

Squid + Webmin



```
aclSSL_portsport443acl
CONNECTmethodCONNECTac
lloc_subnetsrc192.168.
221.0/24aclloc_sysadmi
nsrc192.168.221.110/32
http_accessallowdomini
os_msnloc_sysadminhttp
_accessdenydominios_ms
nhttp_accessallowloc_s
ysadmin
```



Índice

1 Introducción	4
2 Instalación y configuración básica	7
2.1 Instalando los paquetes	7
2.2 Verificando la instalación	8
2.3 Configuración básica de squid	9
2.3.1 Introducción al archivo de configuración de squid	9
2.3.2 Validando los archivos de configuración de Squid	9
2.3.3 Configuración de parámetros de red	10
2.3.4 Configuración de parámetros administrativos	10
2.3.5 Configurando el uso del caché en memoria	11
2.3.6 Configuración y creación del directorio caché	11
2.3.7 Configurando el tamaño para los objetos en caché	12
2.3.8 Configuración de los parámetros de logs	13
2.3.9 Configurando las paginas de error de squid	14
2.3.10 Configuración de otros parámetros	15
2.4 Controlando el servicio squid	15
3 Configuración de los esquemas de control de acceso	17
3.1.1 Tipos de elementos de ACL	18
3.1.2 Sintaxis de los elementos de ACL	19
3.1.3 Reglas de control de acceso	20
3.2 Creando reglas de control de acceso basadas en direcciones IP	22
3.3 Creando reglas de acceso basadas en puertos, dominios, URLs y tipos MIME	23
3.3.1 Listas de control de acceso basadas en puertos y método HTTP	23
3.3.2 Listas de control de acceso basadas en dominios	24
3.3.3 Listas de control de acceso basadas en URLs	25
3.3.4 Listas de control de acceso basadas en tipos MIME	27
Controles de acceso basados en la petición de tipos MIME	27
Controles de acceso basados en la respuesta de tipos MIME	28
4 Squid con Webmin	31
4.1 Instalación de Webmin	31
4.2 Acceso a Webmin	32
4.3 Configuración de Squid	33
4.3.1 Configuración actual	34
4.3.2 Creación de ACLs y restricciones	36
4.3.3 Directrices para la ordenación de restricciones	37

<u>5 Apéndices</u>	38
<u>5.1 Chuleta resumen básico</u>	38
<u>5.2 Archivo de configuración por defecto sin comentarios ni líneas en blanco</u>	40
<u>5.3 Configurar navegador Firefox para usar el proxy</u>	41
<u>5.4 Ejemplo: permitir accesos a la red local</u>	41
<u>5.5 Ejemplo: denegar acceso a la prensa deportiva</u>	42
<u>5.6 Expresiones regulares</u>	44
<u>6 Fuentes</u>	44

1 Introducción

Características principales de Squid:

- ☐ Liberado bajo la Licencia GNU General Public License (GPL).
- ☐ Viene incluido y soportado en la mayoría de distribuciones GNU/Linux.
- ☐ Soporta los protocolos IPv4 e IPv6.
- ☐ Proxy para los protocolos HTTP, HTTPS, FTP y GOPHER.
- ☐ Caché de consultas DNS.
- ☐ Caché de contenido para aceleración web con soporte de diferentes sistemas de archivos para el almacenamiento del caché.
- ☐ Controles de acceso avanzados basados en ACLs.
- ☐ Soporta diferentes esquemas de autenticación.
- ☐ Soporta diferentes métodos de autorización.
- ☐ Registro de Logs y soporte SNMP.
- ☐ Soporte de plugins para autenticación de usuarios y grupos.
- ☐ Integración de filtros de URLs y contenido como squidGuard y DansGuardian.

Squid provee soporte para controles de accesos para los clientes basados en varios criterios de autenticación, como:

- ☐ Direcciones IP: listas, rangos, subredes.
- ☐ Direcciones MAC: sólo redes locales.
- ☐ Usuarios locales NCSA.
- ☐ Usuarios y Grupos LDAP: OpenLDAP, Active Directory.
- ☐ RADIUS.
- ☐ Kerberos.
- ☐ Active Directory (Single Sign On).
- ☐ NTLM (Single Sign On).
- ☐ PAM (Linux).
- ☐ Horarios.

Además Squid provee soporte robusto y extensible para controlar las peticiones basadas en el destino y el contenido, creando reglas para:

- ☐ URLs destino, ejemplo: `http://porn.com/downloads/free/`.
- ☐ Nombres de dominio DNS destino, ejem: `dl.fileshare.com`.
- ☐ Direcciones IP, ejemplo: `http://18.1.3.22/downloads/`.
- ☐ Expresiones regulares para las URLs destino, ejem: `.mp3`, `.torrent`, `.rar`, `.avi`.
- ☐ Tipos MIME para contenido multimedia: `audio/mpeg`, `application/x-messenger`, `application/x-flv`.
- ☐ Control de acceso para peticiones y respuestas HTTP: POST, GET.

Además, se integra el filtro de URLs **squidGuard**, cuya función es analizar los URLs solicitados por los clientes y compararlos contra una base de datos de listas negras y en base al resultado obtenido permitir o denegar el acceso al URL. Las características principales de squidGuard son:

- ☐ Liberado bajo la Licencia GNU General Public License (GPL).
- ☐ Ultra-rápido.
- ☐ Controlar el acceso a URLs para protocolos HTTP y HTTPS.
- ☐ Control basado en:
 - ☐ Direcciones IP.
 - ☐ Usuarios NCSA.
 - ☐ Usuarios y Grupos LDAP.
 - ☐ Usuarios y Grupos MySQL.
- ☐ Permite el uso de listas blancas para excluir sitios bloqueados (falsos positivos) por alguna categoría de lista negra.
- ☐ Permite redireccionar las peticiones denegadas a una página HTML informativa.

Para obtener información sobre las peticiones al proxy en tiempo real existen varias herramientas para el análisis de los logs de acceso de squid.

multitail

multitail es un programa de línea de comando para la visualización de múltiples archivos de log en tiempo real, además posee el soporte de coloreado de logs basado en esquemas. Incluye el esquema de colores para los logs de acceso de **squid**.

squidview

squidview es una interfaz en línea de comando para visualizar las conexiones activas del proxy squid. Algunas de sus funcionalidades son:

- ☐ Ver quien (usuario/host) esta navegando en tiempo real.
- ☐ Que sitios/urls son los que se están visitando en el preciso momento.
- ☐ Ver el número de conexiones y ancho de banda consumido.
- ☐

sarg

SARG es una herramienta de análisis de logs de Squid Mediante los reportes de uso web usted podrá obtener la siguiente información:

- ☐ Top Ten de sitios más visitados
- ☐ Reportes diarios, semanales y mensuales
- ☐ Accesos por usuarios
- ☐ Tiempos de navegación
- ☐ Descargas

calamaris

Calamaris genera reportes y estadísticas del uso del proxy, sus principales características son:

- ☐ Reportes web y por correo
- ☐ Total de peticiones realizadas al proxy
- ☐ Total de usuarios que usan el proxy
- ☐ Total de ancho de banda usado
- ☐ Cantidad de peticiones en caché
- ☐ Cantidad de ancho de banda ahorrado
- ☐ Porcentaje de ancho de banda ahorrado
- ☐ Otras estadísticas sobre dominios visitados, tipos de archivos descargados

2 Instalación y configuración básica

Los objetivos a cubrir en este capítulo se listan a continuación:

- ☐ Instalación de squid versión 3 usando el manejador de paquetes **apt**.
- ☐ Validación de la instalación de squid.
- ☐ Entender y validar la configuración del squid.
- ☐ Configurando parámetros generales de squid.
- ☐ Configurar y crear el caché de disco.
- ☐ Configurar los parámetros para los logs.
- ☐ Controlar el servicio squid.

2.1 Instalando los paquetes

Actualmente tenemos la opción de instalar la versión 2 o la 3

```
root@manolo-debian:/# apt-cache policy squid
squid:
  Instalados: (ninguno)
  Candidato:  2.7.STABLE9-4.1+deb7u1
  Tabla de versión:
    2.7.STABLE9-4.1+deb7u1 0
                        500 http://security.debian.org/ wheezy/updates/main amd64 Packages
    2.7.STABLE9-4.1 0
                        500 http://ftp.es.debian.org/debian/ wheezy/main amd64 Packages
root@manolo-debian:/# apt-cache policy squid3
squid3:
  Instalados: (ninguno)
  Candidato:  3.1.20-2.2+deb7u2
  Tabla de versión:
    3.1.20-2.2+deb7u2 0
                        500 http://ftp.es.debian.org/debian/ wheezy/main amd64 Packages
                        500 http://security.debian.org/ wheezy/updates/main amd64 Packages
```

Instalación de la v3:

```
root@manolo-debian:/# apt-get install squid3
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Se instalarán los siguientes paquetes extras:
  squid3-common
Paquetes sugeridos:
  squidclient squid-cgi resolvconf
Se instalarán los siguientes paquetes NUEVOS:
  squid3 squid3-common
0 actualizados, 2 se instalarán, 0 para eliminar y 193 no actualizados.
Necesito descargar 1.847 kB de archivos.
Se utilizarán 4.550 kB de espacio de disco adicional después de esta
operación.
¿Desea continuar [S/n]?
```

2.2 Verificando la instalación

El paquete **squid3** instala el ejecutable de squid en **/usr/sbin/squid3**. Normalmente no usaremos **squid3** en la línea de comandos salvo alguna excepción.

```
root@manolo-debian:/# ls -l /etc/squid3/
total 212
-rw-r--r-- 1 root root 1547 ago 28 2014 errorpage.css
-rw-r--r-- 1 root root 421 ago 28 2014 msntauth.conf
-rw-r--r-- 1 root root 206557 ago 28 2014 squid.conf
```

Los módulos o plugins de Squid son instalados en el directorio **/usr/lib/squid3**.

Al instalar squid se creó un usuario y un grupo llamado **proxy**. Es con los privilegios de este usuario con el que se ejecutará el proceso del proxy squid, por lo tanto dicho usuario y grupo deben tener privilegios sobre los archivos de configuración, logs y caché para que el proxy funcione apropiadamente.

Validamos la existencia del usuario y grupo:

```
root@manolo-debian:/# grep proxy /etc/{passwd,group}
/etc/passwd:proxy:x:13:13:proxy:/bin:/bin/sh
/etc/group:proxy:x:13:
```

Los permisos del archivo de configuración principal de squid son para root, sin embargo, otros tienen permiso de lectura sobre el archivo. Es importante que tome esto en consideración si almacena contraseñas en formato texto plano en dicho archivo.

```
root@manolo-debian:/# ls -l /etc/squid3/squid.conf
-rw-r--r-- 1 root root 206557 ago 28 2014 /etc/squid3/squid.conf
```


El usuario y grupo **proxy** requieren acceso al directorio del caché en el cual se almacenarán los de objetos descargados, por defecto es el directorio **/var/spool/squid3**. Asegúrese de que los permisos son correctos:

```
root@manolo-debian:/# ls -ld /var/spool/squid3
drwxr-xr-x 2 proxy proxy 4096 ago 28 2014 /var/spool/squid3
```

2.3 Configuración básica de squid

En esta sección veremos una introducción a los archivos de configuración de Squid, así como la configuración de los parámetros generales para una configuración básica de un proxy caché con Squid.

2.3.1 Introducción al archivo de configuración de squid

Squid mantiene sus archivos de configuración en el directorio **/etc/squid3/**. El archivo principal de configuración de squid es **/etc/squid3/squid.conf**.

El archivo de configuración de squid **/etc/squid3/squid.conf** incluye todas las directivas de configuración disponibles, **la mayoría están comentadas y tienen un valor predefinido**. Estos valores predefinidos son óptimos para la mayoría de las instalaciones y solo se tendrán que cambiar en caso de que la configuración lo requiera. En algunos otros casos se descomentan las directivas aún cuando se deje el valor predeterminado con propósitos de hacer explícitas las configuraciones.

Ya que no existe página de manual para el archivo de configuración *squid.conf*, en su lugar, la información de las directivas de configuración están incluidas como comentarios en el mismo archivo *squid.conf*. Se recomienda que antes de que haga cambios en el archivo *squid.conf* haga una copia de respaldo para que la pueda usar como manual de referencia.

```
root@manolo-debian:/# cp /etc/squid3/squid.conf /etc/squid3/squid.conf.bak
```

2.3.2 Validando los archivos de configuración de Squid

Cada vez que se modifique el archivo *squid.conf* se recomienda correr el programa *squid* con el parámetro **-k parse** para realizar una revisión de la sintaxis del archivo.

Si no regresa algún mensaje, entonces significa que el archivo de configuración está correctamente configurado.

```
root@manolo-debian:/# squid3 -k parse
2015/04/30 10:25:29| Processing Configuration File: /etc/squid3/squid.conf
(depth 0)
2015/04/30 10:25:29| Processing: acl manager proto cache_object
2015/04/30 10:25:29| Processing: acl localhost src 127.0.0.1/32 ::1
2015/04/30 10:25:29| Processing: acl to_localhost dst 127.0.0.0/8
0.0.0.0/32 ::1
2015/04/30 10:25:29| Processing: acl SSL_ports port 443
2015/04/30 10:25:29| Processing: acl Safe_ports port 80          # http
2015/04/30 10:25:29| Processing: acl Safe_ports port 21          # ftp
(...)
```

2.3.3 Configuración de parámetros de red

El parámetro `http_port` define el puerto en el que Squid escuchará peticiones HTTP de los clientes. Este es un parámetro requerido; el puerto predeterminado de Squid es el **3128**.

```
root@manolo-debian:/# cat /etc/squid3/squid.conf | grep http_port
http_port 3128
```

Si el sistema es multi-homed, es decir, tiene más de una interfaz de red y desea limitar en que redes dar el servicio de proxy, por ejemplo, para que solo escuche peticiones en la dirección IP 192.168.221.254 asignada a la interfaz que esta conectada a la LAN use:

```
http_port 192.168.221.254:3128
```

Debe recargar la configuración del proxy para que este cambio tome efecto.

La directiva `visible_hostname` define el nombre de host con el que squid se anunciará, y también será el nombre que aparecerá en las paginas de error. Defina un nombre válido, por ejemplo:

```
visible_hostname fwproxy.example.com
```

Squid por defecto usa los servidores DNS definidos en el archivo `/etc/resolv.conf` a menos que se definan los servidores DNS usando la directiva `dns_nameservers`. Si no desea que squid use los servidores DNS del sistema, defina la lista de servidores, por ejemplo:

```
dns_nameservers 192.168.221.253 192.168.221.252
```

Se recomienda que el servidor proxy squid tenga acceso a uno o más servidores DNS caché para acelerar el servicio de resolución de nombres. Puede instalar un servidor BIND en modo sólo-caché.

2.3.4 Configuración de parámetros administrativos

El parámetro `cache_effective_user` define el nombre del usuario con el que Squid operará, por ejemplo:

```
cache_effective_user proxy
```

Importante

Como buena práctica de seguridad nunca corra squid con los privilegios de **root**, use un usuario no privilegiado como **nobody** o uno dedicado como **proxy** o **squid**.

El parámetro `cache_mgr` define la dirección de correo del administrador del proxy. Esta dirección es usada en las páginas de error; además, si el proceso Squid muere, un email es enviado a la dirección usada.

Este parámetro viene predeterminado con la dirección de **root**, se recomienda que use una dirección válida ya que es la que los usuarios usarían para notificar problemas con el proxy, por ejemplo

```
cache_mgr proxy@example.com
```

2.3.5 Configurando el uso del caché en memoria

El parámetro `cache_mem` especifica la cantidad ideal de memoria que será usada para mantener el caché de los objetos entrantes en memoria.

Para nuestra instalación definiremos 64 MB para el cache de objetos en memoria.

```
| cache_mem 64 MB
```

El parámetro **cache_mem** NO especifica el tamaño máximo de memoria asignada al proceso squid. Solo especifica cuanta memoria usará para el caché de objetos.

Nota

Squid usa memoria para otras tareas. La memoria asignada para el caché de objetos es la tercera causa de consumo total de memoria.

Defina un valor no demasiado alto para el caché de memoria ya que podría afectar el rendimiento del proxy.

2.3.6 Configuración y creación del directorio caché

El parámetro `cache_dir` define el directorio para almacenar los objetos en caché de disco, este parámetro define la ruta del directorio, el tipo de cache de disco y su tamaño, su formato es:

```
| cache_dir Type Directory-Name Mbytes Level1 Level2 [options]
```

Nota

Puede especificar múltiples directorios para caché en disco, por ejemplo, para repartir el caché en varios discos duros y balancear el acceso al caché.

El tipo define el sistema de almacenamiento que squid usará para almacenar los objetos en disco. El tipo predeterminado es UFS.

Directory-Name define la ruta absoluta al directorio designado para el caché. En la mayoría de distribuciones GNU/Linux el directorio predeterminado es `/var/spool/squid`. En Debian/Ubuntu se usa `/var/spool/squid3`, por ejemplo:

```
| cache_dir ufs /var/spool/squid3 100 16 256
```

Mbytes define la cantidad de disco en MB (Mbytes) para el directorio definido. El valor predeterminado es 100 MB, por ejemplo:

```
| #cache_dir ufs /var/spool/squid3 100 16 256
```

Importante

- Recuerde que el tamaño de la caché influirá en el tamaño de memoria que squid usará del sistema.
- No ponga el tamaño del disco duro aquí. Mejor, si quiere que squid use todo el disco, reste el 20% de la capacidad total y use este valor.

Level1 define subdirectorios de primer nivel que serán creados en el directorio. El valor predeterminado es 16. *Level2* define el número de subdirectorios para el segundo nivel de subdirectorios. El valor predeterminado es 256.

Squid realiza muchas operaciones de I/O en disco, por lo que se recomienda tener el directorio de caché en un disco rápido, preferentemente que esté separado del disco en el que está instalado el sistema operativo.

Recuerde que puede tener múltiples cachés, es decir, puede tener varias líneas **cache_dir** y así distribuir la carga de acceso a disco de un solo caché en varios discos.

Un vistazo al directorio de caché:

```
root@manolo-debian:/# ls -l /var/spool/squid3
total 72
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 00
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 01
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 02
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 03
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 04
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 05
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 06
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 07
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 08
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 09
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0A
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0B
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0C
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0D
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0E
drwxr-x--- 258 proxy proxy 4096 abr 30 17:28 0F
-rw-r----- 1 proxy proxy 504 abr 30 17:39 swap.state
-rw-r----- 1 proxy proxy 72 abr 30 17:28 swap.state.new
```

2.3.7 Configurando el tamaño para los objetos en caché

El parámetro `maximum_object_size` define el tamaño máximo de los objetos que serán almacenados en el cache de disco. Los objetos mayores a este tamaño NO serán almacenados en disco. El valor está especificado en kilobytes, y su valor predeterminado es 4MB.

```
| maximum_object_size 4096 KB
```

Si desea mantener objetos mayores, incremente el tamaño; si desea incrementar la velocidad en lugar de ahorrar ancho de banda, mantenga un valor bajo.

2.3.8 Configuración de los parámetros de logs

El parámetro `logformat` define el formato para almacenar los logs de acceso. Con este parámetro podemos definir qué información almacenar sobre las peticiones realizadas por los clientes. Esta información será importante para ser usada por las herramientas de análisis de logs y generación de reportes de acceso que veremos en capítulos posteriores.

La directiva `logformat` tiene la sintaxis:

```
| logformat <name> <format specification>
```

El *name* es una etiqueta para el formato especificado. Las especificaciones de formato definen códigos de formato propios de squid. Este parámetro está comentado: se recomienda descomentar la línea predefinida, por ejemplo:

```
| logformat squid %ts.%03tu %6tr %>a %Ss/%03Hs %<st %rm %ru %un %Sh/%<A %mt
```

Nota

Descomentar la línea del formato `squid` y eliminar el espacio entre `squid` y `%t`

En la siguiente tabla se muestran algunos códigos de formato:

Tabla 3.1. Códigos de formato

Código	Descripción
>a	Client source IP address
ts	Seconds since epoch
tu	subsecond time (milliseconds)
tr	Response time (milliseconds)
%	a literal % character

El parámetro `access_log` define la ruta del archivo en el que se va a registrar la actividad de los clientes, además, se define el formato en el que se registrará la información, por ejemplo:

```
| access_log /var/log/squid3/access.log squid
```

Nota

Si no desea registrar la actividad de los clientes, use el argumento *none*.

Importante

El usuario proxy debe tener permisos de escritura sobre este archivo.

El parámetro `cache_log` define la ruta del archivo en la que se registrarán los eventos generados por el proceso squid. Cuando squid es iniciado, registra la información en este log. Si hubiese algún problema al leer ciertos archivos, esa información se registra en este log. Problemas con espacio insuficiente o en la autenticación la veremos en el archivo definido aquí.

```
| cache_log /var/log/squid3/cache.log
```

El parámetro `cache_store_log` define el archivo de log en el que se registran los objetos puestos en el caché, por ejemplo:

```
| cache_store_log /var/log/squid3/store.log
```

Mediante la opción `-k rotate` del comando `squid3` puede ordenar al proceso squid en ejecución que realice una rotación del log `access.log`. El parámetro `logfile_rotate` define el número de archivos `access.log` que se almacenarán antes de ser rotados.

Si ejecuta el comando **squid3 -k rotate** diariamente, probablemente querrá guardar los logs antiguos por una semana. Para hacerlo, cambie el valor a 7, por ejemplo:

```
| logfile_rotate 7
```

2.3.9 Configurando las paginas de error de squid

El parámetro `error_directory` define la ruta del directorio en el cual se almacenan los mensajes que son devueltos a los clientes web. Los mensajes por defecto están en inglés; si desea cambiar los mensajes a Español, cambie el parámetro `error_directory` por ejemplo:

```
| error_directory /usr/share/squid3/errors/Spanish
```

Nota

En la página de acceso denegado no se muestra la dirección de correo del administrador del caché. Si desea mostrar esa información, use la versión en inglés y modifique el HTML.

Las páginas de error son archivos de texto plano en HTML. La dirección de correo especificada en **cache_mgr** es incluida como dirección de contacto. Si desea agregar más información en la página de conexiones rechazadas, edite el archivo

`/usr/share/squid3/errors/Spanish/ERR_ACCESS_DENIED`

y al final del archivo, después de la etiqueta `` agregue algo así::

```
<P>
Si cree que el sitio fué bloqueado por equivocación contacte al
administrador del proxy en el correo
<A HREF="mailto:%w%W">%w</A> o marque al departamento de TI en la extensión
XXXX.
</P>
```

Nota

No incluya las etiquetas HTML de cierre `</HTML>` `</BODY>`, squid las cerrará.

Puede editar estos archivos para personalizar los mensajes devueltos a los clientes cuando entren a una pagina denegada, cuando no puedan acceder a un sitio cuando el servidor esta caído o cuando el puerto sea rechazado por la política del firewall.

2.3.10 Configuración de otros parámetros

El parámetro `ftp_user` define la dirección de correo que se usará como contraseña para las conexiones FTP anónimas. De esta forma, cuando un usuario haga una conexión FTP anónima a través del proxy Squid, se enviará automáticamente la contraseña usando la dirección de correo especificada.

Si va a usar Squid como proxy FTP, cambie el parámetro `ftp_user` a una dirección de correo válida, ya que algunos servidores FTP validan las direcciones, por ejemplo:

```
| ftp_user wwwproxy@example.com
```

Lea los comentarios del archivo de configuración `squid.conf` para más parámetros de configuración.

2.4 Controlando el servicio squid

Para controlar el servicio de Squid se recomienda que use el script de control de ejecución `/etc/init.d/squid3`. A continuación se describen las acciones principales de control del servicio Squid.

Para iniciar el servicio squid use el parametro `start`, por ejemplo:

```
| root@manolo-debian:/# /etc/init.d/squid3 start
| [ ok ] Starting Squid HTTP Proxy 3.x: squid3.
```

Muchas de las veces va a requerir realizar alguna configuración o cambio de ACLs o reglas de acceso y aplicar los cambios **sin tener que reiniciar squid**. Use el siguiente comando para recargar la configuración de squid:

```
| root@manolo-debian:/# squid3 -k reconfigure
```

Nota

No olvide validar la configuración con `-k parse` antes de realizar la reconfiguración.

Si tiene problemas al ejecutar el servicio squid, se recomienda que ejecute el programa `/usr/sbin/squid3` manualmente con la opción `-d` y un nivel alto de debug, por ejemplo:

```
| root@manolo-debian:/# squid3 -d 9
```

Importante

Al usar la opción `-d`, squid registrará todos los eventos en la salida estándar (`stderr`), de esta forma podrá encontrar mensajes importantes o de error que pueden indicarle la causa del problema.

Para detener el servicio squid usando los scripts rc, ejecute:

```
| root@manolo-debian:/# /etc/init.d/squid3 stop
```

Si ejecutó squid3 en primer plano, presione **Ctrl+C** para cancelar su ejecución.

Si realizó cambios significativos en la configuración de squid, como cambio de dirección IP, puertos o directorios de configuraciones, se aconseja reiniciar por completo el servicio Squid con *stop / start*.

Si deseamos que el servicio Squid sea iniciado al arranque del sistema, usaremos el comando:

```
| root@manolo-debian:/# update-rc.d squid3 defaults
```

Nota

Cuando se instala el paquete *squid3*, se agrega el servicio *squid3* para que sea iniciado automáticamente al arranque del sistema, por lo que es posible que reciba un mensaje como el siguiente:

```
| root@manolo-debian:/# update-rc.d squid3 defaults
| System startup links for /etc/init.d/squid3 already exist.
```

Nota

En Redhat/CentOS use el comando: **chkconfig squid on** para activar el servicio al inicio del sistema.

Si deseamos que el servidor squid3 NO sea iniciado al arranque del sistema usaremos el comando:

```
| root@manolo-debian:/# update-rc.d -f squid3 remove
```

Nota

En Redhat/CentOS use el comando: **chkconfig squid off** para desactivar el servicio al inicio del sistema.

3 Configuración de los esquemas de control de acceso

Este capítulo está dedicado a la configuración de las listas de control de acceso o ACLs de Squid. Iniciaremos con una introducción al soporte de ACLs, una explicación sobre la lógica y el flujo del procesamiento en los controles de acceso, para continuar con la creación de reglas de acceso basadas en direcciones IP y controles de acceso basados en la URL o dominio.

Squid incorpora un sistema de control de acceso bastante flexible y potente basado en Listas de Control de Acceso o ACLs. Los controles de acceso en Squid están formados por dos componentes principales: **los elementos de ACL y las listas de acceso**.

LOS ELEMENTOS DE ACL

Definen el origen o destino de la petición realizada por el cliente. Hay diferentes tipos de elementos de ACL.

Los tipos de elementos de ACL de tipo **origen** pueden ser:

- ☐ Dirección IP del cliente
- ☐ Nombre del usuario
- ☐ Grupo de usuarios
- ☐ Navegador

Los tipos de elementos de ACL de tipo **destino** pueden ser:

- ☐ Puerto
- ☐ Dominio DNS
- ☐ URL
- ☐ Tipo MIME

Además hay **otros** tipos de elementos de ACL como:

- ☐ Protocolo
- ☐ Método usado en la petición, GET, POST, CONNECT

La directiva de configuración `acl` define los elementos de la ACL:

```
acl nombreacl tipoacl [-i] valor...  
acl nombreacl tipoacl [-i] "archivo"
```

Nota

Cuando use un archivo, escriba un elemento por línea.

LAS LISTAS DE ACCESO

Son usadas para permitir o denegar el acceso a uno o más elementos de ACL.

La directiva **http_aces** define el tipo de acceso a uno o más elementos de ACL:

```
| http_access allow|deny [!]nombreacl ...
```

3.1.1 Tipos de elementos de ACL

En esta sección se listan los diferentes tipos de elementos de ACL soportados por Squid. Se describen los tipos de elementos de ACL para el origen y destino de la petición.

SEGÚN EL ORIGEN

Para crear controles de acceso basados en el origen de la petición del cliente podemos usar los siguientes tipos de elementos de ACL:

src

Dirección IP del cliente, puede ser una sola dirección lista o rango de direcciones IP, soporta el uso de mascararas de subred en formato CIDR

proxy_auth

Autenticación de usuarios vía procesos externos

browser

User-agent (nombre “interno” del navegador web) desde el cual se realiza la petición

SEGÚN EL DESTINO

Para crear controles de acceso basados en el destino de la petición encontramos los siguientes tipos de elementos de ACL:

dstdomain

Este tipo define uno o más dominios destino solicitados por el cliente

url_regex

Tipo con soporte de expresiones regulares para el URL solicitado por el cliente

urlpath_regex

Tipo con soporte de expresiones regulares para el URL solicitado por el cliente, omitiendo el protocolo y el nombre de host (http://hostname). Es decir, sólo se centra en la ruta al recurso.

port

Este tipo define uno o más números de puerto destino solicitados por el cliente

method

Este tipo define el método usado por el cliente para la petición HTTP (get, post, etc)

SEGÚN EL CONTENIDO MIME

Para crear controles de acceso basados en el tipo MIME de la solicitud o respuesta de la petición podemos usar los siguientes tipos de elementos de ACL:

req_mime_type

Tipo con soporte de expresiones regulares para la cabecera *content-type* de la solicitud (request)

rep_mime_type

Tipo con soporte de expresiones regulares para la cabecera *content-type* de la respuesta (reply). Es decir, el contenido descargado.

SEGÚN LA FECHA/HORA

Además es posible crear controles de acceso basados en el tiempo en el que se realiza la petición:

time

hora del día, y día de la semana

3.1.2 Sintaxis de los elementos de ACL

Cada elemento de ACL **debe tener un nombre único asignado**. Los elementos de ACL consisten de uno o más valores, por ejemplo:

```
| acl lista_a tipo valor
```

Si desea que la coincidencia la haga ignorando si son mayúsculas, minúsculas o mixtos use la opción `-i`, por ejemplo:

```
| acl lista_a tipo -i valor
```

Si desea incluir más de un valor, use una lista de elementos, por ejemplo:

```
| acl lista_b tipo valor1 valor2
```

Nota

Cuando use múltiples valores, separe cada valor con un espacio. Si la lista es muy grande, puede continuar en múltiples líneas.

Si requiere crear listas grandes puede continuar la lista en varias líneas, por ejemplo:

```
| acl lista_x tipo valor1 valor2 valor3 valor4 valor5 valor6 valor7 valor8  
valor9 valor10 valor11 valor12
```

Si la lista esta conformada por más de una línea, se recomienda que guarde los elementos del ACL en un **archivo de texto** plano para su fácil revisión y edición, por ejemplo:

```
| acl lista_x src "/etc/squid3/lista_x.acl"
```

Nota

Cuando use una lista basada en archivos, asegúrese de escribir la ruta absoluta al archivo entre comillas.

Y en el archivo cada valor de la lista debe estar en una línea, por ejemplo:

```
root@manolo-debian:/# cat /etc/squid3/ips_sin_internet.acl
valor1
valor2
valor3
valor4
valor5
```

También es posible usar diferentes valores para una misma ACL en diferentes líneas: Squid las combina en una sola lista, por ejemplo:

```
acl lista_z tipo valor1
acl lista_z tipo valor2
acl lista_z tipo valor3
acl lista_z tipo valor4
```

Siga las siguientes recomendaciones para nombrar los elementos de ACL:

- ☐ No puedes asignar el mismo nombre a dos tipos de elementos de ACL. Generará un error de sintaxis
- ☐ No use espacios en el nombre de la ACL
- ☐ Cree nombres menores a 31 caracteres
- ☐ Puede usar el carácter "_" ó "-" para nombres de más de una palabra

3.1.3 Reglas de control de acceso

En esta sección veremos como funcionan los controles de acceso en Squid. Los controles de acceso se realizan principalmente con las directivas:

- **http_access** para peticiones HTTP
- **http_reply_access** para las respuestas HTTP.

Como se explicó brevemente, las reglas `http_access` se usan para permitir o denegar el acceso a uno o más elementos de ACL, es decir, se podría evaluar tanto el **origen** (dirección IP, usuario, ...) como el **destino** (dominio, URL de la petición, ...). Squid tomará toda la información posible de las cabeceras de la petición HTTP.

Dicho esto, el esquema más simple las reglas `http_access` para determinar el acceso a un a un elemento sería el siguiente:

```
| http_access allow|deny acl
```

Squid evalúa las reglas en el orden en el que son escritas, es decir, de arriba hacia abajo. Si la primer regla no hace coincidencia con la petición, entonces el squid realizará una operación de tipo **OR** y evaluará los elementos de la siguiente regla de acceso.

```
| http_access allow|deny acl
      OR
http_access allow|deny acl
      OR
...
```

Si en una regla de acceso hay más de un elemento de ACL, el sistema utiliza el operador **AND** para cada elemento de la regla, esto quiere decir que **todos los elementos de la ACL deben hacer coincidencia para que una acción se aplique.**

En resumidas cuentas, el esquema de evaluación de las reglas de control de acceso en squid se traduce a:

```
| http_access allow|deny acl AND acl AND ...
      OR
http_access allow|deny acl AND acl AND ...
      OR
...
```

Si después de evaluar todas las reglas de acceso, squid no encuentra una regla que haga coincidencia con la petición, la **acción predeterminada será lo contrario a la acción definida por la última regla de la lista, por lo que es recomendable que la última regla en su lista sea aquella que deniegue el acceso sin condiciones.** Puede usar el acl predeterminada `all`, por ejemplo:

```
| http_access allow|deny acl AND acl AND ...
      OR
http_access allow|deny acl AND acl AND ...
      OR
...
http_access deny all
```

Nota

El **acl all** esta comentado por defecto en la mayoría de instalaciones ya que dicha ACL ya está incorporada en squid.

Tome en consideración que habrá ocasiones en donde quiera definir reglas con más de un elemento de ACL, por ejemplo para diferentes puertos. Para tal caso no podrá listar en una misma regla dos puertos diferentes ya que eso no es posible, para tal caso escriba las reglas una delante de la otra para que use una lógica OR.

En las siguientes secciones se muestran diferentes **ejemplos** de elementos de ACL y reglas de acceso para diferentes tipos de ACL.

3.2 Creando reglas de control de acceso basadas en direcciones IP

Para crear una ACL para una dirección IP individual use:

```
# Una sola IP
acl loc_sysadmin src 192.168.221.110/255.255.255.255
```

O use el formato CIDR:

```
# Una sola IP
acl loc_sysadmin src 192.168.221.110/32
```

Para definir una subred clase C use:

```
# ACL para red local
acl loc_subnet src 192.168.221.0/255.255.255.0
```

O una lista de direcciones IP

```
# Lista separada por espacios
acl loc_gerentes src 192.168.221.130 192.168.221.140 192.168.221.150
192.168.221.160
```

La lista puede ser escrita en un archivo de texto plano, cada elemento en una línea:

```
# Lista de direcciones en archivo
acl loc_gerentes src "/etc/squid3/loc_gerentes.acl"
```

También puede crear una lista basada en un rango de direcciones IP, por ejemplo:

```
# Rango de direcciones IP:
acl loc_rango_directores src 192.168.221.200-192.168.221.210/24
```

Ejemplo configuración de ACLs con control direcciones IP:

```
#
# Elementos de ACL:
#
...
#acl all src all
acl localhost src 127.0.0.1/32
acl loc_subnet src 192.168.221.0/24
acl ip_sin_internet src "/etc/squid3/ip_sin_internet.acl"
#
# Reglas de acceso:
#
...
http_access allow localhost
http_access deny ip_sin_internet
http_access allow loc_subnet
http_access deny all
```

Note el orden de las reglas de acceso.

3.3 Creando reglas de acceso basadas en puertos, dominios, URLs y tipos MIME

En esta sección veremos como crear ACLs y reglas de acceso para puertos, dominios DNS, URLs y tipos MIME, se mostrarán ejemplos de configuraciones comunes para cada uno de los tipos antes definidos.

3.3.1 Listas de control de acceso basadas en puertos y método HTTP

El tipo de ACL `port` define una lista de un puerto, un rango o una lista de puertos usados en la petición HTTP usados en la petición, por ejemplo:

```
acl puertos_seguros port 80 21 443 563 70 210 1025-65535
acl puertos_ssl port 443
```

Nota

La lista anterior son las predeterminadas en squid, modifíquela de acuerdo a sus necesidades o políticas de seguridad.

En la instalación predeterminada de squid, se define el ACL **Safe_ports** la cual incluye los puertos 80(HTTP), 21 (FTP), 443 (HTTPS), 563 (NNTP over SSL), 70 (GHOPER), 210 (WAIS) y del 1025 al 65535. Los primeros podrían ser seguros, pero para algunas organizaciones no se permite conexiones a través del proxy a puertos superiores al 1025, con ciertas excepciones, quizá. Si en su organización no se permite el acceso a puertos no privilegiados, elimine 1025-65535 de la lista de elementos del ACL *Safe_ports*.

Las ACL de tipo `method` define una lista e métodos HTTP usados en la petición. En la mayoría de casos los clientes usan el método *GET* ó *POST*. Sin embargo, algunos clientes usan el método *CONNECT* para realizar un túnel hacia un puerto TCP; por ejemplo, para realizar un túnel y conectarse a un sitio seguro vía HTTPS (TCP/443), la conexión sería **tienda.superofertasonline.com:443**

Otros clientes pueden aprovechar esta característica para, por ejemplo, enviar SPAM haciendo un túnel HTTP usando el método *CONNECT* a **smtp.spammermalignoiii.com:25**. Por lo tanto es altamente recomendable limitar el uso del método *CONNECT* a puertos que considere seguros. Por defecto, solo se permite el uso del método *CONNECT* al puerto HTTPS TCP/443..

Ejemplo configuración de ACLs con control de puertos y direcciones IP:

```
#
# Elementos de ACL:
#
...
#acl all src all
acl localhost src 127.0.0.1/32
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
acl loc_subnet src 192.168.221.0/24
acl ip_sin_internet src "/etc/squid3/ip_sin_internet.acl"
#
# Reglas de acceso:
#
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny ip_sin_internet
http_access allow loc_subnet
http_access deny all
```

Algunos sitios publican servicios que usan HTTPS para sitios seguros pero que no siguen los estándares y usan puertos diferentes al 443, por lo que, si se usa la regla anterior para `SSL_Ports`, no podrá entrar a un sitio que es **supercaja.bancototote.com:1850**. Si confía en el sitio, entonces agregue el puerto 1850 al lista de puertos del ACL `Safe_ports` y `SSL_Ports`.

3.3.2 Listas de control de acceso basadas en dominios

El tipo de ACL `dstdomain` define una lista de uno o más nombres de dominio DNS, por ejemplo:

```
| acl videosweb dstdomain youtube.com justin.tv
```

La siguiente regla solo hará coincidencia para las URLs `http://youtube.com` y `http://justin.tv` pero no a `http://www.youtube.com`. Para que la regla también haga coincidencia con los subdominios use el carácter "." como prefijo del elemento del ACL, por ejemplo:

```
| acl dominios_videos dstdomain .youtube.com .justin.tv
```

Si la lista es muy grande, use una lista de dominios basada en archivos, por ejemplo:

```
| acl dominios_porno dstdomain "/etc/squid3/dominios_porno.acl"
```


Ejemplo configuración de ACLs con control de puertos y direcciones IP, y dominios destino:

```
#
# Elementos de ACL:
#
...
#acl all src all
acl localhost src 127.0.0.1/32
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
acl loc_subnet src 192.168.221.0/24
acl loc_sysadmin src 192.168.221.110/32
acl loc_gerentes src 192.168.221.130 192.168.221.140 192.168.221.150
192.168.221.160
acl ip_sin_internet src "/etc/squid3/ip_sin_internet.acl"
acl dominios_videos dstdomain .youtube.com .justin.tv
acl dominios_msn dstdomain .login.live.com .contacts.msn.com
.messenger.msn.com .messenger.hotmail.com .hotmail.msn.com
.messenger.live.com loginnet.passport.com
acl dominios_porno dstdomain "/etc/squid3/dominios_porno.acl"
#
# Reglas de acceso:
#
...
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny ip_sin_internet
http_access allow dominios_msn loc_sysadmin
http_access deny dominios_msn
http_access allow loc_sysadmin
http_access allow loc_gerentes !dominios_porno
http_access allow loc_subnet !dominios_videos !dominios_porno
http_access deny all
```

3.3.3 Listas de control de acceso basadas en URLs

Significado de algunos caracteres en las expresiones regulares:

- ^ Principio de línea
- \$ Final de línea
- \. El punto

El tipo de ACL **url_regex** define expresiones regulares para elementos de ACL basados en el URL o partes del URL.

Para crear una ACL basado en el URL completo use:

```
| acl urls_permitidos url_regex ^http://www.squid-cache.org/Doc/FAQ/$
```

El tipo `url_regex` también puede ser usado para crear controles de acceso basados en palabras dentro del URL, por ejemplo (recuerda: “-i” para que no sea sensible a mayúsculas/minúsculas):

```
acl urls_descargas url_regex -i rapidshare megaupload
acl msn_url url_regex -i gateway.dll
```

El tipo de ACL `urlpath_regex` es basado en la ruta el URL solicitado, es decir, quita la parte del protocolo y el hostname, por ejemplo, si el cliente solicita el URL

http://www.example.com/downloads/music/featured.mp3, la coincidencia tomaría efecto solo a la parte de */downloads/music/featured.mp3*.

Para crear un ACL para extensiones de archivos multimedia use el ACL:

```
acl archivos_multimedia urlpath_regex -i \.mp3$ \.mp4$ \.wma$ \.avi$ \.wmv$
\.mov$ \.mpg$ \.mpeg$ \.ram$ \.vob$
```

O mejor en un archivo como lista:

```
acl archivos_multimedia urlpath_regex -i
"/etc/squid3/archivos_multimedia.acl"
```

El programa MSN Messenger se conecta a URLs que tienen la cadena **gateway.dll**: podemos crear un ACL:

```
acl url_msn url_regex -i gateway.dll
```

Ejemplo configuración de ACLs con bloqueo de dominios, URLs y archivos adjuntos prohibidos y multimedia:

```
#
# Elementos de ACL:
#
...
#acl all src all
acl localhost src 127.0.0.1/32
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
acl loc_subnet src 192.168.221.0/24
acl loc_sysadmin src 192.168.221.110/32
acl loc_gerentes src 192.168.221.130 192.168.221.140 192.168.221.150
192.168.221.160
acl ip_sin_internet src "/etc/squid3/ip_sin_internet.acl"
acl dominios_videos dstdomain .youtube.com .justin.tv
acl dominios_msn dstdomain .login.live.com .contacts.msn.com
.messenger.msn.com .messenger.hotmail.com .hotmail.msn.com
.messenger.live.com loginnet.passport.com
acl dominios_porno dstdomain "/etc/squid3/dominios_porno.acl"
acl urls_descargas url_regex -i rapidshare megaupload
acl archivos_multimedia urlpath_regex -i \.mp3$ \.mp4$ \.wma$ \.avi$ \.wmv$
\.mov$ \.mpg$ \.mpeg$ \.ram$ \.vob$
acl url_msn url_regex -i gateway.dll
```

```
#
# Reglas de acceso:
#
...
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny ip_sin_internet
http_access allow url_msn loc_sysadmin
http_access deny url_msn
http_access allow dominios_msn loc_sysadmin
http_access deny dominios_msn
http_access allow loc_sysadmin
http_access allow loc_gerentes !dominios_porno !urls_descargas
http_access allow loc_subnet !dominios_videos !dominios_porno !
urls_descargas !archivos_multimedia
http_access deny all
```

3.3.4 Listas de control de acceso basadas en tipos MIME

Recuerde que tanto `req_mime_type` y `rep_mime_type` usan expresiones regulares, por lo que la ACL:

```
acl appsh rep_mime_type application/x-sh
```

hará coincidencia para los tipos MIME `application/x-sh` y también `application/x-shockwave-flash`, por lo que se recomienda usar:

```
acl appsh rep_mime_type ^application/x-sh$
```

Controles de acceso basados en la petición de tipos MIME

Para crear ACLs y reglas de acceso basadas en tipos MIME de la petición HTTP use:

```
acl req_mimetype_audio req_mime_type -i ^audio/mp3$ ^audio/mp4$
^audio/mpeg$ ^audio/wav$ ^audio/x-mp3$ ^audio/x-mp4$
```

Cree un archivo de texto plano para crear una lista de tipos MIME multimedia:

```
root@manolo-debian:/# cat /etc/squid3/tipos_mime_audio.acl
^audio/mp3$
^audio/mp4$
^audio/mpeg$
^audio/wav$
^audio/x-mp3$
^audio/x-mp4$
^audio/x-mpeg$
^audio/x-wav$
```

Y cree el ACL:

```
acl req_mimetype_audio req_mime_type -i "/etc/squid3/tipos_mime_audio.acl"
```

Para crear un ACL para tipos MIME de vídeo cree un ACL:

```
acl req_mimetype_video req_mime_type -i ^video/avi$ ^audio/mpeg$  
^video/ogg$ ^video/quicktime$ ^video/x-ms-wmv$
```

O cree un archivo de texto plano para crear una lista de tipos MIME vídeo:

```
root@manolo-debian:/# cat /etc/squid3/tipos_mime_video.acl  
^video/3gpp$  
^video/avi$  
^video/mp4$  
^audio/mpeg$  
^video/ogg$  
^video/quicktime$  
^video/s-ms-asf$  
^video/x-ms-asx$  
^video/x-msvideo$  
^video/x-ms-asf$  
^video/x-ms-wma$  
^video/x-ms-wmv$  
^video/x-ms-wvx$  
^video/x-pn-realvideo$
```

Y cree la ACL:

```
acl req_mimetype_video req_mime_type -i "/etc/squid3/tipos_mime_video.acl"
```

Controles de acceso basados en la respuesta de tipos MIME

Algunas veces no podemos controlar el acceso a los tipos MIME en base a la petición del cliente. (Por ejemplo, esto es común cuando el cliente solicita un URL con un número ID o algún otro identificador y el servidor web responde con una URL que sí incluye un tipo MIME).

Para crear un ACL de tipo MIME para las respuestas use:

```
acl rep_mimetype_flash rep_mime_type -i ^video/flash$ ^video/flv$ ^video/x-  
flv$ ^video/x-shockwave-flash$ ^video/x-swf$
```

Para denegar el acceso a los archivos de tipo MIME flash use:

```
http_reply_access deny rep_mimetype_audio rep_mimetype_video  
rep_mimetype_flash
```

Ejemplo configuración de ACLs con bloqueo de dominios, URLs y archivos adjuntos prohibidos y multimedia y tipos MIME multimedia y flash:

```
#
# Elementos de ACL:
#
...
#acl all src all
acl localhost src 127.0.0.1/32
acl Safe_ports port 80 21 443 563 70 210 1025-65535
acl SSL_ports port 443
acl CONNECT method CONNECT
acl loc_subnet src 192.168.221.0/24
acl loc_sysadmin src 192.168.221.110/32
acl loc_gerentes src 192.168.221.130 192.168.221.140 192.168.221.150
192.168.221.160
acl ip_sin_internet src "/etc/squid3/ip_sin_internet.acl"
acl dominios_videos dstdomain .youtube.com .justin.tv
acl dominios_msn dstdomain .login.live.com .contacts.msn.com
.messenger.msn.com .messenger.hotmail.com .hotmail.msn.com
.messenger.live.com loginnet.passport.com
acl dominios_porno dstdomain "/etc/squid3/dominios_porno.acl"
acl urls_descargas url_regex -i rapidshare megaupload
acl archivos_multimedia urlpath_regex -i \.mp3$ \.mp4$ \.wma$ \.avi$ \.wmv$
\.mov$ \.mpg$ \.mpeg$ \.ram$ \.vob$
acl url_msn url_regex -i gateway.dll
acl req_mimetype_audio req_mime_type -i ^audio/mp3$ ^audio/mp4$
^audio/mpeg$ ^audio/wav$ ^audio/x-mp3$ ^audio/x-mp4$
acl rep_mimetype_audio rep_mime_type -i ^audio/mp3$ ^audio/mp4$
^audio/mpeg$ ^audio/wav$ ^audio/x-mp3$ ^audio/x-mp4$
acl req_mimetype_video req_mime_type -i ^video/avi$ ^audio/mpeg$
^video/ogg$ ^video/quicktime$ ^video/x-ms-wmv$
acl rep_mimetype_video rep_mime_type -i ^video/avi$ ^audio/mpeg$
^video/ogg$ ^video/quicktime$ ^video/x-ms-wmv$
acl req_mimetype_flash req_mime_type -i ^video/flash$ ^video/flv$ ^video/x-
flv$ ^video/x-shockwave-flash$ ^video/x-swf$
acl rep_mimetype_flash rep_mime_type -i ^video/flash$ ^video/flv$ ^video/x-
flv$ ^video/x-shockwave-flash$ ^video/x-swf$
#
# Reglas de acceso:
#
...
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny ip_sin_internet
http_access allow url_msn loc_sysadmin
http_access deny url_msn
http_access allow dominios_msn loc_sysadmin
http_access deny dominios_msn
http_access allow loc_sysadmin
```

```
http_access allow loc_gerentes !dominios_porno !urls_descargas
http_access deny req_mimetype_audio
http_access deny req_mimetype_video
http_access deny req_mimetype_flash
http_access allow loc_subnet !dominios_videos !dominios_porno !
urls_descargas !archivos_multimedia
http_access deny all
http_reply_access allow rep_mime_msn loc_sysadmin
http_reply_access deny rep_mime_msn
http_reply_access deny rep_mimetype_audio
http_reply_access deny rep_mimetype_video
http_reply_access deny rep_mimetype_flash
```

4 Squid con Webmin

4.1 Instalación de Webmin

La versión actual (en el momento de escribir estas líneas) es la 1.6

En este caso, el software no puede encontrarse en los repositorios de Debian/Ubuntu. Por tanto, deberemos descargar el paquete DEB. La web oficial es www.webmin.com



Para instalarlo mediante interfaz gráfica, debería bastar con un doble clic.

Para su instalación mediante comandos, deberemos utilizar el comando `dpkg -i`:

1. Descargar *archivo.deb* y guardarlo en una carpeta del disco duro.

```
root@manolo-debian:/# ls -l /home/manolo/Descargas/
total 24004
-rw-r--r-- 1 manolo manolo 24578284 may  1 00:43 webmin_1.740_all.deb
```

2. Instalarlo con el comando `dpkg`, que sería algo así como una forma de más bajo nivel que `apt` de acceder a la gestión de paquetes.

```
root@manolo-debian:/# dpkg -i /home/manolo/Descargas/webmin_1.740_all.deb
Seleccionando el paquete webmin previamente no seleccionado.
(Leyendo la base de datos ... 146423 ficheros o directorios instalados actualmente.)
Desempaquetando webmin (de .../Descargas/webmin_1.740_all.deb) ...
dpkg: problemas de dependencias impiden la configuración de webmin:
 webmin depende de libauthen-pam-perl; sin embargo:
   El paquete `libauthen-pam-perl' no está instalado.
 webmin depende de libio-pty-perl; sin embargo:
   El paquete `libio-pty-perl' no está instalado.
 webmin depende de apt-show-versions; sin embargo:
   El paquete `apt-show-versions' no está instalado.

dpkg: error al procesar webmin (--install):
 problemas de dependencias - se deja sin configurar
Se encontraron errores al procesar:
 webmin
```

3. Pero *dpkg* sólo instala, sin tener en cuenta dependencias. De hecho, durante la instalación informa de dependencias no resueltas.

Para resolverlas automáticamente, sí que tendremos que echar mano de *apt*.

```
root@manolo-debian:/# apt-get install -f
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Corrigiendo dependencias... Listo
(...)
```

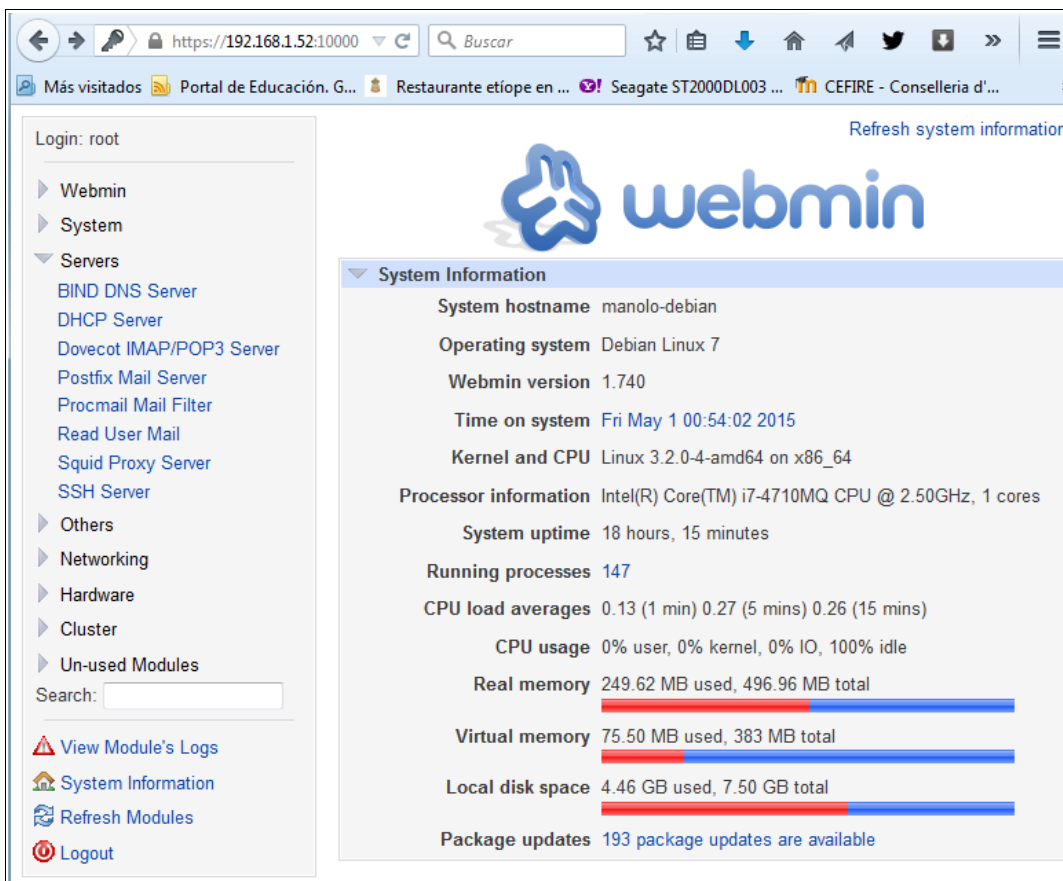
4.2 Acceso a Webmin

Webmin escucha por el puerto 10000 y utiliza conexiones seguras.

Para acceder a Webmin (puede hacerse desde cualquier ordenador de la red) hay que teclear la URL:

https://ip_servidor:10000

Para loguearse y administrar el servidor, el usuario debería ser *root* o tener capacidades de *sudo*.



The screenshot shows the Webmin web interface in a browser. The address bar displays `https://192.168.1.52:10000`. The left sidebar contains a navigation menu with options like 'Webmin', 'System', 'Servers' (with sub-items like BIND DNS Server, DHCP Server, etc.), 'Others', 'Networking', 'Hardware', 'Cluster', and 'Un-used Modules'. The main content area is titled 'System Information' and displays various system metrics:

- System hostname: manolo-debian
- Operating system: Debian Linux 7
- Webmin version: 1.740
- Time on system: Fri May 1 00:54:02 2015
- Kernel and CPU: Linux 3.2.0-4-amd64 on x86_64
- Processor information: Intel(R) Core(TM) i7-4710MQ CPU @ 2.50GHz, 1 cores
- System uptime: 18 hours, 15 minutes
- Running processes: 147
- CPU load averages: 0.13 (1 min) 0.27 (5 mins) 0.26 (15 mins)
- CPU usage: 0% user, 0% kernel, 0% IO, 100% idle
- Real memory: 249.62 MB used, 496.96 MB total
- Virtual memory: 75.50 MB used, 383 MB total
- Local disk space: 4.46 GB used, 7.50 GB total
- Package updates: 193 package updates are available

4.3 Configuración de Squid

Para acceder a la configuración de Squid:

Servers → Squid Proxy Server

Existen varias categorías de configuración: *Ports and Networking*, *Logging*, *Cache Options*, ...

La gestión del filtrado de tráfico se realiza mediante las listas de control de acceso (ACLs) de la categoría **Access Control**, fundamentalmente mediante las pestañas **Access control lists** y **Proxy restrictions**.



La configuración, al igual que la que se realiza manipulando el archivo de configuración, se basa en dos pasos:

1. **Crear listas de acceso** (es una forma de definir condiciones: por ejemplo, las IPs de origen de nuestra red local)
2. **Crear restricciones**, donde debemos escoger, para cada lista:
 - si permitiremos o denegaremos el acceso
 - si buscamos las conexiones que coinciden con la lista o lo contrario, las que no coinciden (por ejemplo: el puerto 80 o todos los puertos excepto el 80)
 - el orden en que se ejecutarán las restricciones

¡Cuidado!

Es muy importante tener en cuenta cómo ordenaremos las restricciones, ya que el proxy tendrá en cuenta únicamente la 1ª coincidencia que encuentre y no analizará las demás.

Las listas se crean en la pestaña **Access control lists** y se agregan, indicando si tienen permitido o denegado el acceso, en la pestaña **Proxy restrictions**. Es aquí, en **Proxy restrictions**, donde el administrador debe poner especial cuidado en ordenar correctamente las restricciones.

4.3.1 Configuración actual

ACLs

La configuración actual incluye las ACL predeterminadas y dos creadas anteriormente por el administrador:

Module Index	Access Control		Apply Changes
Help..			Stop Squid
Access control lists	Proxy restrictions	ICP restrictions	External ACL programs
Name	Type	Matching..	
manager	URL Protocol	cache_object	
localhost	Client Address	127.0.0.1/32 ::1	
to_localhost	Web Server Address	127.0.0.0/8 0.0.0.0/32 ::1	
SSL_ports	URL Port	443	
Safe_ports	URL Port	80	
Safe_ports	URL Port	21	
Safe_ports	URL Port	443	
Safe_ports	URL Port	70	
Safe_ports	URL Port	210	
Safe_ports	URL Port	1025-65535	
Safe_ports	URL Port	280	
Safe_ports	URL Port	488	
Safe_ports	URL Port	591	
Safe_ports	URL Port	777	
CONNECT	Request Method	CONNECT	
loc_subnet	Client Address	192.168.1.0/24	
depor	Web Server Hostname	.sport.es .elmundodeportivo.es .marca.com .as.com	
Create new ACL Browser Regexp			

Edición de la ACL *depor* para observar cómo se crea/modifica:

Module Index	Edit ACL	Apply Changes
		Stop Squid
Web Server Hostname ACL		
ACL Name	depor	
Domains	.as.com .elmundodeportivo.es .marca.com .sport.es	
Failure URL		
Store ACL values in file	<input checked="" type="radio"/> Squid configuration <input type="radio"/> Separate file	
	<input type="text"/> <input type="button" value="..."/>	
<input type="button" value="Save"/>	<input type="button" value="Delete"/>	

RESTRICCIONES

También hay dos creadas por el administrador, utilizando las dos ACLs anteriores:

Module
Index
Help..

Apply
Changes
Stop
Squid

Access control lists **Proxy restrictions** ICP restrictions

Add proxy restriction.

Action	ACLs	Move
<input type="checkbox"/> Allow	manager localhost	↓
<input type="checkbox"/> Deny	manager	↓↑
<input type="checkbox"/> Deny	!Safe_ports	↓↑
<input type="checkbox"/> Deny	CONNECT !SSL_ports	↓↑
<input type="checkbox"/> Allow	localhost	↓↑
<input type="checkbox"/> Deny	depor	↓↑
<input type="checkbox"/> Allow	loc_subnet	↓↑
<input type="checkbox"/> Deny	all	↑

Add proxy restriction.

Delete Selected Restrictions

Edición de la restricción que utiliza la ACL *depor* para observar cómo se crea/modifica:

Module
Index

Edit Proxy Restriction

Proxy Restriction

Action ☐ Allow ☒ Deny

Match ACLs

- all (1)
- manager (2)
- localhost (2)
- to_localhost (0)
- SSL_ports (1)
- Safe_ports (1)
- CONNECT (1)
- loc_subnet (1)
- depor (1)**

Don't match ACLs

- all (1)
- manager (2)
- localhost (2)
- to_localhost (0)
- SSL_ports (1)
- Safe_ports (1)
- CONNECT (1)
- loc_subnet (1)
- depor (1)

Save Delete

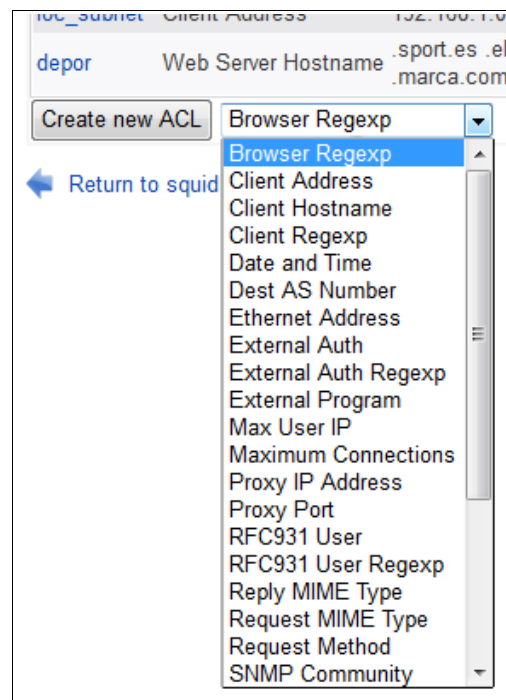
En el fichero de configuración, estas ACLs y restricciones quedan así:

```
(...)
# ACL para red local
acl loc_subnet src 192.168.1.0/24
# ACL para periodicos deportivos
acl depor dstdomain .sport.es .elmundodeportivo.es .marca.com .as.com
(...)
http_access deny depor
http_access allow loc_subnet
(...)
```

4.3.2 Creación de ACLs y restricciones

A modo de ejemplo, vamos a crear una nueva restricción para permitir el acceso a los equipos de la red 10.0.0.0/8

A la hora de crear una ACL, tenemos varios tipos disponibles para filtrar las conexiones: por IP, por puerto, por tipo de contenido, por nombre de dominio, etc.



Los elementos de una lista pueden guardarse en el propio archivo de configuración de Squid o en un archivo de texto plano. Esta segunda opción se recomienda, sobre todo, cuando la lista está formada por más de una línea, para facilitar la revisión y la edición.

From IP	To IP	Netmask
10.0.0.0		255.0.0.0

Creación de la nueva regla:

Reorganizar para insertar la nueva regla en el punto deseado (por defecto, aparece la última):

Al finalizar, pinchar en *Apply changes*.

Access control lists		Proxy restric
Add proxy restriction.		
Action	ACLs	
<input type="checkbox"/> Allow	manager localhost	
<input type="checkbox"/> Deny	manager	
<input type="checkbox"/> Deny	!Safe_ports	
<input type="checkbox"/> Deny	CONNECT !SSL_ports	
<input type="checkbox"/> Allow	localhost	
<input type="checkbox"/> Deny	depor	
<input type="checkbox"/> Allow	loc_subnet	
<input type="checkbox"/> Allow	red_10	
<input type="checkbox"/> Deny	all	

4.3.3 Directrices para la ordenación de restricciones

En general, se suelen seguir las siguientes recomendaciones:

1. Colocar las restricciones predeterminadas al principio y las creadas por nosotros al final.
2. Dentro de nuestras restricciones, colocar las más restrictivas primero.
3. Por defecto no se permite acceso desde los equipos de la red local, por lo que habría que agregar una restricción al respecto.
4. Al final, siempre debería existir una *deny all*, para denegar todo el tráfico que no se ajuste a ninguna restricción.

5 Apéndices

5.1 Chuleta resumen básico

Instalar

```
| apt-get install squid3
```

Archivo de configuración

```
| /etc/squid3/squid.conf
```

Configuración

1. Crear ACL

```
| acl loc_subnet src 192.168.1.0/24
```

2. Agregar restricción basada en dicha ACL

```
| http_access allow loc_subnet
```

Sintaxis ACL

```
| acl nombreacl tipoacl [-i] valor...
```

“-i” para que no sea sensible a mayúsculas/minúsculas

Tipos de ACLs más comunes

- `src` IP de origen (del cliente)
- `dstdomain` Dominio de destino
- `url_regex` Búsqueda de patrones en la URL
- `port` Puerto de destino

Sintaxis restricciones

```
| http_access allow | deny nombre_acl
```

Si existe coincidencia con la ACL, *allow* permite el acceso y *deny* lo deniega.

Otra restricción muy utilizada, asociada a contenidos MIME es *http_reply_access*

```
| http_reply_access allow | deny nombre_acl
```

Ejemplos:

Permitir acceso a los equipos de la red local

```
acl loc_subnet src 192.168.1.0/24  
http_access allow loc_subnet
```

Denegar descargas de archivos MP3

```
acl descargas_audio req_mime_type -i ^audio/mp3$  
http_reply_access deny descargas_audio
```

Aspectos importantes a tener en cuenta

- Las restricciones se comprueban en orden
 - En el momento que se encuentre una coincidencia, se permite o deniega el acceso (según dicha restricción coincidente) y ya no se sigue mirando la lista de restricciones.
- Consejos:
 - Dejar las restricciones predeterminadas las primeras
 - A partir de ahí, colocar las nuevas
 - Es usual que la primera nueva que coloquemos sea la que permita acceso a los equipos de la red local.

5.2 Archivo de configuración por defecto sin comentarios ni líneas en blanco

Archivo original: valores predeterminados:

```
root@manolo-debian:/# grep -E -v '^(#|;|!$|[ ]*#)' /etc/squid3/squid.conf
acl manager proto cache_object
acl localhost src 127.0.0.1/32 ::1
acl to_localhost dst 127.0.0.0/8 0.0.0.0/32 ::1
acl SSL_ports port 443
acl Safe_ports port 80          # http
acl Safe_ports port 21          # ftp
acl Safe_ports port 443         # https
acl Safe_ports port 70          # gopher
acl Safe_ports port 210         # wais
acl Safe_ports port 1025-65535  # unregistered ports
acl Safe_ports port 280         # http-mgmt
acl Safe_ports port 488         # gss-http
acl Safe_ports port 591         # filemaker
acl Safe_ports port 777         # multiling http
acl CONNECT method CONNECT
http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access allow localhost
http_access deny all
http_port 3128
coredump_dir /var/spool/squid3
refresh_pattern ^ftp:          1440 20% 10080
refresh_pattern ^gopher:      1440 0% 1440
refresh_pattern -i (/cgi-bin/|\?) 0 0% 0
refresh_pattern .              0 20% 4320
```

Una buena idea sería realizar una copia de seguridad del archivo original y, posteriormente, eliminar todos los comentarios para obtener un archivo más limpio y manejable.

5.3 Configurar navegador Firefox para usar el proxy

Para configurar Firefox en el cliente para que utilice el proxy:

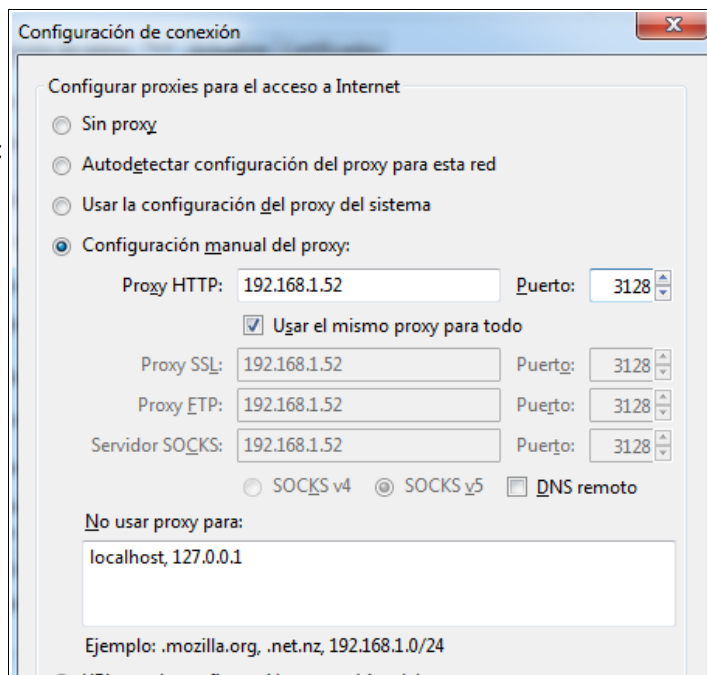
Herramientas → Opciones → Avanzado → Red → Conexión → Configuración
→ Configuración manual del proxy

Ahí debemos indicar la IP del servidor proxy y el puerto (por defecto, el 3128)

Por defecto suele estar configurado para que no se utilice para las conexiones locales: por ejemplo, para acceder a un servidor web instalado en el propio equipo.

No usar proxy para:

localhost, 127.0.0.1



5.4 Ejemplo: permitir accesos a la red local

Al utilizar el proxy con su configuración por defecto, deniega el acceso a los equipos de la red local.



Un vistazo al log de accesos:

```
root@manolo-debian:/# tail /var/log/squid3/access.log
1430389550.218      1 192.168.1.3 TCP_DENIED/403 3686 CONNECT versioncheck-
bg.addons.mozilla.org:443 - NONE/- text/html
(...)
1430389670.138      1 192.168.1.3 TCP_DENIED/403 3638 CONNECT
addons.mozilla.org:443 - NONE/- text/html
1430390109.387      0 192.168.1.3 TCP_DENIED/403 4368 GET
http://www.noticiasdenavarra.com/ - NONE/- text/html
(...)
```

Agregamos nueva ACL *loc_subnet* para permitir conexiones a los equipos de la red local:

```
(...)
acl CONNECT method CONNECT
# ACL para red local
acl loc_subnet src 192.168.1.0/24

http_access allow loc_subnet
http_access allow manager localhost
(...)
```

Y ya podemos acceder.



5.5 Ejemplo: denegar acceso a la prensa deportiva

Ahora que la red local ya tiene acceso (ver ejemplo anterior), podemos empezar restringiendo el acceso a los periódicos deportivos más populares.

Hay que poner cuidado, porque si colocamos la restricción después de la que da acceso a la red local (*loc_subnet*), esta de nueva creación nunca será observada y la restricción que pretendíamos no tendrá efecto.

Agregamos ACL *depor* y colocamos la restricción antes de *loc_subnet*:

```
(...)  
acl CONNECT method CONNECT  
  
# ACL para red local  
acl loc_subnet src 192.168.1.0/24  
  
# ACL para periodicos deportivos  
acl depor dstdomain .sport.es .elmundodeportivo.es .marca.com .as.com  
  
http_access deny depor  
http_access allow loc_subnet  
  
http_access allow manager localhost  
(...)
```

El resultado:



5.6 Expresiones regulares

<http://enavas.blogspot.com.es/2008/03/linux-expresiones-regulares.html>

.	Significa cualquier caracter.
^	Indica el principio de una línea.
\$	Indica el final de una línea.
*	Indica cero o más repeticiones del caracter anterior.
+	Indica una o más repeticiones del caracter anterior.
\<	Indica el comienzo de una palabra.
\>	Indica el final de una palabra.
\	Caracter de escape. Da significado literal a un metacaracter.
[]	Uno cualquiera de los caracteres entre los corchetes. Ej: [A-Z] (desde A hasta Z).
[^]	Cualquier caracter distinto de los que figuran entre corchetes: Ej: [^A-Z].
{ }	Nos permiten indicar el número de repeticiones del patrón anterior que deben darse.
	Nos permite indicar caracteres alternativos: Ej: (^ [?&])
()	Nos permiten agrupar patrones. Ej: ([0-9A-F]+:)+

Ojo: En las expresiones regulares se distingue entre mayúsculas y minúsculas.

EJEMPLOS

```
# grep '^La' fichero
```

El comando anterior nos devuelve todas las líneas del fichero que comienzan por La.

```
# grep '^ *La' fichero
```

El comando anterior nos devuelve todas las líneas del fichero que comienzan por cualquier número de espacios seguido de La.

```
# grep '\.*' fichero
```

El comando anterior nos devuelve todas las líneas del fichero que comienzan por punto y tienen cualquier número de caracteres.

```
# ls -la | grep '\.*'
```

El comando anterior nos devuelve la lista de ficheros que comienzan por un espacio seguido de un punto y cualquier número de caracteres, es decir, la lista de ficheros ocultos.

```
# ls -l | grep '^d'
```

El comando anterior nos devuelve la lista de ficheros que comienzan por d, es decir, la lista de directorios.

6 Fuentes

http://tuxjm.net/docs/Manual_de_Instalacion_de_Servidor_Proxy_Web_con_Ubuntu_Server_y_Squid/html-onechunk/