Squid, un proxy caché para GNU/Linux

24 de febrero de 2007

Resumen

En este documento se describe la instalación y configuración del proxy-cache squid, para controlar el acceso de una red local a Internet. Esta documentación se elaboró para el curso *Máquinas virtuales para la puesta en marcha de un portal educativo* organizado por el CEP de Sevilla en Septiembre de 2006.

1. Visión general

Como alternativa al software comercial existente, apareció Squid. Su funcionamiento se basa en guardar las peticiones que hacen los usuarios a servidores web remotos. Cuando un usuario quiere acceder a una página la solicita a Squid, que se encarga de acceder al servidor web remoto. Una vez obtenida, la reenvía al usuario, guardando una copia. En el caso que otro usuario solicite de nuevo esa página, únicamente tendrá que recuperarla de su disco local y servirla.

Otra función que realiza Squid es la de proporcionar un servicio de proxy a ordenadores que necesiten acceder a Internet a través de algún tipo de cortafuegos. Por eso es común denominar a Squid como un proxy caché, al unir las dos funcionalidades que presenta.

Squid puede almacenar datos de los protocolos HTTP, FTP, Gopher y DNS. El tener un servidor de caché especializado puede reducir considerablemente el uso que se haga del ancho de banda disponible. En lugar de descargar páginas repetidamente, se comprobará si la página del servidor remoto es más nueva que la que tiene almacenada en disco. De no ser así, no se molestará en descargarla.

2. Conceptos sobre cachés

Los servidores que actúan de proxy-caché se pueden configurar de varias formas. La forma más simple es un solo servidor proxy-caché en la red en el que todos los ordenadores pertenecientes a esa red accederán a este servidor, que será el que almacenará todos los datos. Cuando un usuario solicita al servidor una página, éste comprueba si fue actualizada desde que fue almacenada. Si tiene la versión actualizada ahorra al usuario final la descarga de la misma proporcionándosela directamente.

Otro método de configurar la salida a Internet de una red de ordenadores es creando una jerarquía de servidores proxy-caché. Los servidores en un nivel superior a un servidor son denominados padres (*parent*) y los que se encuentran al mismo nivel son hermanos o iguales (*sibblings*, *neighbor* o *peer*).

Cuando Squid obtiene una petición de un cliente, comprueba si el objeto solicitado (página, gráfico o fichero) está en el disco del servidor. Si está, comprueba que el objeto no está caducado y procede a enviarlo al cliente. Si, por el contrario, el objeto no está o ha caducado, comprueba que otras cachés (padres o hermanas) lo tengan. Lo hace a su vez enviando paquetes UDP a esas máquinas con la URL.

3. Instalación

Se instala simplemente con:

```
apt-get install squid
```

Los ficheros y directorios más importantes son:

- En el directorio `/etc/squid` se guardan los ficheros de configuración. Específicamente en el fichero `squid.conf` se encuentra la mayor parte de ella.

- Una parte importante de ficheros se encuentran en `/usr/lib/squid`, pero no tendremos que preocuparnos de ellos por ahora.
- La documentación se encuentra en `/usr/share/doc/squid-x.x.x/`
- En `/var/spool/squid` se van a encontrar las páginas "cacheadas", es decir, las traídas desde Internet y que se almacenan para la próxima vez que las solicite alguien y no hayan cambiado.
- En `/var/log/squid` se guardan los accesos de nuestros usuarios a Internet a través del proxy, así como los posibles errores que hayan ocurrido.

4. Configuración de Squid

El archivo de configuración que utiliza Squid es `/etc/squid/squid.conf`, pudiendo encontrarse en otras localizaciones dependiendo de la instalación. Este archivo está ampliamente comentado por lo que no lo analizaremos de forma detallada, sino que haremos un rápido recorrido por el fichero de configuración centrándonos en los aspectos que consideremos más importantes.

Una de las primeras directivas de configuración que aparece es:

```
http_port 3128
```

que indica el puerto en el que va a estar escuchando Squid.

Como ya hemos descrito, Squid es un proxy-caché y pueden existir elementos que no queramos almacenar. Esto puede conseguirse a través del fichero de configuración. Con la siguiente línea no almacenaríamos en caché ningún objeto que se encuentre en la ruta `cgi-bin`:

```
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY
```

Posteriormente veremos con más detalle la sintaxis y usos de la directiva `acl` para la creación de clases.

Es posible también definir parámetros de uso de memoria. Si queremos definir la cantidad de memoria RAM que deseamos asignar a las funciones de Squid con un valor de 8 Mb[1]

```
cache_mem 8 MB
```

Es posible también definir cuando se empieza a eliminar archivos de la caché. Cuando la caché llega al total del porcentaje de `cache_swap_high` Squid comienza a eliminar los elementos almacenados menos utilizados hasta que llega al total del porcentaje `cache_swap_low`.

```
cache_swap_low 90
cache_swap_high 95
```

[1]Con las configuraciones actuales y en función del uso de nuestra máquina, en general, debemos optar por un valor mayor.

Otra alternativa para regular el caché es configurarlo para que no almacene archivos que tengan un tamaño mayor que el indicado:

```
maximum_object_size 4096 KB
```

Ya hemos comentado que el caché de Squid es un espacio en disco reservado para almacenar los distintos objetos que se piden a través del proxy. Será necesario definir el lugar donde se va a almacenar el caché[2].

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

El formato genérico de esta directiva es:

```
cache_dir tipo directorio Mbytes L1 L2 [options]
```

- `tipo`. Tipo de sistema de almacenamiento a utilizar (ufs es el único que está definido por defecto en la instalación).
- `directorio`. Ruta del directorio que se va a utilizar para guardar los datos del caché.
- `Mbytes`. Cantidad de espacio en disco que se va a utilizar para el caché. Si queremos que utilice el disco entero es recomendable poner aquí un 20 % menos del tamaño.
- `L1`. Número de subdirectorios de primer nivel que serán creados bajo directorio.
- `L2`. Número de subdirectorios de segundo nivel que serán creados bajo cada subdirectorio de primer nivel.

Una consideración a tener en cuenta es que el contenido de este directorio va a cambiar con frecuencia, siendo recomendable colocarlo en una partición separada por varias razones:

- La caché podría sobrepasar al resto del sistema de archivos o de la partición que comparte con otros procesos.
- Cuanto más cambie un sistema de archivos, mayores son también las posibilidades de que se encuentre dañado. Mantener la caché en una partición limita la parte de su sistema completo de archivos que resulta dañado.

También es posible configurar la localización de los archivos de log así como la información general de la caché[3]

```
access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log /var/log/squid/store.log
```

[2]El valor de 100MB es escaso para los discos duros actuales, un valor de 1GB puede ser mejor si nuestro disco lo permite.

[3]Los ficheros de contabilidad que deja, pueden ser monitorizados para impedir accesos a Internet no deseados. Un ejemplo real es el de un organismo que manda semanalmente a los usuarios de Internet un fichero con los accesos en ese periodo. El usuario se siente controlado y es más responsable con sus accesos.
Una herramienta sencilla de configurar y muy útil para obtener estadísticas sobre las páginas visitadas es `sarg`.

5. Control de acceso

Otro aspecto importante en la configuración de Squid son las listas de control de acceso o ACL[4]. Una de sus principales funciones es la de permitir o denegar el acceso a la caché, aunque no se queda aquí. Las ACL pueden usarse también para definir las jerarquías de caché.

El procedimiento que se sigue es definir las distintas ACL y posteriormente se permite o deniega el acceso a una determinada función de la caché. La opción de configuración encargada es `http_access`, que permite o deniega al navegador web el acceso a Squid. Es importante tener en cuenta que Squid lee las directivas de arriba a abajo para determinar qué regla aplicar.

Veamos un primer ejemplo de uso. Supongamos que disponemos de una red de clase C (con direcciones IP dentro de la red 172.26.0.0) y que solo queremos permitir el acceso a Internet a través de Squid a estas máquinas.

```
acl hostpermitidos src 172.26.0.0/255.255.255.0
acl all src 0.0.0.0/0.0.0.0
http_access allow hostpermitidos
http_access deny all
```

Las dos primeras líneas de este ejemplo crean las ACL `hostpermitidos` y `all`.

Cuando utilicemos un fichero como origen de los datos de la ACL deberá contener un elemento por línea. Algunos de los tipos de ACL que podemos utilizar en esta directiva son:

- Direcciones IP de los clientes. Especifica la dirección IP local, dirección de red o rango de direcciones a buscar

```
acl nombreACL src dirIP/máscara
acl nombreACL src dirIP1-dirIP2/máscara
```

- Dirección IP de la URL destino. Especifica la dirección IP de la máquina remota, dirección de red o rango de dirección a buscar

```
acl nombreACL dst dirIP/máscara
acl nombreACL dst dirIP1-dirIP2/máscara
```

- Dominio de la máquina cliente. Especifica el `host.dominio.extensión` o bien el `dominio.extensión` a buscar

```
acl nombreACL srcdomain nombreDominio
```

- Dominio de la máquina destino. Especifica el `host.dominio.extensión` o bien el `dominio.extensión` a buscar

```
acl nombreACL dstdomain nombreDominio
```

- Expresión regular que concuerda con el nombre del cliente

```
acl nombreACL srcdom_regex [-i] expresión
```

[4] *Access Control List*

- Expresión regular que concuerda con el nombre del servidor

```
acl nombreACL dstdom_regex [-i] expresión
```

- Control por día y hora. Especifica la información de tiempo a buscar

```
acl nombreACL time [día] [h1:m1-h2:m2]
M - Lunes
T - Martes
W - Miércoles
H - Jueves
F - Viernes
A - Sábado
S - Domingo
h1:m1 < h2:m2
```

- Expresión regular que concuerda con la URL completa

```
acl aclname url_regex [-i] ^http://expresión
```

- Expresión regular que concuerda con la ruta de la URL

```
acl aclname urlpath_regex [-i] \.gif$
```

La primera de las opciones permite decidir en qué ACL se encuentra la dirección IP del usuario. También podemos decidir sobre aspectos como el tiempo actual o el sitio al que se dirigen. Para más información sobre estos y otros tipos de ACL recomendamos acceder al fichero `/etc/squid/squid.conf`.

Una vez definidas las ACL entra en juego la directiva `http_access`, que se utiliza para permitir o denegar el acceso de una determinada ACL, siempre y cuando el cliente utilice el método HTTP para solicitar el objeto al servidor web remoto.

Hay que tener en cuenta que la lectura se realiza de arriba hacia abajo, y se para en la primera coincidencia para decidir si permitir o denegar la petición. En el ejemplo anterior, Squid verá que la primera línea `http_access` se cumple, y procederá a aplicarla. En este caso, permitirá el acceso y ejecutará la petición.

Pero tengamos en cuenta el siguiente ejemplo:

```
acl hostpermitidos src 172.26.0.0/255.255.255.0
acl all src 0.0.0.0/0.0.0.0
http_access deny all
http_access allow hostpermitidos
```

En este caso no funcionará, ya que Squid aplicará la primera coincidencia —la primera línea— y denegará el acceso.

Las ACL son especialmente útiles cuando queremos prohibir el acceso a una lista de sitios inapropiados (enlaces a páginas web con contenido pornográfico, violento, etc.). Squid no está optimizado para gestionar una larga lista de sitios, pero puede gestionar un número concreto de sitios sin problemas.

```
acl porno dstdomain playboy.com sex.com sevillafc.es# :-P
acl hostpermitidos src 172.26.0.0/255.255.255.0
acl all src 0.0.0.0/0.0.0.0
http_access deny porno
http_access allow hostpermitidos
http_access deny all
```

En este caso, las direcciones que serán consideradas como inadecuadas son las que tienen los dominios `playboy.com` o `sex.com`. Estas URL tendrán filtrado el acceso y no será posible acceder a ellas, tal como indica la directiva `http_access deny porno`. Si se piden otras URL, Squid pasará a evaluar la siguientes directivas. Por tanto, si el cliente se conecta dentro del rango permitido se cursará la petición. De lo contrario, la petición será rechazada.

La configuración que viene por defecto es denegar todos los accesos, mediante:

```
http_access deny all
```

Obviamente deberemos al menos incluir algún grupo que pueda acceder, porque si no, nuestro proxy sería innecesario.

La limitación que tiene restringir acceso a diferentes dominios de la manera anterior es que cada vez que añadimos un dominio, hay que reiniciar squid. Para evitar esto puede utilizarse un fichero donde ir añadiendo los dominios prohibidos, mediante la directiva:

```
acl adultos dstdomain "/etc/squid/adultos.txt"
```

Y deberíamos crear el correspondiente fichero con una línea por dominio.

Una última observación, este método considera exclusivamente los dominios. Para evitar conexiones especificando la IP de la máquina, utilizaremos la directiva `dst acl`.

Suponiendo que ya hemos definido nuestra política de acceso, arrancamos el servicio:

```
#/etc/init.d/squid start
```

La primera vez que lo ejecutemos tardará un ratito porque tiene que construir sus índices para el almacenamiento de páginas.

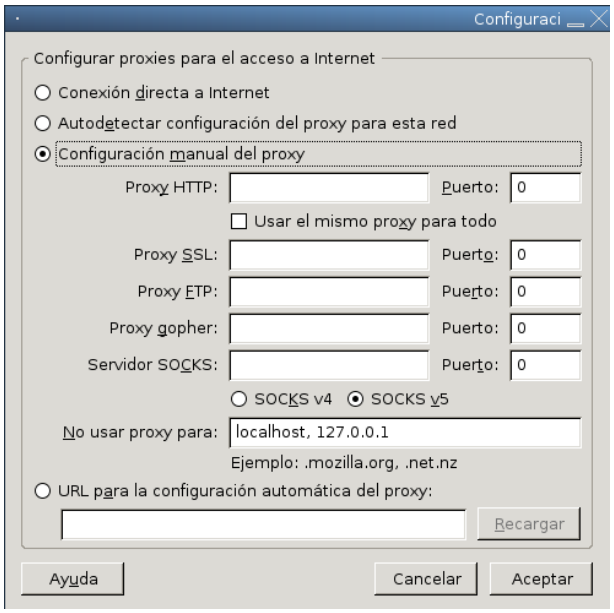
```
root@guadalinex:/usr/lib/squid# /etc/init.d/squid start
Starting proxy server: Creating squid spool directory structure
2005/02/15 14:18:23| Creating Swap Directories
squid.
root@guadalinex:/usr/lib/squid#
```

En función de la versión de squid puede que nos aparezca un error al ponerlo en marcha, en general se debe a que hay que configurar correctamente la directiva `visible_hostname` que aparece comentada por defecto.

6. Configuración de los clientes

El cliente lo configuraremos de la siguiente forma:

- Si es Mozilla-firefox, seleccionamos `Editar` → `Preferencias` → `General` → `Configuración de conexión` → `Configuración Manual del Proxy` y en los distintos protocolos ponemos el host correspondiente con el puerto 3128 que es por el que escucha el squid peticiones de sus clientes. Podemos ponerlo en todos los protocolos menos en el socks.



  Veremos que nuestros accesos a Internet son mucho más rápidos y el control que podemos llegar a tener es muy importante.

- Si trabajamos en modo consola y deseamos definir la variable de entorno[5] `http_proxy`, podemos usar:

  ```
  export http_proxy="http://ip_proxy:3128"
  ```

  Para ver que todo está bien:

  ```
  lynx http://www.iesmurgi.org
  ```

  Es útil, por ejemplo, para actualizar un sistema com `yum` o `apt-get` que accede a Internet a través de un proxy

7. Autenticación de usuarios

Existe la posibilidad —utilizada fundamentalmente en el entorno empresarial— de acceder a la navegación web mediante nombre de usuario y contraseña.

Para configurar squid de este modo hay que utilizar la siguiente directiva:

[5]Si añadimos la variable a algún script de arranque se tomará como valor por defecto. Desde GNOME o KDE también podemos configurarla.

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

Donde especificamos que el programa `ncsa_auth` realizará la autenticación y el fichero `/etc/squid/passwd` será donde se almacenará la información de autenticación [6].

Añadiríamos la ACL:

```
acl pass_web proxy_auth REQUIRED
```

Y para aplicarla podríamos poner:

```
http_access allow pass_web
http_access allow hostpermitidos
http_access deny all
```

De manera que sólo los usuarios autorizados e incluidos en `hostpermitidos` podrían navegar por Internet.

Para terminar hay que cerrar la navegación a través del puerto 80, para que todos los clientes deban configurar sus navegadores para acceder a Internet a través del proxy. Suponiendo que todo nuestro tráfico pasa a través de una máquina que actúa de cortafuegos, bastaría con hacer algo como:

```
iptables -A FORWARD -s 172.16.0.0/24 -i eth1 -p tcp \
--dport 80 -j DROP
```

8. Proxy transparente

En contraposición a la navegación a través de un proxy con autenticación que hemos visto anteriormente, existe la posibilidad de configurar squid para que todos los usuarios de una red naveguen a través del proxy sin necesidad de tener que configurar cada una de sus aplicaciones, esto es lo que se denomina *proxy transparente*. Es más, se hace para que naveguen a través del proxy sin saberlo o sin quererlo, asumiendo todas las restricciones que éste imponga.

En primer lugar hay que utilizar la siguiente línea de iptables:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT \
--to-port 3128
```

En el caso de que el proxy se encuentre en la misma máquina que el cortafuegos, en caso de que sea en otra máquina —por ejemplo con IP 192.168.44.44—, tendríamos que poner:

```
iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT \
--to 192.168.44.44:3128
```

Y modificar la línea `http_port 3128`, de manera que quede:

```
http_port 3128 transparent
```

En versiones anteriores a la 2.6, era necesario añadir varias directivas del tipo `httpd_accel`, pero actualmente se consigue el funcionamiento en modo transparente con esta simple modificación

[6] squid no genera las entradas de este fichero, para ello deberemos utilizar un programa externo como htpasswd incluido en el paquete apache2-utils