Cómo configurar Squid

1.- Parámetros básicos para servidor Proxy.

Introducción.

Squid es el software para servidor Proxy más popular y extendido entre los sistemas operativos basados sobre UNIX®. Es muy confiable, robusto y versátil. Al ser *software libre*, además de estar disponible el código fuente, está libre del pago de costosas licencias por uso o con restricción a un uso con determinado número de usuarios.

Entre otras cosas, Squid puede hacer Proxy y cache con los protocolos HTTP, FTP, GOPHER y WAIS, Proxy de SSL, cache transparente, WWCP, aceleración HTTP, cache de consultas DNS y otras muchas más como filtración de contenido y control de acceso por IP y por usuario.

NOTA ESPECIAL: Squid **no puede funcionar como proxy** para servicios como SMTP, POP3, TELNET, SSH, etc. Si se requiere hacer proxy para cualquier cosa distinta a HTTP, HTTPS, FTP, GOPHER y WAIS se requerirá o bien implementar enmascaramiento de IP a través de un NAT *(Network Address Translation)* o bien hacer uso de un servidor SOCKS como Dante.

Software requerido.

Para poder llevar la cabo los procedimientos descritos en este manual y documentos relacionados, usted necesitará tener instalado al menos lo siguiente:

- squid-2.5.STABLE1
- httpd-2.0.x (Apache)
-
  -

  Todos los parches de seguridad disponibles para la versión de Red Hat™ que esté utilizando.

Tómese en consideración que, de ser posible, se debe utilizar **siempre** las versiones estables más recientes de todo el software que vaya a instalar al realizar los procedimientos descritos en este manual, a fin de contar con los parches de seguridad necesarios.**Ninguna versión de Squid anterior a la 2.5.STABLE1 se considera como apropiada** debido a fallas de seguridad de gran

importancia, y ningún administrador *competente* utilizaría una versión inferior a la 2.5.STABLE1. Por favor visite el sito Web de su distribución predilecta para estar al tanto de cualquier aviso de actualizaciones de seguridad.

Para Red Hat™ Linux 9 hay paquetería de actualización en los siguientes enlaces:
- ftp://updates.redhat.com/9/en/os/i386/, si posee alguna distribución basada sobre Red Hat™ Linux 9

Instalación del software necesario.

Regularmente Squid no se instala de manera predeterminada a menos que especifique o contrario durante la instalación del sistema operativo, sin embargo viene incluido en casi todas las distribuciones actuales. El procedimiento de instalación es exactamente el mismo que con cualquier otro software:

```
mount /mnt/cdrom/
rpm -Uvh /mnt/cdrom/*/RPMS/squid-*.i386.rpm
eject
```

Si utiliza Fedora™ Core, ejecute lo siguiente y se instalará todo lo necesario junto con sus dependencias:

```
yum install squid httpd
```

Iptables se utilizará para generar las reglas necesarias para el guión de Enmascaramiento de IP. Se instala por defecto en todas las distribuciones actuales que utilicen kernel-2.4.

Es importante tener actualizado el kernel por diversas cuestiones de seguridad. No es recomendable utilizar versiones del kernel anteriores a la **2.4.20**. En el manual "Cómo actualizar el Kernel a partir de paquetes RPM®" se describe a detalle lo necesario.

Antes de continuar

Tenga en cuenta que este manual ha sido comprobado varias veces y ha funcionado en todos los casos y si algo no funciona solo significa que usted no lo leyó a detalle y no siguió correctamente las indicaciones.

Evite dejar **espacios vacíos** en lugares indebidos. El siguiente es un ejemplo de como **no** debe des-comentarse un parámetro.
Mal

```
# Opción incorrectamente des-comentada
  http_port 3128
```

El siguiente es un ejemplo de como **si** debe des-comentarse un parámetro.
Bien

```
# Opción correctamente des-comentada
http_port 3128
```

Configuración básica.

Squid utiliza el fichero de configuración localizado en */etc/squid/squid.conf*, y podrá trabajar sobre este utilizando su editor de texto preferido. Existen un gran número de parámetros, de los cuales recomendamos configurar los siguientes:

- http_port
- cache_mem
- ftp_user
- cache_dir
- Al menos una *Lista de Control de Acceso*
- Al menos una *Regla de Control de Acceso*
- httpd_accel_host
- httpd_accel_port
- httpd_accel_with_proxy

Parámetro http_port: ¿Que puerto utilizar para Squid?

Squid por defecto utilizará el puerto 3128 para atender peticiones, sin embargo se puede especificar que lo haga en cualquier otro puerto o bien que lo haga en varios puertos a la vez.

En el caso de un *Proxy Transparente*, regularmente se utilizará el puerto 80 y se valdrá del re-direccionamiento de peticiones de modo tal que no habrá necesidad alguna de modificar la configuración de los navegadores Web para utilizar el servidor Proxy. bastará con utilizar como puerta de enlace al servidor. Es importante recordar que los servidores Web, como Apache, también utilizan dicho puerto, por lo que será necesario reconfigurar el servidor Web para utiliza otro puerto disponible, o bien desinstalar o deshabilitar el servidor Web.

Hoy en día ya no es del todo práctico el utilizar un *Proxy Transparente*, a menos que se trate de un servicio de *Café Internet* u oficina pequeña, siendo que uno de los principales problemas con los que lidian los administradores es el mal uso y/o abuso del acceso a Internet por parte del personal. Es por esto que puede resultar más conveniente configurar un servidor Proxy con restricciones por contraseña, lo cual no puede hacerse con un *Proxy Transparente*, debido a que se requiere un diálogo de nombre de usuario y contraseña.

Regularmente algunos programas utilizados comúnmente por los usuarios suelen traer por defecto el puerto 8080 *-servicio de cacheo WWW-* para utilizarse al configurar que servidor proxy utilizar. Si queremos aprovechar esto en nuestro favor y ahorrarnos el tener que dar explicaciones innecesarias al usuario, podemos especificar que Squid escuche peticiones en dicho puerto también. Siendo así localice la sección de definición de *http_port*, y especifique:

```
#
#       You may specify multiple socket addresses on multiple lines.
#
# Default: http_port 3128
http_port 3128
http_port 8080
```

Si desea incrementar la seguridad, puede vincularse el servicio a una IP que solo se pueda acceder desde la red local. Considerando que el servidor utilizado posee una IP 192.168.1.254, puede hacerse lo siguiente:

```
#
#       You may specify multiple socket addresses on multiple lines.
#
# Default: http_port 3128
http_port 192.168.1.254:3128
```

```
http_port 192.168.1.254:8080
```

Parámetro cache_mem

El parámetro cache_mem establece la cantidad ideal de memoria para lo siguiente:
- Objetos en tránsito.
- Objetos Hot.
- Objetos negativamente almacenados en el caché.

Los datos de estos objetos se almacenan en bloques de 4 Kb. El parámetro cache_mem especifica un límite máximo en el tamaño total de bloques acomodados, donde los objetos en tránsito tienen mayor prioridad. Sin embargo los objetos *Hot* y aquellos negativamente almacenados en el caché podrán utilizar la memoria no utilizada hasta que esta sea requerida. De ser necesario, si un objeto en tránsito es mayor a la cantidad de memoria especificada, Squid excederá lo que sea necesario para satisfacer la petición.

Por defecto se establecen 8 MB. Puede especificarse una cantidad mayor si así se considera necesario, dependiendo esto de los hábitos de los usuarios o necesidades establecidas por el administrador.

Si se posee un servidor con al menos 128 MB de RAM, establezca 16 MB como valor para este parámetro:

```
cache_mem 16 MB
```

Parámetro cache_dir: ¿Cuanto desea almacenar de Internet en el disco duro?

Este parámetro se utiliza para establecer que tamaño se desea que tenga el cache en el disco duro para Squid. Para entender esto un poco mejor, responda a esta pregunta: *¿Cuanto desea almacenar de Internet en el disco duro?* Por defecto Squid utilizará un cache de 100 MB, de modo tal que encontrará la siguiente línea:

```
cache_dir ufs /var/spool/squid 100 16 256
```

Se puede incrementar el tamaño del cache hasta donde lo desee el administrador. Mientras más grande el cache, más objetos de almacenarán en éste y por lo tanto se utilizará menos el ancho de banda. La siguiente línea establece un cache de 700 MB:

```
cache_dir ufs /var/spool/squid 700 16 256
```

Los números *16* y *256* significan que el directorio del cache contendrá 16 subdirectorios con 256 niveles cada uno. No modifique esto números, no hay necesidad de hacerlo.

Es muy importante considerar que si se especifica un determinado tamaño de cache y este excede al espacio real disponible en el disco duro, Squid se bloqueará inevitablemente. Sea cauteloso con el tamaño de cache especificado.

Parámetro ftp_user

Al acceder a un servidor FTP de manera anónima, por defecto Squid enviará como contraseña *Squid*@. Si se desea que el acceso anónimo a los servidores FTP sea más informativo, o bien si

se desea acceder a servidores FTP que validan la autenticidad de la dirección de correo especificada como contraseña, puede especificarse la dirección de correo electrónico que uno considere pertinente.

```
ftp_user proxy@su-dominio.net
```

Controles de acceso.

Es necesario establecer *Listas de Control de Acceso* que definan una red o bien ciertas maquinas en particular. A cada lista se le asignará una *Regla de Control de Acceso* que permitirá o denegará el acceso a Squid. Procedamos a entender como definir unas y otras.

Listas de control de acceso.

Regularmente una lista de control de acceso se establece siguiendo la siguiente sintaxis:

```
acl [nombre de la lista] src [lo que compone a la lista]
```

Si uno desea establecer una lista de control de acceso que defina sin mayor trabajo adicional a toda la red local definiendo la IP que corresponde a la red y la máscara de la sub-red. Por ejemplo, si se tienen una red donde las máquinas tienen direcciones IP 192.168.1.n con máscara de sub-red 255.255.255.0, podemos utilizar lo siguiente:

```
acl miredlocal src 192.168.1.0/255.255.255.0
```

También puede definirse una *Lista de Control de Acceso* invocando un fichero localizado en cualquier parte del disco duro, y en el cual se en cuenta una lista de direcciones IP. Ejemplo:

```
acl permitidos src "/etc/squid/permitidos"
```

El fichero */etc/squid/permitidos* contendría algo como siguiente:

```
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.15
192.168.1.16
192.168.1.20
192.168.1.40
```

Lo anterior estaría definiendo que la *Lista de Control de Acceso* denominada *permitidos* estaría compuesta por las direcciones IP incluidas en el fichero */etc/squid/permitidos*.

Reglas de Control de Acceso

Estas definen si se permite o no el acceso a Squid. Se aplican a las *Listas de Control de Acceso*. Deben colocarse en la sección de reglas de control de acceso definidas por el administrador, es decir, a partir de donde se localiza la siguiente leyenda:

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
```

La sintaxis básica es la siguiente:

```
http_access [deny o allow] [lista de control de acceso]
```

En el siguiente ejemplo consideramos una regla que establece acceso permitido a Squid a la *Lista de Control de Acceso*denominada *permitidos*:

```
http_access allow permitidos
```

También pueden definirse reglas valiéndose de la expresión **!**, la cual significa *excepción*. Pueden definirse, por ejemplo, dos listas de control de acceso, una denominada *lista1* y otra denominada *lista2*, en la misma regla de control de acceso, en donde se asigna una expresión a una de estas. La siguiente establece que se permite el acceso a Squid a lo que comprenda*lista1* excepto aquello que comprenda *lista2*:

```
http_access allow lista1 !lista2
```

Este tipo de reglas son útiles cuando se tiene un gran grupo de IP dentro de un rango de red al que se debe **permitir** acceso, y otro grupo dentro de la misma red al que se debe **denegar** el acceso.

Aplicando Listas y Reglas de control de acceso.

Una vez comprendido el funcionamiento de la Listas y las Regla de Control de Acceso, procederemos a determinar cuales utilizar para nuestra configuración.

Caso 1

Considerando como ejemplo que se dispone de una red 192.168.1.0/255.255.255.0, si se desea definir toda la red local, utilizaremos la siguiente línea en la sección de *Listas de Control de Acceso*:

```
acl todalared src 192.168.1.0/255.255.255.0
```

Habiendo hecho lo anterior, la sección de listas de control de acceso debe quedar más o menos del siguiente modo:

Listas de Control de Acceso: definición de una red local completa

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl todalared src 192.168.1.0/255.255.255.0
```

A continuación procedemos a aplicar la regla de control de acceso:

```
http_access allow todalared
```

Habiendo hecho lo anterior, la zona de reglas de control de acceso debería quedar más o menos de

este modo:

Reglas de control de acceso: Acceso a una Lista de Control de Acceso.

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow todalared
http_access deny all
```

La regla **http_access allow todalared** permite el acceso a Squid a la *Lista de Control de Acceso* denominada *todalared*, la cual está conformada por 192.168.1.0/255.255.255.0. Esto significa que cualquier máquina desde 192.168.1.1 hasta 192.168.1.254 podrá acceder a Squid.

Caso 2

Si solo se desea permitir el acceso a Squid a ciertas direcciones IP de la red local, deberemos crear un fichero que contenga dicha lista. Genere el fichero */etc/squid/lista*, dentro del cual se incluirán solo aquellas direcciones IP que desea confirmen la Lista de Control de acceso. Ejemplo:

```
192.168.1.1
192.168.1.2
192.168.1.3
192.168.1.15
192.168.1.16
192.168.1.20
192.168.1.40
```

Denominaremos a esta lista de control de acceso como *redlocal*:

```
acl redlocal src "/etc/squid/lista"
```

Habiendo hecho lo anterior, la sección de listas de control de acceso debe quedar más o menos del siguiente modo:

Listas de Control de Acceso: definición de una red local completa

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl redlocal src "/etc/squid/lista"
```

A continuación procedemos a aplicar la regla de control de acceso:

```
http_access allow redlocal
```

Habiendo hecho lo anterior, la zona de reglas de control de acceso debería quedar más o menos de este modo:

Reglas de control de acceso: Acceso a una Lista de Control de Acceso.

```
#
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow redlocal
http_access deny all
```

La regla **http_access allow redlocal** permite el acceso a Squid a la *Lista de Control de Acceso* denominada *redlocal*, la cual está conformada por las direcciones IP especificadas en el fichero */etc/squid/lista*. esto significa que cualquier máquina no incluida en */etc/squid/lista* no tendrá acceso a Squid.

Parámetro cache_mgr.

Por defecto, si algo ocurre con el Cache, como por ejemplo que muera el procesos, se enviará un mensaje de aviso a la cuenta*webmaster* del servidor. Puede especificarse una distinta si acaso se considera conveniente.

```
cache_mgr joseperez@midominio.net
```

Parámetro cache_peer: caches padres y hermanos.

El parámetro cache_peer se utiliza para especificar otros proxy-cache en una jerarquía como *padres* o como *hermanos*. es decir, definir si hay un proxy adelante o en parelelo. La síntaxis básica es la siguiente:

```
cache_peer servidor tipo http_port icp_port opciones
```

Ejemplo: Si su cache va a estar trabajando detrás de otro servidor cache, es decir un cache padre, y considerando que el cache padre tiene una IP 192.168.1.1, escuchando peticiones HTTP en el puerto 8080 y peticiones ICP en puerto 3130 **(puerto utilizado por defecto por Squid)** ,especificando que no se almacenen en cache los objetos que ya están presentes en el cache del proxy padre, utilice la siguiente línea:

```
cache_peer 192.168.1.1 parent 8080 3130 proxy-only
```

Cuando se trabaja en redes muy grandes donde existen varios servidores proxy haciendo cache de contenido de Internet, es una buena idea hacer trabajar todos los cache entre si. Configurar caches vecinos como *sibbling* (hermanos) tiene como beneficio el que se consultarán estos caches localizados en la red local antes de acceder hacia Internet y consumir ancho de banda para acceder hacia un objeto que ya podría estar presente en otro cache vecino.

Ejemplo: Si su cache va a estar trabajando en paralelo junto con otros caches, es decir caches hermanos, y considerando los caches tienen IP 10.1.0.1, 10.2.0.1 y 10.3.0.1, todos escuchando peticiones HTTP en el puerto 8080 y peticiones ICP en puerto 3130, especificando que no se almacenen en cache los objetos que ya están presentes en los caches hermanos, utilice las siguientes líneas:

```
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
```

Pueden hacerse combinaciones que de manera tal que se podrían tener caches padres y hermanos trabajando en conjunto en una red local. Ejemplo:

```
cache_peer 10.0.0.1 parent 8080 3130 proxy-only
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 petición hacia un objeto en Internet, este es almacenado en el cache de Squid. Si otro usuario hace petición hacia el mismo objeto, y este no ha sufrido modificación alguna desde que lo accedió el usuario anterior, Squid mostrará el que ya se encuentra en el cache en lugar de volver a descargarlo desde Internet.

Esta función permite navegar rápidamente cuando los objetos ya están en el cache de Squid y además optimiza enormemente la utilización del ancho de banda.

En la sección *HTTPD-ACCELERATOR OPTIONS* deben habilitarse los siguientes parámetros:

Proxy Acelerado: Opciones para Proxy Convencional.

```
httpd_accel_host virtual
httpd_accel_port 0
httpd_accel_with_proxy on
```

Si se trata de un Proxy transparente deben utilizarse con las siguientes opciones, considerando que se hará uso del cache de un servidor web (Apache) como auxiliar:

Proxy Acelerado: Opciones para Proxy Transparente.

```
# Debe especificarse la IP de cualquier servidor Web en la red local
# o bien el valor virtual
httpd_accel_host 192.168.1.254
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Nota acerca de Internet Explorer 5.5 y versiones anteriores

Si va a utilizar Internet Explorer 5.5 y versiones anteriores con un proxy transparente, es importante recuerde que dichas versiones tiene un pésimo soporte con los proxies transparentes imposibilitando por completo la capacidad de refrescar contenido. Lo más conveniente es actualizar hacia Internet Explorer 6.x o defintivamente optar por otras alternativas comoMozilla, que consiste en una suite completa de aplicaciones para Internet, o bien Mozilla Firebird, que consiste en un navegador muy ligero y que cumple con los estándares, de las cuales encontrará versión para Windows. Si se utiliza el parámetro *ie_refresh* con valor *on* puede hacer que se verifique en los servidores de origen para nuevo contenido para todas las peticiones **IMS-REFRESH** provenientes de Internet Explorer 5.5 y versiones anteriores.

Proxy Acelerado: Opciones para Proxy Transparente para redes con Internet

Exlorer 5.5 y versiones anteriores.

```
# Debe especificarse la IP de cualquier servidor Web en la red local
# o bien virtual
httpd_accel_host 192.168.1.254
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
ie_refresh on
```

La configuración de Squid como proxy transparente solo requiere complementarse utilizando una regla de *iptables* que se encargará de redireccionar peticiones haciéndolas pasar por el puerto 8080.

Proxy Acelerado: Regla de iptables.

```
# Considerando que la red local accede a través de eth0 y que Squid
# escucha peticiones en puerto 8080, se utiliza la siguiente línea:
/sbin/iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT
  --to-port 8080
```

Por defecto el parámetro *httpd_accel_with_proxy* viene con el valor *off*, es importante no olvidar cambiar este valor por *on*.

Estableciendo el idioma por defecto.

Squid **incluye traducción a distintos idiomas de las distintas páginas de error e informativas que son desplegadas en un momento dado. Dichas traducciones se pueden encontrar en /*usr/lib/squid/errors/*. Para poder hacer uso de las páginas de error traducidas al español, es necesario cambiar un enlace simbólico localizado en /etc/squid/errors para que apunte hacia */usr/lib/squid/errors/Spanish* en lugar de hacerlo hacia */usr/lib/squid/errors/English*.**

Elimine primero el enlace simbólico actual:

```
rm -f /etc/squid/errors
```

Coloque un nuevo enlace simbólico apuntando hacia el directorio con los ficheros correspondientes a los errores traducidos al español.

Red Hat™ Linux 7.x y 8.0

```
ln -s /usr/lib/squid/errors/Spanish /etc/squid/errors
```

Red Hat™ Linux 9 y Fedora™ Core 1

```
ln -s /usr/share/squid/errors/Spanish /etc/squid/errors
```

Iniciando, reiniciando y añadiendo el servicio al arranque del sistema.

Una vez terminada la configuración, ejecute el siguiente comando para iniciar por primera vez Squid:

```
service squid start
```

Si necesita reiniciar para probar cambios hechos en la configuración, ejecute lo siguiente:

```
service squid restart
```

Si desea que Squid inicie de manera automática la próxima vez que inicie el sistema, ejecute lo siguiente:

```
/sbin/chkconfig squid on
```

Lo anterior habilitará a Squid en los niveles de corrida 3, 4 y 5.

Nota para los novatos: Usted **NO** tiene porque editar cosa alguna en */etc/rc.d/rc.local* o */etc/inittab* para queSquid *-así como cualquier otro servicio-* inicie en el arranque del sistema. Mientras usted sea novato, por favor, olvide que existen esos ficheros y **exclame una fuerte amenaza y alejese** de quien le indique que desde ahí debe arrancar servicios.

Depuración de errores

Cualquier error al inicio de squid solo significa que hubo errores de síntaxis, errores de dedo o bien se están citando incorrectamente las rutas hacia los ficheros de las *Listas de Control de Acceso*.

Puede realizar disgnóstico de problemas indicandole a Squid que vuelva a leer configuración, lo cual devolvera los errores que existan en /etc/squid/squid.conf.

```
service squid reload
```

También puede iniciar Squid directamente desde la líena de comando especicicando el modo de depuración:

```
squid -d 5
```

Ajustes para el muro contrafuegos o guión de Enmascaramiento de IP.

A continuación comentaremos algunos ajustes que pueden añadirse o editarse en el guión de el muro contrafuegos, como el generado por herramientas como Firestarer, o bien un simple guión de Enmascaramiento de IP.

Sugerimos utilizar Firestarer debido a que permite configurar tanto el enmascaramiento de IP como el muro contrafuegos y la importancia que tiene la presencia de éste último en un servidor que sirve como puerta de enlace para la red local.

Use Iptables en lugar de ipchains.

Desde el kernel 2.4, GNU/Linux utiliza Netfilter, el cual se configura a través de *iptables*. La sintaxis cambia con respecto a *ipchains*, y a fin de permitir a los administradores darse tiempo de adaptarse, distribuciones como Red Hat™ ™ incluyeron soporte para *ipchains* a manera de *aplicación de legado*.

Pudiendo utilizarse *iptables* no tiene sentido mantener instalado *ipchains*, que aún es utilizado por defecto en Red Hat™ Linux ™ 7.1 y 7.2. Se recomienda desinstalar ipchains y los paquetes que dependan de este.

Es importante utilizar la más reciente versión de *iptables* para la distribución utilizada. **Ninguna versión de iptables anterior a la 1.2.4 se considera como apropiada** debido a fallas de seguridad de gran importancia, y ningún administrador *competente* utilizaría una versión inferior a la 1.2.4. Por favor visite el sito Web de su distribución predilecta para estar al tanto de cualquier aviso de actualizaciones de seguridad. Ejemplo: para Red Hat™ Linux 7.2, 7.3. 8.0 y 9 hay paquetería de actualización en los siguientes enlaces:

- ftp://updates.redhat.com/7.2/en/os/i386/, si posee alguna distribución basada sobre Red Hat™ Linux 7.2
- ftp://updates.redhat.com/7.3/en/os/i386/, si posee alguna distribución basada sobre Red Hat™ Linux 7.3
- ftp://updates.redhat.com/8.0/en/os/i386/, si posee alguna distribución basada sobre Red Hat™ Linux 8.0
- ftp://updates.redhat.com/9/en/os/i386/, si posee alguna distribución basada sobre Red Hat™ Linux 9

Antes de desinstalar *ipchains*, que se instala de modo pre-determinado en Red Hat™ Linux 7.x, primero debe eliminarse cualquier regla que pudiese existir.

```
/sbin/ipchains -X
/sbin/ipchains -F
/sbin/ipchains -Z
```

A continuación debe removerse el módulo de *ipchains* para permitir la carga del módulo *ip_tables*.

```
/sbin/rmmod ipchains
/sbin/modprobe ip_tables
```

Para terminar, si utiliza Red Hat™ Linux 7.2 y 7.3, se debe desinstalar *ipchains* y toda la paquetería que dependa de éste.

```
rpm -e ipchains lokkit gnome-lokkit firewall-config
```

Esto ajustes deben poder permitir utilizar *iptables* en lugar de *ipchains* sin mayor problema.

Re-direccionamiento de peticiones.

En un momento dado se requerirá tener salida transparente hacia Internet para ciertos servicios, pero al mismo tiempo se necesitará re-direccionar peticiones hacia servicio Web, Web SSL, ftp, gopher o WAIS hacia el el puerto donde escucha peticiones Squid (8080), de modo que no haya salida alguna hacia alguno de estos protocolos sin que ésta pase antes por Squid.

El re-direccionamiento lo hacemos a través de *iptables*. Considerando para este ejemplo que la red

local se accede a través de una interfaz eht0, el siguiente esquema ejemplifica un re-direccionamiento:

```
/sbin/iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT
  --to-port 8080
```

Lo anterior hace que cualquier petición hacia el puerto 80 (servicio HTTP) hecha desde la red local hacia el exterior, se re-direccionará hacia el puerto 8080 del servidor.

Guión ejemplo de Enmascaramiento de IP con iptables.

El guión que mostramos en la tabla a continuación considera que se dispone de dos interfaces: eth0 y eth1. Para nuestro ejemplo la red local se accede por la interfaz eth0 y la salida hacia Internet de hace por la interfaz eth1. Utilice Firestarer para configurar el enmascaramiento y muro contrafuegos siempre que sea posible. **Este guión NO es sustituto para un guión de muro corta-fuegos.**

Guión básico de Enmascaramiento de IP.

```
#!/bin/sh

# cargamos los módulos del kernel necesarios:
/sbin/modprobe ip_conntrack
/sbin/modprobe ip_conntrack_ftp
/sbin/modprobe ip_conntrack_irc
/sbin/modprobe ipt_REJECT
/sbin/modprobe ipt_REDIRECT
/sbin/modprobe ipt_TOS
/sbin/modprobe ipt_MASQUERADE
/sbin/modprobe ipt_LOG
/sbin/modprobe iptable_mangle
/sbin/modprobe iptable_nat
/sbin/modprobe ip_nat_ftp
/sbin/modprobe ip_nat_irc

# Habilitamos el reenvío de direcciones IP
if [ -e /proc/sys/net/ipv4/ip_forward ]; then
 echo 0 > /proc/sys/net/ipv4/ip_forward
fi

# Estableciendo política de reenvío del enmascaramiento
/sbin/iptables -t filter -P FORWARD DROP

# Reenvío de trafico intento-externo y externo-interno
/sbin/iptables -t filter -A FORWARD -d 0/0 -s 192.168.1.0/255.255.255.0
  -o eth0 -j ACCEPT
/sbin/iptables -t filter -A FORWARD -d 192.168.1.0/255.255.255.0 -j
  ACCEPT

# Enmascaramiento de todo el trafico saliente
# NOTA: recordemos que la salida hacia Internet es por
# la interfaz eth0
/sbin/iptables -t nat -A POSTROUTING -o eth1 -j MASQUERADE

# No enmascararemos tráfico externo
/sbin/iptables -t nat -A POSTROUTING -o eth1 -d 0/0 -j ACCEPT
```

```
# Permitir al tráfico de la red interna ir a donde sea
/sbin/iptables -t filter -A INPUT -s 192.168.1.0/255.255.255.0 -d 0/0 -j
  ACCEPT
/sbin/iptables -t filter -A OUTPUT -s 192.168.1.0/255.255.255.0 -d 0/0 -j
  ACCEPT
/sbin/iptables -t filter -A OUTPUT -p icmp -s 192.168.1.0/255.255.255.0
  -d 0/0 -j ACCEPT

# Re-direccionamiento hacia el puerto 8080 (donde Squid escucha
# peticiones) para cualquier petición originada desde la red
# local hacia servicios que utilicen protocolo http, https y ftp
# Pueden añadirse más re-direccionamientos a discreción del
# administrador.
# NOTA 1: recordemos que la red local se accede con la interfaz eth0

# HTTP
/sbin/iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 -j REDIRECT
  --to-port 8080
```

2.- Acceso por Autenticación.

Introducción.

Es muy útil el poder establecer un sistema de autenticación para poder acceder hacia Internet, pues esto permite controlar quienes si y quienes no accederán a Internet sin importar desde que máquina de la red local lo hagan. Sera de modo tal que tendremos un doble control, primero por dirección IP y segundo por nombre de usuario y clave de acceso.

Para tal fin nos valdremos de un programa externo para autenticar, como es *ncsa_auth*, de la NCSA (National Center for Supercomputing Applications), y que ya viene incluido como parte del paquete principal de Squid en la mayoría de las distribuciones actuales.

Este manual considera que usted ya ha leído previamente, a detalle y en su totalidad el manual "*Como configurar Squid: Servidor Proxy*" y que ha configurado exitosamente Squid como servidor proxy.

Software requerido.

Para poder llevar la cabo los procedimientos descritos en este manual y documentos relacionados, usted necesitará tener instalado al menos lo siguiente:

- squid-2.5.STABLE1
- httpd-2.0.x (Apache)

Procedimientos

Creación del fichero de claves de acceso.

Se requerirá la creación previa de un fichero que contendrá los nombres de usuarios y sus correspondientes claves de acceso (cifradas). El fichero puede localizarse en cualquier lugar del

sistema, con la única condición que sea asequible para el usuario *squid*.

Debe procederse a crear un fichero */etc/squid/claves*:

```
touch /etc/squid/claves
```

Salvo que vaya a utilizarse un guión a través del servidor web para administrar las claves de acceso, como medida de seguridad, este fichero debe hacerse leíble y escribible solo para el usuario *squid*:

```
chmod 600 /etc/squid/claves
chown squid:squid /etc/squid/claves
```

A continuación deberemos dar de alta las cuentas que sean necesarias, utilizando el comando *htpasswd -mismo que viene incluido en el paquete httpd-2.0.x-*. Ejemplo:

```
htpasswd /etc/squid/claves joseperez
```

Lo anterior solicitará teclear una nueva clave de acceso para el usuario *joseperez* y confirmar tecleando ésta de nuevo. Repita con el resto de las cuentas que requiera dar de alta.

Todas las cuentas que se den de alta de este modo son independientes a las ya existentes en el sistema. Al dar de alta una cuenta o cambiar una clave de acceso lo estará haciendo **EXCLUSIVAMENTE** para el acceso al servidor Proxy. Las cuentas son independientes a las que se tengan existentes en el sistema como serían *shell*, correo y Samba.

Parámetros en /etc/squid/squid.conf

Lo primero será especificar que programa de autenticación se utilizará. Localice la sección que corresponde a la etiqueta *auth_param basic program*. Por defecto no está especificado programa alguno. Considerando que *ncsa_auth* se localiza en */usr/lib/squid/ncsa_auth*, procederemos a añadir el siguiente parámetro:

```
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/claves
```

Nota importante: Si usted está utilizando **software obsoleto** como Squid-2.4STABLE y versiones anteriores, que es utilizado en distribuciones **obsoletas** como Red Hat™ Linux 7.x y 8.0, Yellow Dog™ Linux 2.x, Mandrake Linux 8.x y 9.0, etc., es importante saber que para definir el programa de autenticación se utiliza otro parámetro muy distinto:

```
authenticate_program /usr/lib/squid/ncsa_auth /etc/squid/claves
```

/usr/lib/squid/ncsa_auth corresponde a la localización de el programa para autenticar y */etc/squid/claves* al fichero que contiene las cuentas y sus claves de acceso.

El siguiente paso corresponde a la definición de una *Lista de Control de Acceso*. Especificaremos una denominada *passwd* la cual se configurará para utilizar obligatoriamente la autenticación para poder acceder a Squid. Debe localizarse la sección de *Listas de Control de Acceso* y añadirse la

siguiente línea:

```
acl password proxy_auth REQUIRED
```

Habiendo hecho lo anterior, deberemos tener en la sección de *Listas de Control de Acceso* algo como lo siguiente:
Listas de Control de Accesos: autenticación.

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl redlocal src 192.168.1.0/255.255.255.0
acl password proxy_auth REQUIRED
```

Procedemos entonces a modificar la regla de control de accesos que ya teníamos para permitir el acceso a Internet. Donde antes teníamos lo siguiente:

```
http_access allow redlocal
```

Le añadimos *passwd*, la definición de la *Lista de Control de Acceso* que requiere utilizar clave de acceso, a nuestra regla actual, de modo que quede como mostramos a continuación:

```
http_access allow redlocal password
```

Habiendo hecho lo anterior, la zona de reglas de control de acceso debería quedar más o menos de este modo:
Reglas de control de acceso: Acceso por clave de acceso.

```
#
# INSERT YOUR OWN RULE(S) HERE TO allow ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow redlocal password
http_access deny all
```

Finalizando procedimiento.

Finalmente, solo bastará reiniciar Squid para que tomen efecto los cambios y podamos hacer pruebas.

```
service squid restart
```

3.- Restricción de acceso a sitios Web.

Introducción.

Denegar el acceso a ciertos sitos Web permite hacer un uso más racional del ancho de banda con el que se dispone. El funcionamiento es verdaderamente simple, y consiste en denegar el acceso a nombres de dominio o direcciones Web que contengan patrones en común.

Este manual considera que usted ya ha leído previamente, a detalle y en su totalidad el manual "*Como configurar Squid: Servidor Proxy*" y que ha configurado exitosamente Squid como servidor proxy.

Software requerido.

Para poder llevar la cabo los procedimientos descritos en este manual y documentos relacionados, usted necesitará tener instalado al menos squid-2.5STABLE1.

Definiendo patrones comunes.

Lo primero será generar una lista la cual contendrá direcciones Web y palabras usualmente utilizadas en nombres de ciertos dominios. Ejemplos:

```
www.sitioporno.com
www.otrositioporno.com
sitioindeseable.com
otrositioindeseable.com
napster
sex
porn
mp3
xxx
adult
warez
celebri
```

Esta lista, la cual deberá ser completada con todas las palabras (muchas de está son palabras obscenas en distintos idiomas) y direcciones Web que el administrador considere pertinentes, la guardaremos como */etc/squid/sitiosdenegados*.

Parámetros en /etc/squid/squid.conf

Debemos definir una *Lista de Control de Acceso* que a su vez defina al fichero */etc/squid/sitiosdenegados*. Esta lista la denominaremos como *"sitiosdenegados"*. De modo tal, la línea correspondiente quedaría del siguiente modo:

```
acl sitiosdenegados url_regex "/etc/squid/sitiosdenegados"
```

Habiendo hecho lo anterior, deberemos tener en la sección de *Listas de Control de Acceso* algo como lo siguiente:

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl redlocal src 192.168.1.0/255.255.255.0
acl password proxy_auth REQUIRED
acl sitiosdenegados url_regex "/etc/squid/sitiosdenegados"
```

A continuación especificaremos modificaremos una *Regla de Control de Acceso* existente agregando con un símbolo de *!* que se denegará el acceso a la *Lista de Control de Acceso* denominada *sitiosdenegados*:

```
http_access allow redlocal !sitiosdenegados
```

La regla anterior permite el acceso a la *Lista de Control de Acceso* denominada *redlocal*, pero le niega el acceso a todo lo que coincida con lo especificado en la *Lista de Control de Acceso* denominada *sitiosdenegados*.

Ejemplo aplicado a una *Regla de Control de Acceso* combinando el método de autenticación explicado en el documento *Cómo configurar Squid: Acceso por Autenticación*:
Reglas de control de acceso: denegación de sitios.

```
#
# INSERT YOUR OWN RULE(S) HERE TO allow ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow redlocal password !sitiosdenegados
http_access deny all
```

Permitiendo acceso a sitios inocentes incidentalmente bloqueados.

Si por ejemplo el incluir una palabra en particular afecta el acceso a un sitio Web, también puede generarse una lista de dominios o palabras que contengan un patrón pero que consideraremos como apropiados.

Como ejemplo: vamos a suponer que dentro de la *Lista de Control de Acceso* de sitios denegados está la palabra *sex*. esta denegaría el acceso a cualquier nombre de dominio que incluya dicha cadena de caracteres, como extremesex.com. Sin embargo también estaría bloqueando a sitios como sexualidadjovel.cl, el cual no tiene que ver en lo absoluto con pornografía, sino orientación sexual para la juventud. Podemos añadir este nombre de dominio en un ficheros que denominaremos */etc/squid/sitios-inocentes*.

Este fichero será definido en una *Lista de Control de Acceso* del mismo modo en que se hizo anteriormente con el fichero que contiene dominios y palabras denegadas.

```
acl inocentes url_regex "/etc/squid/sitios-inocentes"
```

Para hacer uso de el fichero, solo bastará utilizar la expresión **!** en la misma línea utilizada para la *Regla de Control de Acceso*establecida para denegar el mismo.

```
http_access allow all inocentes
```

La regla anterior especifica que se denegará el acceso a todo lo que comprenda la *Lista de Control de Acceso* denominada *denegados* **excepto** lo que comprenda la *Lista de Control de Acceso* denominada *inocentes*. es decir, se podrá acceder sin dificultad a www.sexualidadjoven.cl manteniendo la restricción para la cadena de caracteres *sex*.

Finalizando procedimiento.

Finalmente, solo bastará reiniciar Squid para que tomen efecto los cambios y podamos hacer pruebas.

```
service squid restart
```

4.- Restricción de acceso a contenido por extensión.

Introducción.

Denegar el acceso a ciertos tipos de extensiones de fichero permite hacer un uso más racional del ancho de banda con el que se dispone. El funcionamiento es verdaderamente simple, y consiste en denegar el acceso a ciertos tipos de extensiones que coincidan con lo establecido en una *Lista de Control de Acceso*.

Este manual considera que usted ya ha leído previamente, a detalle y en su totalidad el manual "*Como configurar Squid: Servidor Proxy*" y que ha configurado exitosamente Squid como servidor proxy.

Software requerido.

Para poder llevar la cabo los procedimientos descritos en este manual y documentos relacionados, usted necesitará tener instalado al menos squid-2.5STABLE1.

Definiendo elementos de la Lista de Control de Acceso.

Lo primero será generar una lista la cual contendrá direcciones Web y palabras usualmente utilizadas en nombres de ciertos dominios. Ejemplos:

```
\.avi$
\.mp4$
\.mp3$
\.mp4$
\.mpg$
\.mpeg$
\.mov$
\.ra$
\.ram$
\.rm$
\.rpm$
\.vob$
\.wma$
\.wmv$
\.wav$
\.doc$
\.xls$
\.mbd$
\.ppt$
\.pps$
\.ace$
\.bat$
\.exe$
\.lnk$
\.pif$
\.scr$
\.sys$
\.zip$
\.rar$
```

Esta lista, la cual deberá ser completada con todas las extensiones de fichero que el administrador considere pertinentes, la guardaremos como */etc/squid/listaextensiones*.

Parámetros en /etc/squid/squid.conf

Debemos definir una *Lista de Control de Acceso* que a su vez defina al fichero */etc/squid/listaextensiones*. Esta lista la denominaremos como *"listaextensiones"*. De modo tal, la línea correspondiente quedaría del siguiente modo:

```
acl listaextensiones urlpath_regex "/etc/squid/listaextensiones"
```

Habiendo hecho lo anterior, deberemos tener en la sección de *Listas de Control de Acceso* algo como lo siguiente:

```
#
# Recommended minimum configuration:
acl all src 0.0.0.0/0.0.0.0
```

```
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl redlocal src 192.168.1.0/255.255.255.0
acl password proxy_auth REQUIRED
acl sitiosdenegados url_regex "/etc/squid/sitiosdenegados"
acl listaextensiones urlpath_regex "/etc/squid/listaextensiones"
```

A continuación especificaremos modificaremos una *Regla de Control de Acceso* existente agregando con un símbolo de *!* que se denegará el acceso a la *Lista de Control de Acceso* denominada *listaextensiones*:

```
http_access allow redlocal !listaextensiones
```

La regla anterior permite el acceso a la *Lista de Control de Acceso* denominada *redlocal*, pero le niega el acceso a todo lo que coincida con lo especificado en la *Lista de Control de Acceso* denominada *listaextensiones*.

Ejemplo aplicado a una *Regla de Control de Acceso* combinando el método de autenticación explicado en el documento *Cómo configurar Squid: Acceso por Autenticación* y el de denegación hacia sitios web explicado en el documento *Cómo configurar Squid: Restricción de acceso a sitios Web*:
Reglas de control de acceso: denegación de extensiones.

```
#
# INSERT YOUR OWN RULE(S) HERE TO allow ACCESS FROM YOUR CLIENTS
#
http_access allow localhost
http_access allow redlocal password !sitiosdenegados !listaextensiones
http_access deny all
```

Finalizando procedimiento.

Finalmente, solo bastará reiniciar Squid para que tomen efecto los cambios y podamos hacer pruebas.

```
service squid restart
```

5.- Restricción de acceso por horarios.

Introducción.

Denegar el acceso a ciertos en ciertos horarios permite hacer un uso más racional del ancho de banda con el que se dispone. El funcionamiento es verdaderamente simple, y consiste en denegar el acceso en horarios y días de la semana.

Este manual considera que usted ya ha leído previamente, a detalle y en su totalidad el manual "*Como configurar Squid: Servidor Proxy*" y que ha configurado exitosamente Squid como servidor proxy.

Software requerido.

Para poder llevar la cabo los procedimientos descritos en este manual y documentos relacionados, usted necesitará tener instalado al menos squid-2.5STABLE1.

Procedimientos

La sintaxis para crear *Listas de control de acceso* que definan horarios es la siguiente:

```
acl [nombre del horario] time [días de la semana] hh:mm-hh:mm
```

Los días de la semana se definen con letras, las cuales corresponden a la primera letra del nombre en inglés, de modo que se utilizarán del siguiente modo:

- **S** - Domingo
- **M** - Lunes
- **T** - Mastes
- **W** - Miercoles
- **H** - Jueves
- **F** - Viernes
- **A** - Sábado

Ejemplo:

```
acl semana time MTWHF 09:00-21:00
```

Esta regla define a la lista *semana*, la cual comprende un horario de 09:00 a 21:00 horas desde el Lunes hasta el Viernes.

Este tipo de listas se aplican en las *Reglas de Control de Acceso* con una mecánica similar a la siguiente: se permite o deniega el acceso en el horario definido en la *Lista de Control de Acceso* denominada *X* para las entidades definidas en la *Lista de Control de Acceso* denominada *Y*. Lo anterior expresado en una *Regla de Control de Acceso*, quedaís del siguiente modo:

```
http_access [allow | deny] [nombre del horario] [lista de entidades]
```

Ejemplo: Se quiere establecer que los miembros de la *Lista de Control de Acceso* denominada *clasematutina* tengan permitido acceder hacia Internet en un horario que denominaremos como *matutino*, y que comprende de lunes a viernes de 09:00 a 15:00 horas.

La defincón para le horario correspondería a:

```
acl clasematutina src 192.168.1.0/255.255.255.0
acl matutino time MTWHF 09:00-15:00
```

La definición de la *Regla de Control de Acceso* sería:

```
http_access allow matutino clasematutina
```

Lo anterior, en resumen, significa que quienes conformen *clasematutina* podrán acceder a Internet de Lunes a Viernes de 09:00-15:00 horas.

Más ejemplos.

Restringiendo el tipo de contenido.

Como se explica en el documento *"Cómo configurar Squid: Restricción de acceso a contenido por extensión"*, es posible denegar acceso a cierto tipo de contenido de acuerdo a su extensión. Igual que con otras funciones, se requiere una *Lista de Control de Accesoy* una *Regla de Control de Acceso*

Si se necesita una lista denominada *musica* que defina a todos los ficheros con extensión .mp3, utilizaríamos lo siguiente:

```
acl clasematutina src 192.168.1.0/255.255.255.0
acl musica urlpath_regex \.mp3$
```

Si queremos denegar el acceso al todo contenido con extensión .mp3, la regla quedaría del siguiente modo:

```
http_access allow clasematutina !musica
```

Combinando reglas de tiempo y contenido.

Si por ejemplo queremos restringir parcialmente el acceso a cierto tipo de contenido a ciertos horarios, pueden combinarse distintos tipos de reglas.

```
acl clasematutina src 192.168.1.0/255.255.255.0
acl matutino time MTWHF 09:00-15:00
acl musica urlpath_regex \.mp3$

http_access allow matutino clasematutina !musica
```

La *Regla de Control de Acceso* anterior especifica **acceso permitido** a en el horario definido como *matutino* a quienes integran la *Lista de Control de Acceso* denominada *clasematutina* a todo contenido [por omisión] **excepto** a los contenidos que coincidan con los definidos en la *Lista de Control de Acceso* denominada *musica*.

Finalizando procedimiento.

Finalmente, solo bastará reiniciar Squid para que tomen efecto los cambios y podamos hacer pruebas.

```
service squid restart
```

Apéndice: Listas y reglas de control de acceso para Squid.

Opciones que habilitan función de autenticación en squid-2.5.STABLEx (Red Hat Linux 9)

```
auth_param basic children 5
auth_param basic realm Servidor Proxy-Cache Squid
auth_param basic credentialsttl 2 hours
# anteriores ya están descomentadas.
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/
squidpasswords
```

NOTA: Red Hat Linux 8.0 y 7.x utilizan *'authenticate_program'* en lugar de *'auth_param basic program'*

Para generar el fichero de contraseñas correspondiente, se debe utilizar el siguiente comando:

```
htpasswd -c /etc/squid/squidpasswords nombre_usuario
```

Para modificar las contraseñas del fichero de contraseñas, se debe utilizar el siguiente comando:

```
htpasswd /etc/squid/squidpasswords nombre_usuario
```

El fichero */etc/squid/squidpasswords* **debe** ser leíble para el usuario squid:

```
chown squid:squid /etc/squid/squidpasswords
```

Reglas aplicadas

```
# Lista que define método de autenticación:

acl password proxy_auth REQUIRED

# Listas de control de acceso por defecto:
acl all src 0.0.0.0/0.0.0.0
acl localhost src 127.0.0.1/255.255.255.255

# Listas que definen conjuntos de maquinas
acl redlocal src "/etc/squid/redlocal"
acl privilegiados src "/etc/squid/privilegiados"
acl restringidos src "/etc/squid/restringidos"
acl administrador src 192.168.1.254

# Listas que definen palabras contenidas en un URL
acl porno url_regex "/etc/squid/porno"
# Contenido:
#
# sex
# porn
# girl
# celebrit
# extasis
# drug
# playboy
# hustler

# Lista de sitios inocentes que accidentealmente sean bloqueados
acl noporno url_regex "/etc/squid/noporno"
# Contenido:
#
# missingheart
```

```
# wirelessexcite
# msexchange
# msexcel
# freetown
# geek-girls
# adulteducation

# Listas que definen tipos de extensiones

# Define uan lista estricta de extensiones prohibidas
acl multimedia urlpath_regex "/etc/squid/multimedia"
# Contenido:
#
# \.mp3$
# \.avi$
# \.mov$
# \.mpg$
# \.bat$
# \.pif$
# \.sys$
# \.lnk$
# \.scr$
# \.exe$

# Define una lista moderada de extensiones prohibidas
acl peligrosos urlpath_regex "/etc/squid/peligrosos"
# Contenido:
#
# \.bat$
# \.pif$
# \.sys$
# \.lnk$
# \.scr$
# \.exe$

# Define una sola extensión
acl realmedia urlpath_regex \.rm$

# Reglas de control de acceso

# Regla por defecto:
http_access allow localhost

# Ejemplos de reglas de control de acceso
http_access allow restringidos password !porno !multimedia
http_access allow redlocal password !porno !peligrosos
http_access allow privilegiados password !peligrosos
http_access allow administrador

http_access allow noporno all

# Regla por defecto:
http_access deny all
```