# Proxy web: SQUID

Ismael Macareno Chouikh

2025-02-03

## Índice

# 1. Introducción

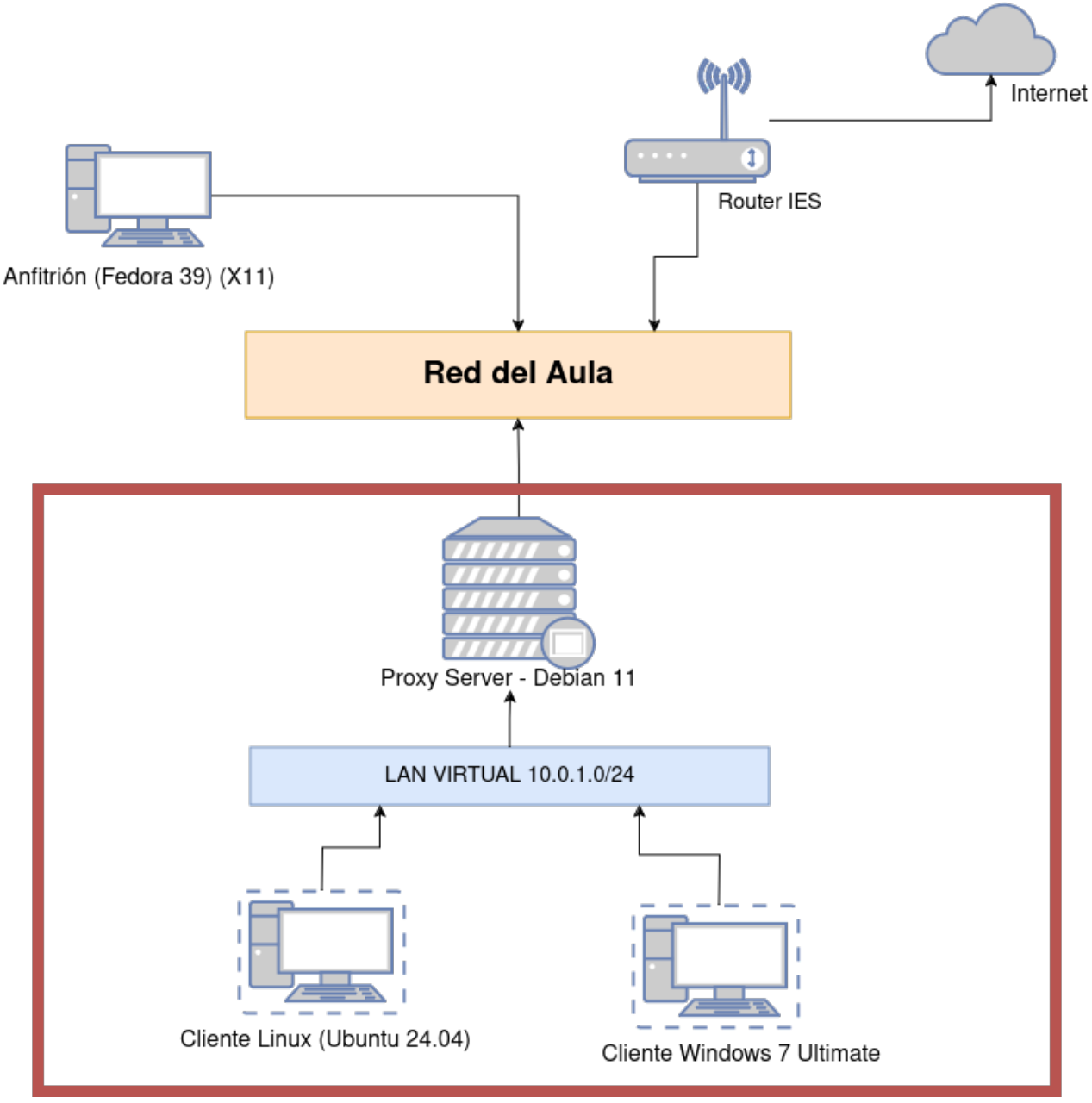
El cortafuegos toma decisiones sobre tráfico en función de los datos de las cabeceras de las capas 3 y 4 (y puede que de la capa 2). Si queremos un control específico sobre protocolos de la capa 5, entonces necesitamos una ampliación que comprenda el protocolo correspondiente, analice "sus cabecerasz decida qué hacer en función de ellas.

En el caso del protocolo de capa 5 (HTTP/HTTPS) (web), a esa aplicación se la llama `proxy`.

Además de servir para tomar decisiones de filtrado también puede realizar caché.

`Squid` es el proxy web por exlencia del mundo del SW libre.

# 2. Entorno



## 3. Instalación y probando `squid`

En este esquema, para facilitar la gestión del servicio proxy, vamos a hacer que el *router* SW que separa la red del aula de nuestra red interna sea una distribución Linux/Debian.

Tiene que estar configurada con dos interfaces de red:

1. Interfaz **modo puente**
   - Realiza NAT
2. Interfaz **red interna**

Una vez configurada la red podremos proceder a la instalación de `squid` mediante los siguientes comandos:

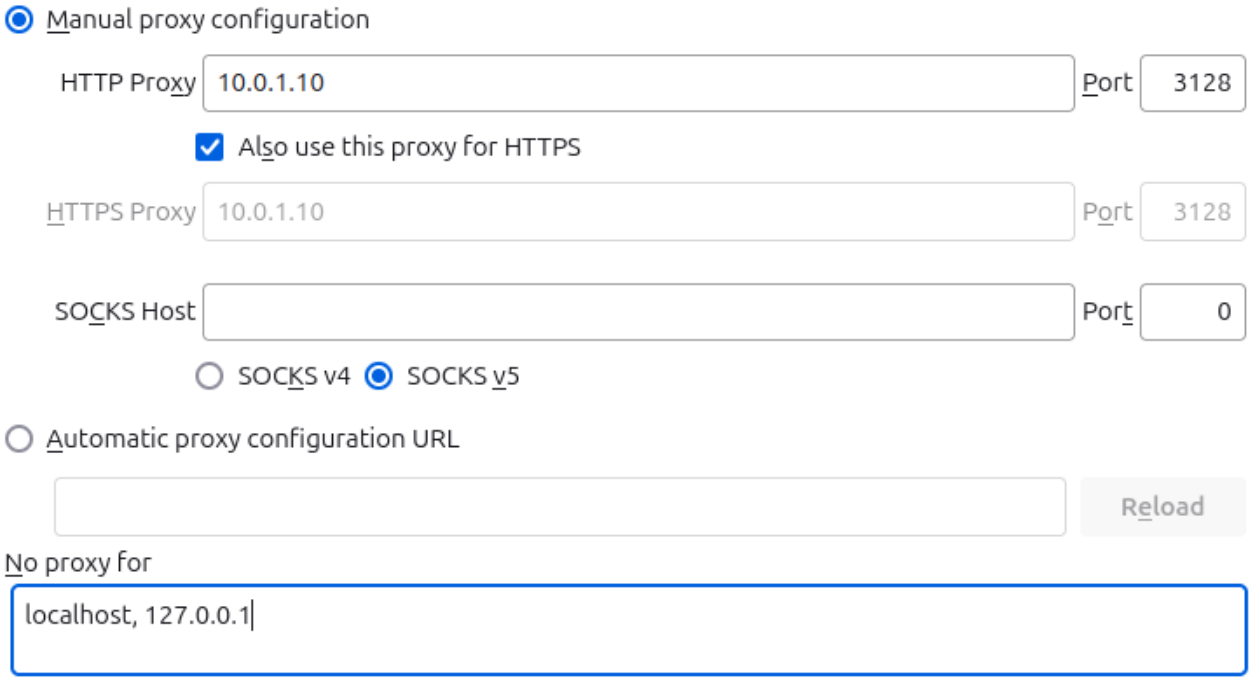
- `sudo apt update`
- `sudo apt install squid`

Habrá que comprobar que `squid` se ha lanzado y que su puerto (3128) esta abierto y escuchando:

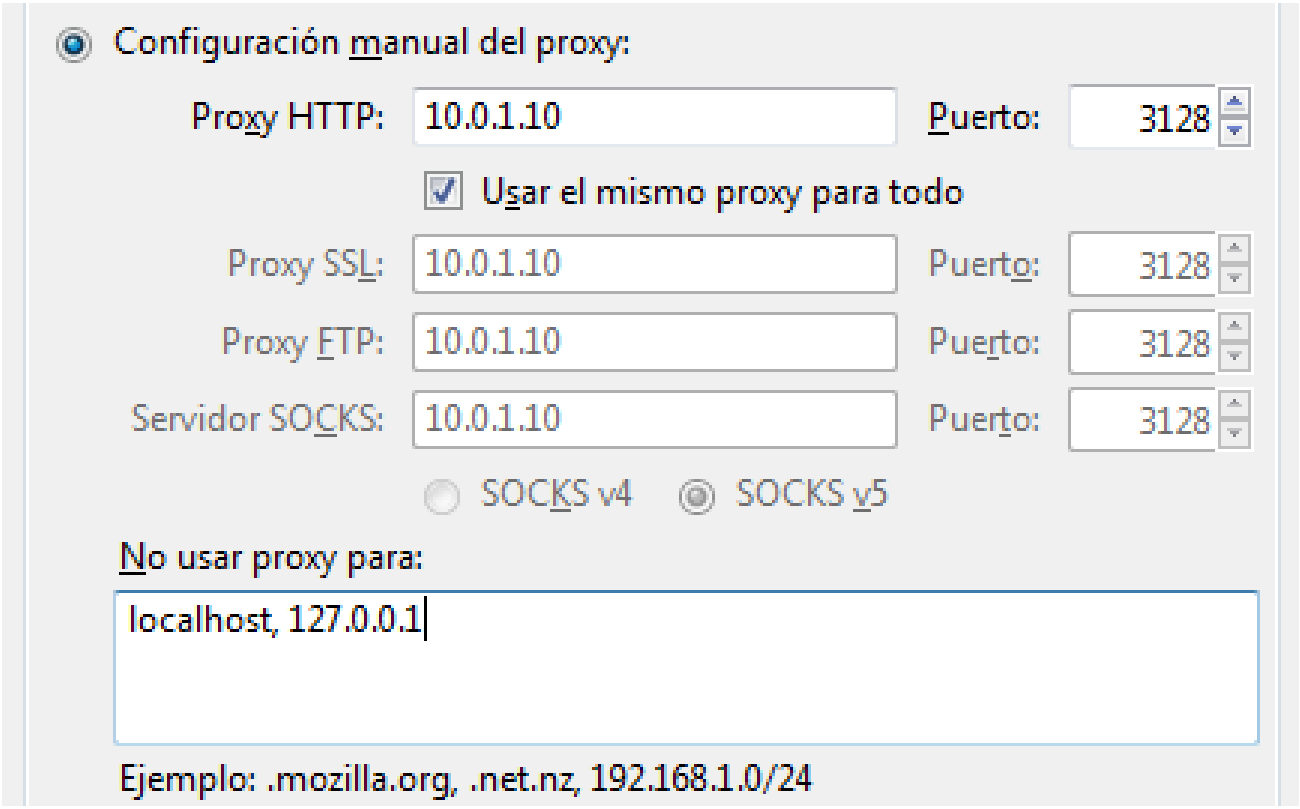
- `ps -ef | grep -i squid`
- `netstat -tulpn | grep 3128`

En el **ubuntu cliente**, habrá que configurar el navegador web firefox para que use al *server* debian 11 como proxy.

1. Menú Editar
2. Preferencias
3. Categoría avanzado
4. Pestaña red
5. Pulsar sobre el botón configuración
6. Sección de conexión:
   - poner la dirección IP del *server* debian 11
   - puerto 3128
   - marcar üsar el mismo proxy para todo"




Habrá que configurar también el cliente *Windows 7 Ultimate*




Una vez configurado el servicio proxy en el *server* debian 11, los cliente ya tendrán acceso a internet pero con restricciones en función de las reglas que se establezcan.

Tal y como está configurado el acceso mediante proxy en los clientes, evidentemente sería muy fácil por parte de los usuarios "quitar.[es]ta configuración de los navegadores web. La cuestión entonces sería cómo se forzaría a los usuarios de la red interna de nuestro laboratorio a usar el proxy.

Como el equipo donde hemos instalado el proxy también es un *router*, podremos hacer que este proxy sea **transparente** para el usuario. Es decir, que el usuario cliente no tenga que configurar absolutamente nada en el navegador web y aún así las peticiones web pasen obligatoriamente por el servidor proxy.

# 4. Configuración de `squid`

El servicio `squid`, al igual que la mayoría de los servicios en Linux, se configura a través de un fichero `.conf` en el directorio `/etc/nombre-servicio`.

En el caso de `squid` es en el fichero `/etc/squid/squid.conf`

Antes de modificar el fichero `squid.conf`, sería recomendable realizar un *backup* por si en un futuro se tuviera que volver a la configuración inicial.

- `sudo cp /etc/squid/squid.conf /etc/squid/squid.conf.orig`

## 4.1. Parámetros generales: almacenamiento caché, nombre y puerto

```
#  TAG: visible_hostname
#        If you want to present a special hostname in error messages, etc,
#        define this.  Otherwise, the return value of gethostname()
#        will be used. If you have multiple caches in a cluster and
#        get errors about IP-forwarding you must set them to have individual
#        names with this setting.
#Default:
# Automatically detect the system host name

#  TAG: unique_hostname
#        If you want to have multiple machines with the same
#        'visible_hostname' you must give each machine a different
#        'unique_hostname' so forwarding loops can be detected.
#Default:
# Copy the value from visible_hostname localhost
```

Éste será el nombre del proxy que mostrará a los clientes proxy cuando `squid` detecte una página o una palabra no permitida.

```
# Squid normally listens to port 3128
http_port 3128
```

```
# Uncomment and adjust the following to add a disk cache directory.
#cache_dir ufs /var/spool/squid 100 16 256
```

```
#Default:
# cache_mem 256 MB
```

```
#Default:
# maximum_object_size_in_memory 512 KB
```

- **cache$_{\text{dir}}$**: determina el tipo de sistema de almacenamiento en caché que usará `squid`
- **cache$_{\text{mem}}$**: determina la cantidad de memoria RAM que será usada como caché

- **maximum$_{\text{objectsizeinmemory}}$**: se indica que los ficheros descargados con un tamaño mayor del indicado no se guardarán en el disco duro.

### 4.2. Archivos de `log`

```
#Default:
# access_log daemon:/var/log/squid/access.log squid
```

```
#Default:
# cache_log /var/log/squid/cache.log
```

El contenido del fichero `/var/log/squid/access.log` son todas las:

- Direcciones IP
- Webs
- Fechas
- etc.

a las que acceden y cuyas peticiones pasan a través de `squid`. En este fichero también aparecen aquellas peticiones que son denegadas debido a que han sido bloqueadas mediante la configuración del proxy.

```
root@debian11:~# tail -f /var/log/squid/access.log
1738321442.826      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321442.833      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321442.835      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321442.851      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321533.510      0 10.0.1.11 TCP_DENIED/403 4023 CONNECT push.services.mozilla.com:443 - HIE
R_NONE/- text/html
1738321929.411      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321929.411      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321929.422      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321929.422      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
1738321929.435      0 10.0.1.11 TCP_DENIED/403 4038 CONNECT incoming.telemetry.mozilla.org:443
- HIER_NONE/- text/html
```

- **Primera columna:** indica la fecha de acceso en formato `epoch`[1]
- **Segunda columma:** tiempo de respuesta
- **Tercera columna:** dirección IP correspondiente a la petición
- **Cuarta columna:** protocolo usado

[1] cantidad de segundos transcurridos desde 00:00:00 UTC 01/01/1970

- **Quinta columna:** tamaño del fichero en bytes
- **Sexta columna:** método de envío del fichero. Puede ser `GET` o `POST`
- **Séptima columna:** URL a la que se quiere acceder
- **Octava columna:** tipo de contenido al cuál estamos haciendo la petición.

### 4.3. Autenticación de usuarios

A través de este servicio podemos solicitar un usuario y una contraseña para poder tener acceso al proxy, y por tanto, a la navegación web.

De esta manera, podemos acotar qué usuarios tendrán acceso al proxy y quiénes no y, de aquellos que tengan acceso, podemos tener un control de qué peticiones realizan.

Los parámetros que permiten configurar el sistema de autenticación son:

```
##auth_param basic program /usr/lib/squid/ncsa_auth /usr/etc/passw
##auth_param basic children 5 startup=5 idle=1
##auth_param basic realm Squid proxy-caching web server
##auth_param basic credentialsttl 2 hours
```

- **Primera línea:** indica el módulo que se va a usar para la autenticación de los usuarios que se encuentra en `/usr/lib/squid/ncsa_auth` así como el fichero que contiene las contraseñas de los distintos usuarios que habrá que generar más adelante
- **Segunda línea:** indica el número de procesos de autenticación que se va a llevar a cabo
- **Tercera línea:** determina el mensaje que aparecerá en la ventana que solicite el usuario y la contraseña
- **Cuarta línea:** el tiempo que tardará el proxy en volver a solicitar de nuevo la clave a cada usuario

Con estos parámetros hemos terminado de configurar la autenticación en el proxy pero ahora hay que añadir los distintos usuarios que tendrán acceso al proxy.

**Aviso Informativo**

Hay que diferenciar los usuarios del S.O con los usuarios del proxy

Para añadir los nuevos usuarios lo haremos a través del comando `htpasswd` de la siguiente manera:

- `htpassdw -c /etc/squid/claves NOMBRE_USUARIO`
- `htpasswd /etc/squid/claves NOMBRE_OTRO_USUARIO`

**Nota**

Para poder usar la utilidad htpasswd será necesario tener instalado previamente el paquete apache2-utils

```
root@debiansad:~# cat /etc/squid/claves
usuario1:$apr1$wEL61v/D$THaY4bS/oLArcpsqY99hZ/
usuario2:$apr1$O1Xw4YEf$sGfODXQCls1FS5FYmmWY7/
```

Creado(s), debemos dejar al menos dos líneas, que indican el tipo de autenticación y una acl requerirla, respectivamente:

```
auth_param basic program /usr/lib/squid/ncsa_auth /etc/squid/claves
acl password proxy_auth REQUIRED
```

### 4.4. Control de Acceso

En `squid` se definen ACL[2], de manera que ante una nueva solicitud HTTP, se mira qué ACLs pertenecen. Una misma solicitud puede pertenecer a varias.

El formato para definir una ACL en `squid` es:

- `acl <nombre><tipo>(cadena|"fichero")+`
  - `nombre:` nombre de la ACL
  - `<tipo>:` que parámetro de la solicitud HTTP se mirará para saber si pertenece o no a la ACL. IP de origen, IP destino, nombre de dominio destino, identificador del navegador o *time*

El fichero de configuración de `squid` (`/etc/squid/squid.conf`) contiene numerosos ejemplos comentados (más o menos sobre la línea 430) a modo de documentación. Nos posicionamos para la configuración sobre la línea **1388**

```
include /etc/squid/conf.d/*

# Example rule allowing access from your local networks.
# Adapt localnet in the ACL section to list your (internal) IP networks
# from where browsing should be allowed
#http_access allow localnet
http_access allow localhost

# And finally deny all other access to this proxy
http_access deny all
```

Se puede apreciar que se permite el acceso HTTP a `localhost` y se deniega a todo lo demás. Debido a esto los clientes de la red interna no tienen aún acceso a internet.

Vamos a configurar el proxy para que los clientes de la red interna tengan acceso a internet. Podemos empezar a definir nuestras ACLs a partir de la línea **1198**.

```
acl Safe_ports port 777                # multiling http
acl CONNECT method CONNECT

acl RedInterna src 10.0.1.0/24

#  TAG: proxy_protocol_access
#        Determine which client proxies can be trusted to provide correct
#        information regarding real client IP address using PROXY protocol.
```

Las ACL sirven para categorizar las solicitudes, pero no toman decisiones sobre ellas. Para eso están los operadores sobre ACL:

- `<tipo operador>allow|deny ([!]<listaACL>)+`
  - `<tipo operador>:` aspecto a decidir sobre la consulta. (Ej. realizarla o no)

[2] *Access control list*

- `allow|deny`: permitir o denegar
- **Lista de nombres de ACL separadas por espacio**

El operador más importante es `http_access` que sirve para decidir si cursar o no la solicitud.

De nuevo, en el fichero de configuración se proporcionan varios ejemplos. La que más nos interesa es la última, `http_access deny all` que deniega el acceso a la ACL a la que, como vimos, pertenecían las solicitudes realizadas desde la IP origen 0.0.0.0/0, es decir, todas.

**¡OJO!**

El orden de colocación de las $\text{http}_{a}ccesses importante. Portanto, las reglas que pongamos deberan ir antes que esta$

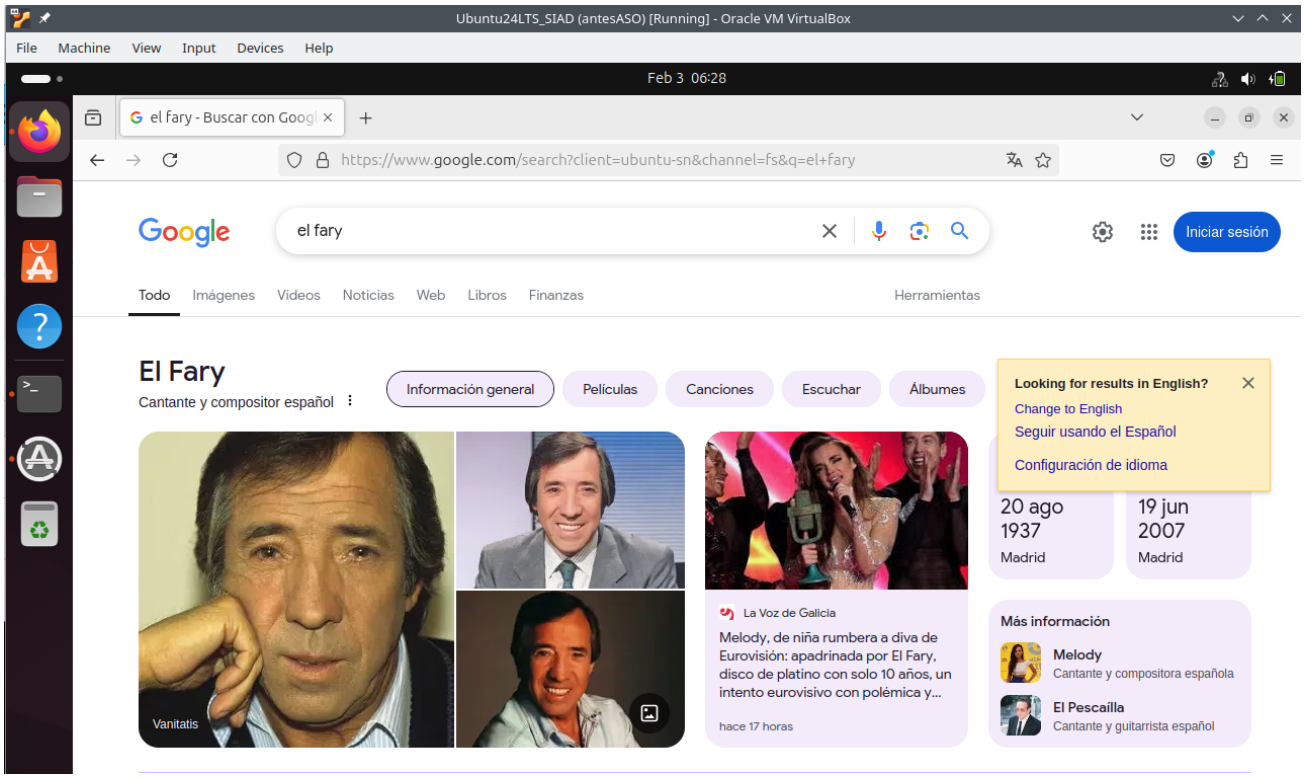
Hasta ahora, en el ejemplo de dar acceso a internet a la red interna, solamente hemos definido la ACL. Seguidamente hay que configurar la regla. Asi que justo antes de la regla `http_access deny all`, añadimos:

```
#http_access allow localnet
http_access allow localhost

http_access allow RedInterna

# And finally deny all other access to this proxy
http_access deny all
```

Salimos del fichero guardando los cambios y reiniciamos el servicio `squid` con el siguiente comando:

- `sudo systemctl restart squid.service`
- `sudo service squid restart`

A partir de ahora los equipos de la red interna sí que podrán navegar por internet.



Vamos a continuar con el ejemplo definiendo tres tipos de filtros diferentes:

```
# Mis listas de control de acceso
acl RedInterna src 10.0.1.0/24
acl acceso src 0.0.0.0/0
acl pal_nopermitidas url_regex "/etc/squid/pal_nopermitidas.txt"
acl dom_nopermitidas dstdomain "/etc/squid/dom_nopermitidos.txt"
```

- La **primera ACL** permite escuchar todas las peticiones procedentes de todas las direcciones IP que se comuniquen con el proxy. Con este parámetro podemos acotar qué parte de una red queremos que pase por el proxy.
- La **segunda ACL** (`pal_nopermitidos`) es la que define el filtrado de contenido (`url_regex`). Esta lista nos permite añadir el fichero `/etc/squid/pal_nopermitidas` todas las palabras que queremos que sean filtradas con el fin de bloquear el acceso a las *webs* que contengan este conjunto de palabras.
- La **tercera ACL** contiene las *webs* a las que el proxy no permitirá el acceso (dominios destino: `dstdomain`). De igual manera que la lista que filtra el contenido, esta lista también permite añadir el fichero `/etc/squid/dom_nopermit` con todas las URL de las páginas web que queremos bloquear.

Las ACL por sí mismas no hacen nada, ya que solo definen qué lista realizará qué tipo de filtrado. Por ello, es necesario permitir o denegar las diferentes ACLs. Sobre la línea 1400 del fichero:

```
#http_access allow localnet
http_access allow localhost

http_access deny acl pal_nopermitidas
http_access deny dom_nopermitidos
http_access allow RedInterna

# And finally deny all other access to this proxy
http_access deny all
```

A continuación, se deben crear los ficheros `/etc/squid/pal_nopermitidas` y `/etc/squid/dom_nopermitidos`

Fichero `/etc/squid/pal_nopermitidas.txt`

porn

xxx

chatgpt

Fichero `/etc/squid/dom_nopermitidos.txt`

marca.com

`.chatgpt.com`

## 5. Práctica resuelta de configuración de `squid`

### 5.1. Definiendo ACLs

Crea las siguientes ACLs, definiéndolas en la zona del fichero donde están las ACL de ejemplo:

- `redInterna`: aquellas solicitudes que vengan de la red interna
- `redCasa`: aquellas solicitudes que vengan de la red del casa
- `ipPrivilegiada`: aquellas solicitudes que vengan de la MV Ubuntu 24.04 LTS cliente
- `ipNoPrivilegiada`: aquellas solicitudes que vengan de la MV *Windows 7 Ultimate*

- `noPermitidas`: consultas que contengan palabras como las definidas en el fichero `/etc/squid/pal_nopermitidas`
- `redSocial`: consultas que contengan los subdominios youtube.com, facebook.com
- `horaRecreo`: consultas realizadas en el período 11:15-11:40 en días laborales

```
# Mis listas de control de acceso
acl RedInterna src 10.0.1.0/24
acl redCasa src 192.168.1.0/24
acl ipPrivilegiada src 10.0.1.11/24
acl ipNoPrivilegiada src 10.0.1.12/24
acl noPermitidas url_regex -i "/etc/squid/pal_nopermitidas.txt"
acl redSocial dstdomain .youtube.com .facebook.com
acl horaRecreo time MTWHF 11:15-11:40
```

## 5.2. Definiendo permisos

Realiza las siguientes configuraciones.

- Permitir el uso del proxy únicamente desde la red interna
  - `http_access allow redInterna`
- Permitir la salida a internet desde el Ubuntu cliente pero no desde el cliente *Windows 7*
  - `http_access allow ipPrivilegiada`
  - `http_access deny ipNoPrivilegiada`
- Permitir la salida a la red interna salvo a las redes sociales. La IP de Ubuntu cliente
  - `http_access allow ipPrivilegiada`
  - `http_acces allow redInterna !redSocial`
- Impedir el acceso a redes sociales a la red interna excepto durante el recreo, pero permitirlo a la IP del ubuntu cliente
  - `http_access allow ipPrivilegiada`
  - `http_acces allow redInterna redSocial !horaRecreo`
- Permitir salir a internet a toda la red interna únicamente solicitando identificación de usuarios
  - `http_acces allow redInterna password`

```
http_access allow RedInterna
http_access allow ipPrivilegiada
http_access deny ipNoPrivilegiada
http_access allow ipPrivilegiada
http_acces allow redInterna !redSocial
http_access allow ipPrivilegiada
http_acces allow redInterna redSocial !horaRecreo
http_acces allow redInterna password
```

### 5.3. Otras reglas

- Descargas de ficheros `.iso` y `.mp3` (`urlpath_regex`)
  - `acl extensionIso urlpath_regex .iso$ .mp3$`
- Acceso a enlaces
  - `acl clicEnGoogle referer_regex https://wwww.google.*`

```
# Mis listas de control de acceso
acl RedInterna src 10.0.1.0/24
acl redCasa src 192.168.1.0/24
acl ipPrivilegiada src 10.0.1.11/24
acl ipNoPrivilegiada src 10.0.1.12/24
acl noPermitidas url_regex -i "/etc/squid/pal_nopermitidas.txt"
acl redSocial dstdomain .youtube.com .facebook.com
acl horaRecreo time MTWHF 11:15-11:40
acl extensionIso urlpath_regex .iso$ .mp3$
acl clicEnGoogle referer_regex https://wwww.google.*
acl acceso src 0.0.0.0/0
acl pal_nopermitidas url_regex "/etc/squid/pal_nopermitidas.txt"
acl dom_nopermitidas dstdomain "/etc/squid/dom_nopermitidos.txt"
```

### 5.4. Proxy transparente

Tener que configurar el proxy en todos los navegadores de los usuarios puede ser tedioso. Para evitarlo, podemos usar un **proxy transparente**.

Usando `iptables` se puede redirigir las solicitudes cambiando la IP y puertos destino. Podemos así redirigir cualquier cosa que intente salir por el *router* llevando el puerto destino 80 al puerto 3128 en local.

- Desconfigura el proxy en el Firefox del cliente Ubuntu 24.04 LTS para que salga directamente a la web
- Usando `iptables` en el *server*, consigue que cualquier paquete que intente salir al puerto 80 sea redirigido al 3128, donde espera `squid`
  - `iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j REDIRECT --to-port 3128`
- Comprueba que puedes navegar desde el cliente Ubuntu. Intenta conectarte a una de las páginas denegadas

**AVISO**

Si has copiado directamente mis bloques de código he tenido varios fallos de sintáxis y debido a esto no se va a reiniciar bien el servicio. Copia lo siguiente

```
#http_access allow localnet
http_access allow localhost

http_access deny pal_nopermitidas
http_access deny dom_nopermitidos
http_access allow RedInterna
http_access allow ipPrivilegiada
http_access deny ipNoPrivilegiada
http_access allow ipPrivilegiada
```

```
http_access allow redInterna !redSocial
http_access allow ipPrivilegiada
http_access allow redInterna redSocial !horaRecreo
http_access allow redInterna password

# And finally deny all other access to this proxy
http_access deny all
```

```
# Mis listas de control de acceso
acl RedInterna src 10.0.1.0/24
acl redCasa src 192.168.1.0/24
acl ipPrivilegiada src 10.0.1.11
acl ipNoPrivilegiada src 10.0.1.12
acl redSocial dstdomain .youtube.com .facebook.com
acl horaRecreo time MTWHF 11:15-11:40
acl extensionIso urlpath_regex .iso$ .mp3$
acl clicEnGoogle referer_regex https://wwww.google.*
acl acceso src all
acl pal_nopermitidas url_regex "/etc/squid/pal_nopermitidas.txt"
acl dom_nopermitidos dstdomain "/etc/squid/dom_nopermitidos.txt"
```