUT –s 2.3.4.5 –p tcp –dport 20:21 –j ACCEPT
iptables –A INPUT –d 2.3.4.5 –p tcp –sport 20:21 –j ACCEPT

#Cerramos
iptables –A INPUT –p tcp –dport 1:1024
iptables –A INPUT –p udp –dport 1:1024
```

Cerramos otros puertos abiertos por encima de 1024

Depurar el funcionamiento del firewall
IPTRAF : Programa práctico para depurar firewalls con el que se puede observar si las conexiones se establecen o no y muestra en tiempo real el trafico que atraviesa la máquina con lujo de detalles. Se puede añadir a la regla algo así para ver si se ejecuta la regla

```
Iptables –A INPUT –s 1.2.3.4 –j ACCEPT && echo “regla 1 ok”
```

Servidor Proxys

Se puede traducir como “hacer algo en nombre de otro” es decir se encargara de actuar en nombre de muchos clientes. Otra característica importante es que puede tener caché de modo que hace un uso eficiente del ancho de banda.
Se usa para controlar o supervisar el tráfico hacia el exterior. Cuando usted se comunica con el mundo exterior el cliente le envia primero al servidor proxy. El servidor proxy establece la comunicación con el servidor solicitado y devuelve los datos. Estos se encargan de manejar todas las comunicaciones, característica que le permite registrar todo lo que ellos (usted) haga. Los servidores proxy http tienen muy en cuenta las URL que ellos o usted visiten. Los proxy de aplicación pueden autentificar a los usuarios. Antes de establecer una conexión con el exterior, el servidor le puede pedir que se identifique primero. A un usuario de la red le pediría una identificación para cada sitio que visite. SQUID es un gran paquete y trabaja con el caracteristico proxy transparente de linux.
SQUID es el software para el servidor proxy más popular entre sistemas operativos linux. Puede hacer de proxy y caché con los protocolos http, FTP, GOPHER, y WAIS, Proxy transparente, aceleración de http, caché de consultas DNS y otras cosas como filtrado de control de acceso por ip y contraseña.

Necesitamos aparte del software httpd (apache) el squid y los parches de seguridad disponibles. Utilizar siempre las versiones estables mas recientes.

Instalación

Rpm – ivh squid-*.i386.rpm

Se usara iptables para generar las reglas necesarias para el guion de enmascaramiento ip. Se instala con todas las distribuciones con kernel superior a 2.4

Configuración Básica
Utiliza el fichero de configuración localizado en /etc/squid/squid.conf y podrá trabajar sobre cualquier editor de texto. Existen gran número de parámetros, de los cuales se recomienda configurar.

http_port: Indicara el puerto a utilizar por squid, que por defecto para atender peticiones usa el 3128. En el caso de un proxy transparente regularmente se usara el 80 y se valdrá del redireccionamiento de peticiones de modo tal que no habrá necesidad de modificar la configuración de los navegadores para usar el proxy. Bastará con usarlo como puerta de enlace al servidor. Si se usa Apache por el puerto 80 deberá desintalarse o usar otro puerto.
Hoy en día no es del todo práctico usar un proxy transparente a menos que se trate de un servidor de internet o una oficina pequeña debido al abuso del acceso por parte del personal a

internet. Por ello resulta más conveniente un servidor proxy con restricciones por contraseña.
Se puede usar multiples veces. Ej.
http_port ip:puerto
http_port 192.168.0.1:3128
http_port 192.168.0.1:8080

cache_mem: Establece la cantidad de memoria ideal para lo siguiente.
Objetos en transito
Objetos HOT
Objetos negativamente almacenados en cache.
Los datos de esos objetos se almacenan en bloques de 4 kb. Este parámetro especifica un limite de tamaño total de bloques acomodados donde los objetos en transito tienen mayor prioridad. Por defecto se establecen 8MB. Puede utilizarse una cantidad mayor
Cache_mem nº MB
Cache_mem 16 MB

Cache_dir : Se usa para establecer que tamaño desea que tenga el caché en el disco duro. Para una gestión correcta e inteligente es necesario tener en cuenta con claridad que objetos deben ser cacheados. Por ejemplo no se cachean scripts o páginas con cabeceras de periodos de caducidad. Así que es posible especificar reglas para indicar que no se debe cachear y por cuanto tiempo.
Nota : esta probado que con una caché pequeña de un 10 de gigas se obtienen unos resultados de altos cercanos al 25% pero aumentando a una cache mas grande de por ejemplo 20 gigas es probable que no se llegue al 50% a no ser que las páginas se mantengan durante mucho tiempo.
Se puede incrementar el tamaño de la caché hasta donde el administrador lo desee. Mientras más grande más objetos se almacenarán y por lo tanto se usará menos ancho de banda. Si el tamaño excede de los disponible en disco squid se bloqueará.

Cache_dir ufs /var/spool/squid 700 16 256
700 MB de cache, con 16 subdirectorios de primer nivel y 256 de segundo nivel.

Referente_age 1 month : Se puede configurar el tiempo que pueden permanecer los objetos almacenados en la caché dependiendo de nuestras necesidades. Si el tiempo es bajo desaprovechamos una de las principales ventajas del servidor proxy y si es alto saturamos innecesariamente la capacidad de almacenaje. 1 mes es una opción razonable.

ftp_user: indicará una contraseña que deberá usarse al entrar como usuario anónimo en un servidor ftp.
ftp_user email
ftp_user proxy@dominio.net

Listas de control de acceso
Se establecen siguiente la siguiente sintaxis.
Acl nombre lista src componentes de la lista

Por ejemplo
Acl mired src 192.168.0.0/255.255.255.0
Puede definirse una lista de control de acceso invocando a un fichero con las direcciones validas
Acl permitidos src “/etc/squid/permitidos”

Reglas de control de acceso
Definen si permiten o no acceso a Squid. Se aplican a las listas de control de acceso. La sintaxis es la siguiente:
http_access deny/allow lista_control_acceso
http_access allow permitidos

También pueden definirse reglas usando ! lo cual significa excepcion.
http_access allow mired !permitidos
Aquí se le permite acceso a la lista mi red y no se le permite a permitidos.

Ejemplo
Acl all src 0.0.0.0/0.0.0.0
Acl manager proto cache-objetc
Acl localhost src 127.0.0.1/255.255.255.255
Acl todalared src 192.168.0.0/255.255.255.0

http_access allow localhost
http_access allow todalared
http_access deny all

cache_mgr : Por ejemplo si muere el proceso de la caché se enviara un mensaje a la cuenta del webmaster del servidor.
Cache_mgr email
Cache_mgr webmaster@informatica.com

Cache_peer : Se usa para especificar otros proxy-cache en una jerarquía como padres o hermanos, es decir definir si jay un proxy delante o en paralelo. Sintaxis
Cache_peer servidor tipo http_port icp_port opciones
Ej. Si nuestra caché trabaja detrás de otro servidor cache con ip 192.168.0.1 escuchando peticiones http por el 8080 y especificar que no se almacenen los objetos presentes en la cache padre
Cache_peer 192.168.0.1 parent 8080 3130 proxy-only

Si se trabaja con otros caches hermanos o vecinos con ip 10.1.0.1, 10.2.0.1, 10.3.0.1 todos escuchando peticiones http por el 8080 y trabajando con puerto 3130 y especificando que no se almacenen los objetos de las otras caches.
Cache_peer 10.1.0.1 sibbling 8080 3130 proxy-only
Cache_peer 10.2.0.1 sibbling 8080 3130 proxy-only
Cache_peer 10.3.0.1 sibbling 8080 3130 proxy-only

Cache con aceleración
Cuando un usuario hace una petición hacia un objeto en internet, este se almacena en la caché squid. Si posteriormente se hace una petición al mismo objeto y este no ha sido modificado desde que se accedió squid mostrará el que tiene en caché en lugar de

volver a descargarlo de internet. Esta funcion permite navegar rapidamente cuando los objetos ya estan en cache de squid y mejora enormemente la utilización del ancho de banda.

Proxy convencional acelerado
Httpd_accel_host virtual
Httpd_accel_port 0
Httpd_accel_with_proxy on

Proxy transparente acelerado
Httpd_accel_host 192.168.0.2
Httpd_accel_port 80
Httpd_accel_with_proxy on
Httpd_Accel_uses_host_header on
Si usa navegador internet explorer con versiones anteriores a la 5.5 que tienen un pésimo soporte para proxies transparentes imposibilitando la capacidad de refrescar el contenido añadir
Ie_refresh on

La regla para iptables para el proxy podría ser
Iptables –t nat –A PREROUTING –i eth0 –p tcp –dport 80 –j REDIRECT –to-port 8080
Cualquier peticion hacia el Puerto 80 hecha desde la red local hacia el exterior lo redireccionará hacia el 8080

Establecer el idioma español por defecto en las páginas de error informativas.
Ln –s /usr/share/squid/errors/spanish /etc/squid/errors

Iniciar el servicio: service squid start
Reiniciar : service squid restart
Hacer que arranque automáticamente la próxima vez en niveles 3,4,5 : chkconfig squid on

Depurar errores
Para realizar un diagnostico indicándole a squid que vuelva a leer la configuración, lo cual devolverá los errores que existan en /etc/squid/squid.conf
Service squid reload
O
Squid –d 5 → modo depuración

Filtrando correo en postfix

Postfix es un servidor de correo que intenta ser rápido, fácil de administrar y seguro, mientras que al mismo tiempo es suficientemente compatible con sendmail para seguir usando usuarios existentes.
Un servidor de correo smtp postfix se encarga de enviar y recibir correo y el servidor imap permite acceder al correo en una maquina para cada usuario y tiene la ventaja sobre pop3 de poder bajarse solo los encabezados.

Crear las carpetas Maildir dentro de los home de cada usuario.

```
Mkdir $HOME/Maildir
Mkdir $HOME/Maildir/tmp
Mkdir $HOME/Maildir/new
Mkdir $HOME/Maildir/cur

Mkdir $HOME/Maildir/.Spam
Mkdir $HOME/Maildir/.Spam/tmp
Mkdir $HOME/Maildir/.Spam/new
Mkdir $HOME/Maildir/.Spam/cur
```

/etc/login.defs
QMAIL_DIR/Mailidir #Todos los usuarios que hagan login en el sistema usan Maildir y comentar las lineas que aparezcan al lado

/etc/postfix/master.cf
Añadir la linea

```
Virtual unix – n n - - virtual
```

(Reemplazar la n con y todas las columnas excepto proxymap, local, virtual o pipe en la ultima columna)
/etc/postfix/main.cf

```
command_directory = /usr/sbin
config_directory = /etc/postfix
daemon_directory = /usr/lib/postfix (libexec)
program_directory = /usr/lib/postfix
debug_peer_level=2
mail_owner = postfix
smtpd_banner = $myhostname ESMTP $mail_name
setgid_group = postdrop
biff = no
maximal_queue_lifetime=1d
message_size_limit = 50000000
queue_directory=/var/spool/postfix
append_dot_mydomain = yes
myhostname = nct.nct-informatica.com
mydomain = nct-informatica.com
alias_maps = hash:/etc/postfix/aliases
alias_database = hash:/etc/postfix/aliases
myorigin = /etc/mailname ($mydomain)
mydestination =  $mydomain, $myhostname, localhost
mynetworks = 192.168.0.0/24 127.0.0.0/8
inet_interfaces = all
```

```
mailbox_size_limit = 0
home_mailbox = Maildir/
queue_directory = /var/spool/postfix
readme_directory = no
sample_directory = /etc/postfix
setgid_group=postdrop
unknown_address_reject_code = 550
unknown_local_recipient_reject_code = 550

recipient_delimeter = +
relayhost =
content_filter = smtp-amavis:(127.0.0.1):10024

smtpd_recipient_restrictions =
      reject_non_fqdn_sender,
      reject_non_fqdn_recipient,
      reject_unknown_sender_domain,
      reject_unauth_pipeling
      check_client_access hash:/etc/postfix/mynetworks,
      permit_mynetworks,
      reject_unauth_destination

smtpd_restricction_clases =
      has_our_domain_As_sender

has_our_domain_as_sender =
      check_sender_access hash:/etc/postfix/our_domain_as_sender
      reject

/etc/postfix/mynetworks
192.168.0 has_our_domain_as sender
… ips que pueden mandar mails
 217.127.245.37 has_our_domain_as sender
127.0.0 has_our_domain_as sender

/etc/postfix/our_domain_as_sender
<>    OK
nct-informatica.com   OK
… lista de direcciones y dominios que pueden mandar con ok

pw2userdb

postconf –n

Abrir un .forward en cada home del usuario
| procmail

/etc/procmailrc
MAILDIR=$HOME/Maildir
DEFAULT=./

Mailq ; Para ver la cola de correo

Postsuper –d ALL ; Para borrar la cola

Reiniciar postfix: service postfix restart
```

amavisd-new : Version de amavis (A mail virus scanner) se diferencia de la versión anterior en que recoge el correo desde un puerto del servidor de correo y lo devuelve a otro puerto, además de que sabe usar spamassassin. Lo que hay que conseguir es que postfix envie el correo a ese puerto antes de hacer la entrega. Luego amavisd –new devuelve el mensaje a postfix y este hace la entrega definitiva.

/etc/postfix/master.cf
smtp-amavis
```
amavis-smtp     unix -     -     y     -     2     smtp
-o smtp_data_done_timeout=1200s
-o smtp_never_send_ehlo=yes
-o disable_dns_lookups=yes
localhost:10025 inet n     -     y     -     -     smtpd
-o content_filter=
-o local_recipient_maps=
-o smtpd_helo_restrictions=
-o smtpd_client_restrictions=
-o smtpd_sender_restrictions=
-o smtpd_recipient_restrictions=permit_mynetworks,reject
-o mynetworks=192.168.0.0/24
-o strict_rfc821_envelopes=yes
```

(la primera línea crea el filtro para amavis y la segunda crea el servidor local por el que permitirá cualquier correo sin intentar filtrarlo. Para que amavisd-new haga uso de spamassassin debemos cerciorarnos de que no existe esta línea en el fichero de configuración de amavisd-new
@bypass_spam_checks_acl = qw(.);)

/etc/postfix/main.cf

```
content_filter= amavis-smtp:localhost:10024
```

/etc/postfix/aliases

/etc/amavis/amavisd.conf
Se encarga de interceptor mensajes y apartarlos descomprimiendo los ficheros adjuntos en ficheros separados que son escaneados por un antivirus externo.
La configuración de este archive es trivial basta con comentar o descomentar ciertas líneas. Colocar el dominio correcto y si se añaden un correo colocar \ antes de @ , es decir comercial\@pepe-informatica.com.

Se debe crear una cuenta spam que recogerá los mensajes clasificados como tal. El archivo de configuración envía todos los mensajes de spam a una única cuenta de correo. Los usuarios no recibirán estos mensajes en ningún momento.

```
$daemon_user =’amavis’;
$daemon_group=’amavis’;
$mydomain = ‘pepe-informatica.com’;
```

```
$forward_method= ‘smtp:127.0.0.1:10025’
$notify_method = $forward_method;
$final_spam_destiny = D_PASS;
$sa_tag_level_deflt = 4.0;
$sa_Tag2_level_deflt = 5.0;
$sa_kill_level_deflt = $sa_tag2_level_deftl;

[‘clam antivirus-clamd’,
     \&ask_daemon,                [“CONTSCAN                {}\n,
‘/var/run/clamav/clamd.ctl’],
     qr/\bOK$/, qr/\bFOUND$/,
     qr/^.*?: (?!Infected Archive)(.*) FOUND$/ ],
```

Spamassassin : Aplicación que realiza una serie de pruebas. Para cada prueba que supera, le asigna una puntuación. Cuando la puntuación llega a 5 (valor por defecto) entiende que se trata de un mensaje de spam. Además desde hace poco también incluye filtros bayesianos que permiten ajustar la clasificacion de los mensajes. En caso de postitivo/negativo se puede adiestrar para aprender de esta caracteristica en el futuro. Aunque necesita de gran numero de mensajes para aprender. Para facilitar esta operación

- Crear la cuenta de spam, tipo imap.
- Crear una cuenta imap llamada spam y que accederá a la cuenta spam del servidor de correo.
- Mediante el cliente de correo se crearan dos subcarpetas de la carpeta “entrada” de la cuenta spam: una llamada spam-si y otra spam-no
- Periódicamente el servidor de correo recogerá los mensajes del directorio /var/mail/dominio.com/spam/spam-si/cur /var/mail/dominio.com/spam/spam-no/cur y readistraremos spamassassin según la nueva regla.

De esta forma, si recibimos un falso positivo (mensaje legitimo clasificado erróneamente como spam) solo lo arrastraremos a la subcarpeta spam-no. Y al revés. No hará falta borrarlos, puesto que será el mismo servidor el que lo haga. Los falsos positivos, no los moveremos sino que los copiamos a spam-no.

Entrar al servidor de correo como root y editamos su crontab –e
3… /bin/mv /var/mail/dominio.com/spam/.spam-si/cur/* /var/lib/amavis/spam/. > /dev/null 2>/Dev/null
3… /bin/mv /var/mail/dominio.com/spam/.spam-no/cur/* /var/lib/amavis/ham/. >/dev/null 2>/dev/null
4… /bin/chown amavis:amavis /var/lib/amavis/spam/* >/dev/null 2>/dev/null
4… /bin/chown amavis:amavis /var/lib/amavis/ham/* >/dev/null 2>/dev/null

cambiamos a usuario amavis
cd par air al directorio de inicio de amavis´
crear dos directories spam y ham para procesar mensajes legitimos
editar crontab –e y añadir

5… /usr/bin/sa-learn –forget –dir /var/lib/amavis/spam >/dev/null 2>/dev/null
5… /usr/bin/sa-learn –forget –dir /var/lib/amavis/ham >/dev/null 2>/dev/null
6… /usr/bin/sa-learn –spam –dir /var/lib/amavis/spam >/dev/null 2>/dev/null
6… /usr/bin/sa-learn –ham –dir /var/lib/amavis/ham >/dev/null 2>/dev/null
7… rm –f /var/lib/amavis/spam/* >/dev/null 2>/dev/null
7… rm –f /var/lib/amavis/ham/* >/dev/null 2>/dev/null

Puesto que spamassassin aprende automaticamente, lo primero que se deve hacer es que olvide lo que ha aprendido de los mensajes mediante la opcion –forget, a los directorios ham y spam.
Adiestrarlo con la información correcta

- sa-learn –spam aprende del contenido del directorio spam
- sa-learn –ham aprende del directorio ham (no spam)

Una vez terminado se borra el contedido de los directorios.

Puede ser preconfigurado /etc/mail/spammassassin/local.cf

Whitelist_from jcerpa@nct-informatica.com
Whitelist_from amartin@nct-informatica.com