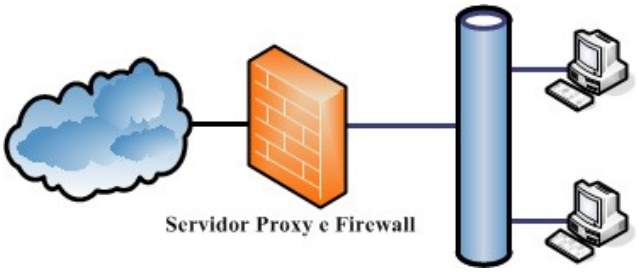
1. Cortafuegos de nivel de aplicación: Proxys

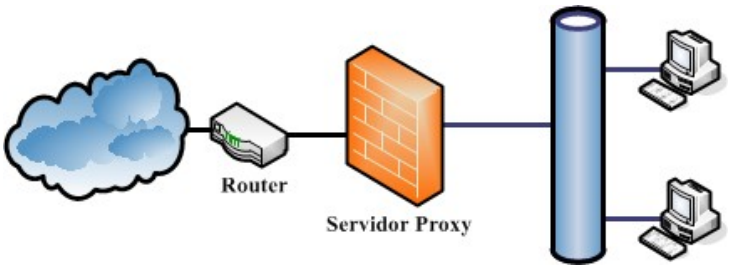
Los Proxys o firewalls de nivel de aplicación se consideran los cortafuegos más seguros e inteligentes al aislar completamente a los clientes y trabajar en la capa superior de la pila de protocolos. Sin embargo, estas características tienen un coste que se traduce en una mayor necesidad de recursos (procesador y memoria). Entre los usos típicos en las organizaciones se encuentran:

- Restringir la entrada a usuarios a zonas controladas.
- Restringir los permisos y accesos de los usuarios en la navegación por Internet.
- Prevenir los ataques a nivel de protocolo y/o aplicación desde el exterior.
- Prevenir la divulgación de información privada o sensible de una organización y/o sus usuarios.

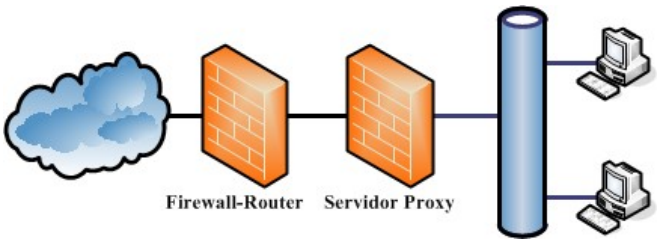
Como ya se indicó, un proxy es un sistema intermedio entre los dispositivos internos de una red y los hosts de una red pública, de tal forma que recibe las peticiones y las envía, después de la verificación de accesos y privilegios. Los proxys son efectivos sólo si se utilizan junto a otros métodos de filtrado y restricción de tráfico IP, por lo que generalmente se suele incorporarlos en los produtos de firewalls de nivel de red o se colocan en la red para trabajar conjuntamente y así complementarse. En las siguientes imágenes se pueden ver las arquitecturas típicas:



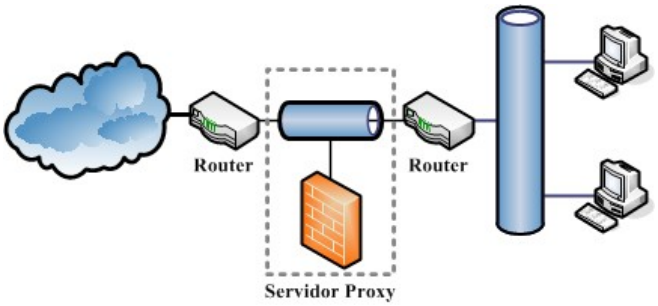
Arquitectura dual-homed host



Arquitectura screened-host



Arquitectura screened-host



Arquitectura screened subnetwork

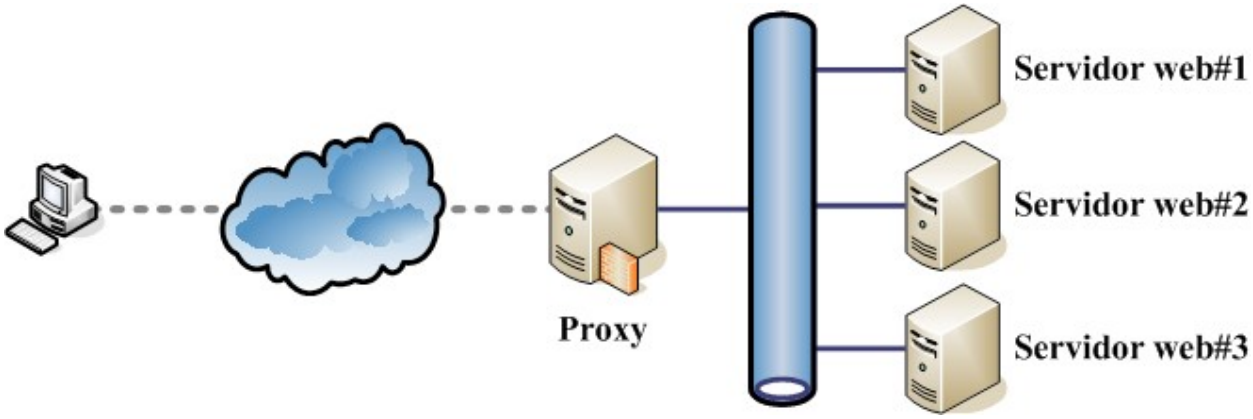
Hay dos tipos fundamentales de proxys:

- **Servidor proxy de aplicación (Application Level Proxy)**: es un sistema que es capaz de entender e interpretar el protocolo de nivel aplicación para el que está dando servicio.
- **Servidor proxy inverso (Reverse Proxy)**: es un sistema que se coloca delante de un conjunto de servidores de forma que todo el tráfico entrante desde Internet con destino a esos servidores pase por él. El proxy inverso recibirá todas las peticiones, se encargará de analizarlas y de ser el caso reenviarlas a los servidores web.

2. Servidor proxy de aplicación

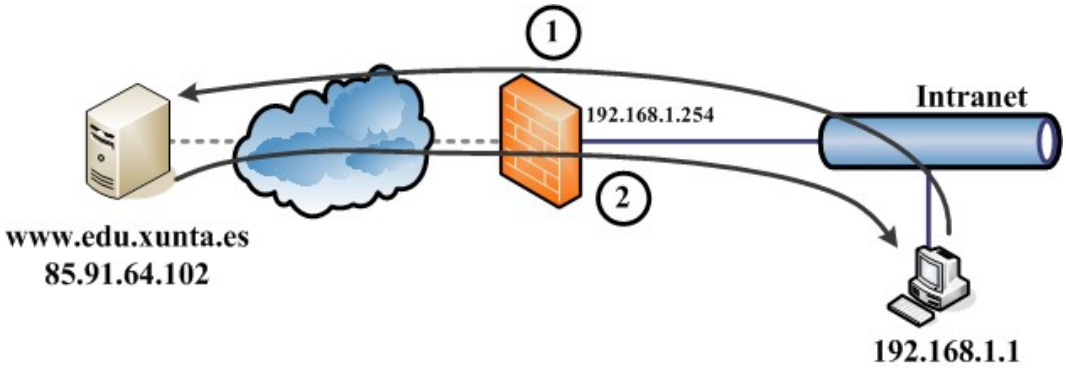
Un proxy de aplicación es un sistema intermedio entre los dispositivos internos de una red y los hosts de una red pública, de forma tal que recibe las peticiones de los clientes, efectuadas mediante un protocolo del nivel de aplicación, verifica los privilegios y las envía al servidor

correspondiente como si fuesen propias. Cuando el servidor responde al proxy, éste le envía la respuesta al cliente.



Proxy Inverso

Para ver la diferencia fundamental entre un network level firewall y un proxy de aplicación, se analizará que es lo que sucede cuando una persona situada en el equipo 192.168.1.1 abre un navegador y escribe http://www.edu.xunta.es para ver la página principal del servidor web de la Consellería de Educación.



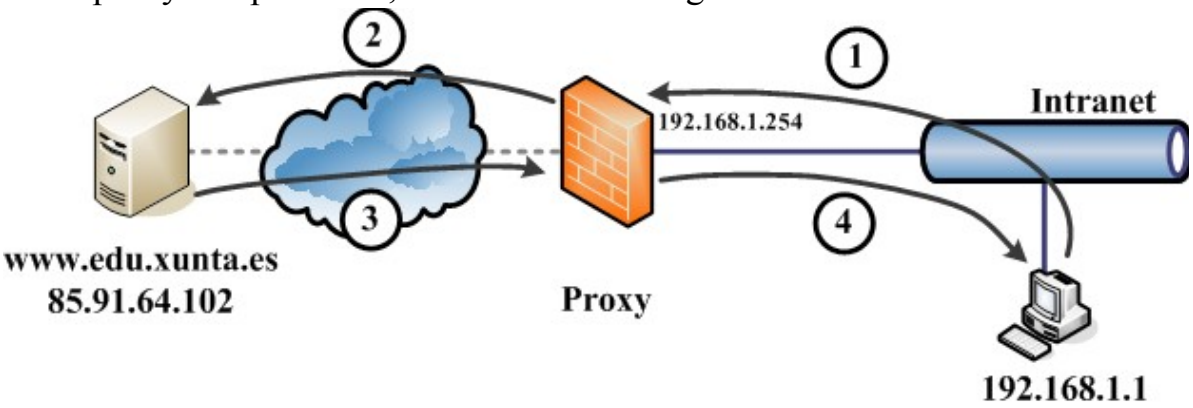
En el caso del network level firewall se estararía ante la siguiente situación:

1. Una vez hecha la resolución dns, el cliente lanza una petición al servidor web www.edu.xunta.es para ver la página de inicio. Esa petición http viajará en un datagrama IP con:
   - IP origen: la del cliente (192.168.1.1).
   - IP destino: la del servidor web (85.91.64.102).
2. Para salir a Internet el cliente mirará su tabla de enrutamiento y el paquete se enrutará hacia el destino llegando al firewall. Una vez que el datagrama IP llega al firewall, se verificarán los privilegios, se hará nat y se reenviará el paquete hacia el servidor web.
3. El servidor 85.91.64.102 recibe el paquete con la solicitud y responde al cliente.

La comunicación es entre el cliente y el servidor; de hecho, si se ejecuta netstat en el cliente para ver las conexiones establecidas, aparecerá una con el servidor 85.91.64.102 y ninguna con el firewall-router:

```
$ sudo netstat -putan
Conexiones activas de Internet (servidores y establecidos)
Proto Recib Enviad Dirección local    Dirección remota   Estado       PID/Program name
tcp   1    1       192.168.1.1:41208  85.91.64.102:80    ESCUCHAR     5718/firefox
```

En el caso del proxy de aplicación, se estaría ante la siguiente situación:



1. El cliente establece una conexión con el Proxy para solicitarle el recurso http://www.edu.xunta.es. La IP destino de la comunicación es la IP del Proxy.
2. El proxy verifica los privilegios y si todo es correcto, hace la resolución dns y procede a crear una solicitud con destino al servidor web www.edu.xunta.es para ver su página de inicio. La IP origen de esta nueva comunicación es la del Proxy.
3. El servidor 85.91.64.102 recibe la solicitud del proxy y responde.
4. El proxy, una vez recibida y analizada la respuesta del servidor web, procede a enviarle la respuesta al cliente.

```
 6 192.168.1.1      192.168.1.254   HTTP    GET http://www.edu.xunta.es/ HTTP/1.1
11 192.168.0.254    85.91.64.102    HTTP    GET / HTTP/1.0
15 85.91.64.102     192.168.0.254   HTTP    [TCP Retransmission] HTTP/1.1 302 Found  (text/html)
18 192.168.1.254    192.168.1.1     HTTP    HTTP/1.0 302 Moved Temporarily  (text/html)
```

Es muy importante darse cuenta que la comunicación real se produce entre cliente-proxy y entre proxy-servidor. Es decir, **hay dos conexiones tcp independientes, una para hablar por http entre el cliente y el proxy, y otra distinta para que hablen http el proxy y el servidor web**. Aunque la comunicación real se produce entre cliente-proxy y entre proxy-servidor, un usuario situado en el cliente tiene la 'ilusión' de comunicarse directamente con el servidor. De hecho:

- De ejecutarse netstat en el cliente, se vería una conexión establecida con el proxy (y ninguna con el servidor).
- De ejecutarse netstat en el servidor web, se vería una conexión establecida con el proxy (y ninguna con el cliente 192.168.1.1).

Ejecutando netstat en la máquina proxy se pueden ver las dos conexiones:

```
$ sudo netstat -putan
Conexiones activas de Internet (servidores y establecidos)
Proto Recib Enviad Dirección local    Dirección remota     Estado      PID/Program name
tcp   0      0     192.168.1.254:3128   0.0.0.0:*          LISTEN      3933/squid3
tcp   0      0     192.168.1.254:3128   192.168.1.7:37149  ESTABLISHED 3933/squid3
tcp   0      0     192.168.0.254:55198  85.91.64.102:80    ESTABLISHED 3933/squid3
```

Usando este mismo ejemplo se puede profundizar en las diferencias entre los network level firewalls y los proxys. El firewall de nivel de red del ejemplo permitía las comunicaciones con destino al puerto 80 tcp, lo cual en principio permitiría el tráfico web (http). Analizando esa regla y conociendo el funcionamiento de los firewalls de nivel de red, se tiene que llegar a la conclusión de que no solamente se permite el tráfico web; sino que también se permite todo el tráfico tcp con destino el puerto 80, aunque a nivel de aplicación no se use http (p.e., se permitiría el tráfico a un servidor ftp trabajando en el puerto 80). Con un proxy, se podrá

verificar que el protocolo a nivel de aplicación es http y por tanto se deja pasar; y en caso de ser otro protocolo, se rechazaría la comunicación.

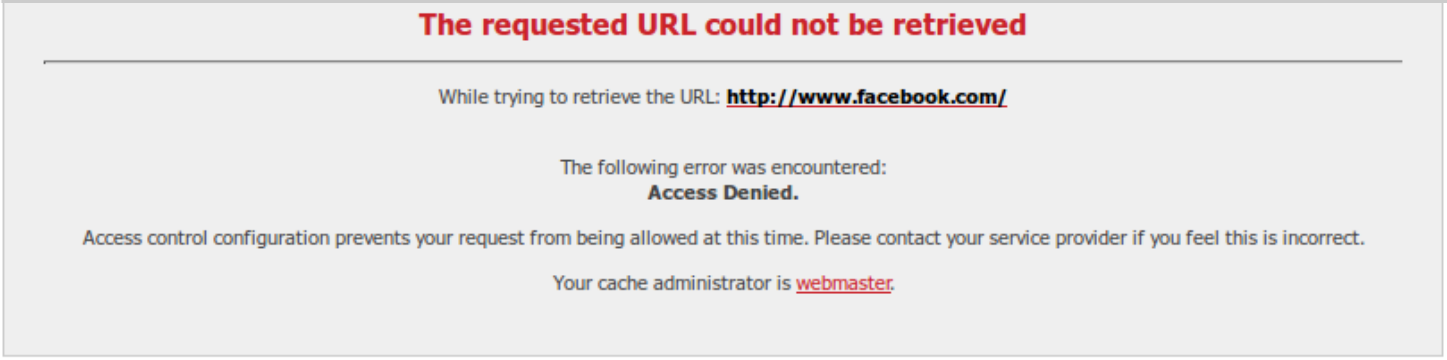
El proxy de aplicación tiene que hablar el protocolo de nivel de aplicación que usa el cliente. De esta forma, independientemente del puerto usado, sólo se podrá acceder a los servicios para los que se proporciona un proxy. Aunque el servidor proxy más usado es el proxy para el protocolo http (web); puede haber servidores proxy para cada protocolo de aplicación en particular (ftp, dns, ldap, smtp, imap, pop, etc.).

Funcionalidades

Al trabajar a nivel de aplicación se abren nuevas posibilidades y funcionalidades. Algunas funcionalidades de los proxys son:

- Permitir (o no) las comunicaciones en función de:
  - Su origen y/o destino, a nivel de aplicación. Por ejemplo, denegar el acceso a sitios web, a URLs concretas, a direcciones de correo electrónico.

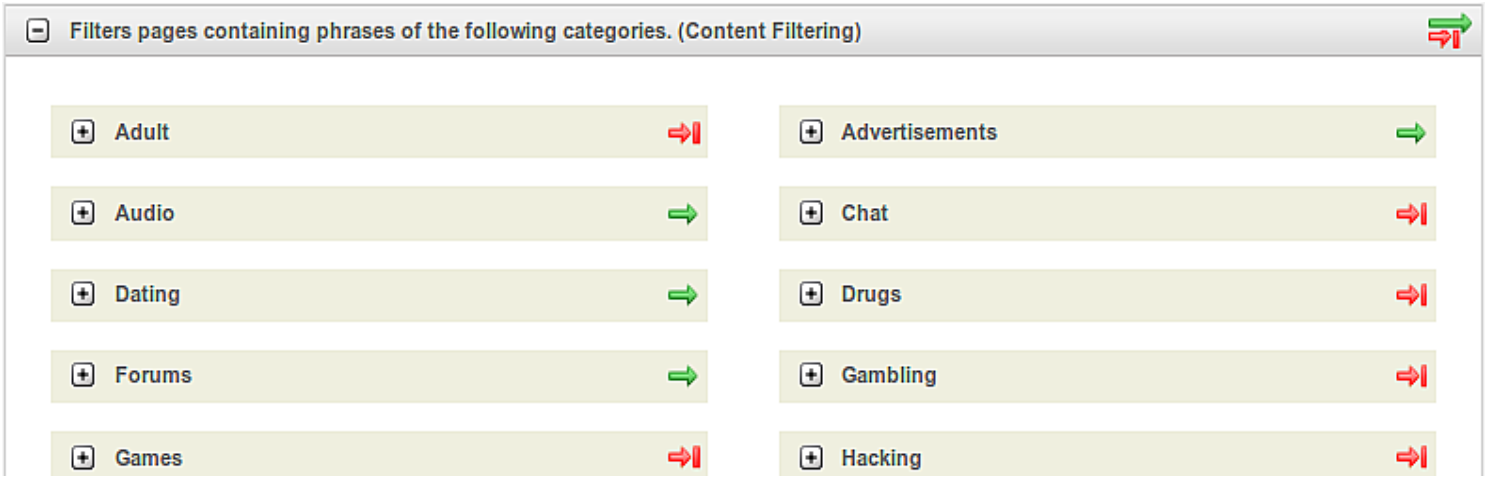
#	Policy	Source	Destination	Authgroup/-user	When	Useragent
1	access denied	GREEN	.facebook.com .facebook.es	not required	Always	ANY
2	unfiltered access	GREEN	ANY	not required	Always	ANY



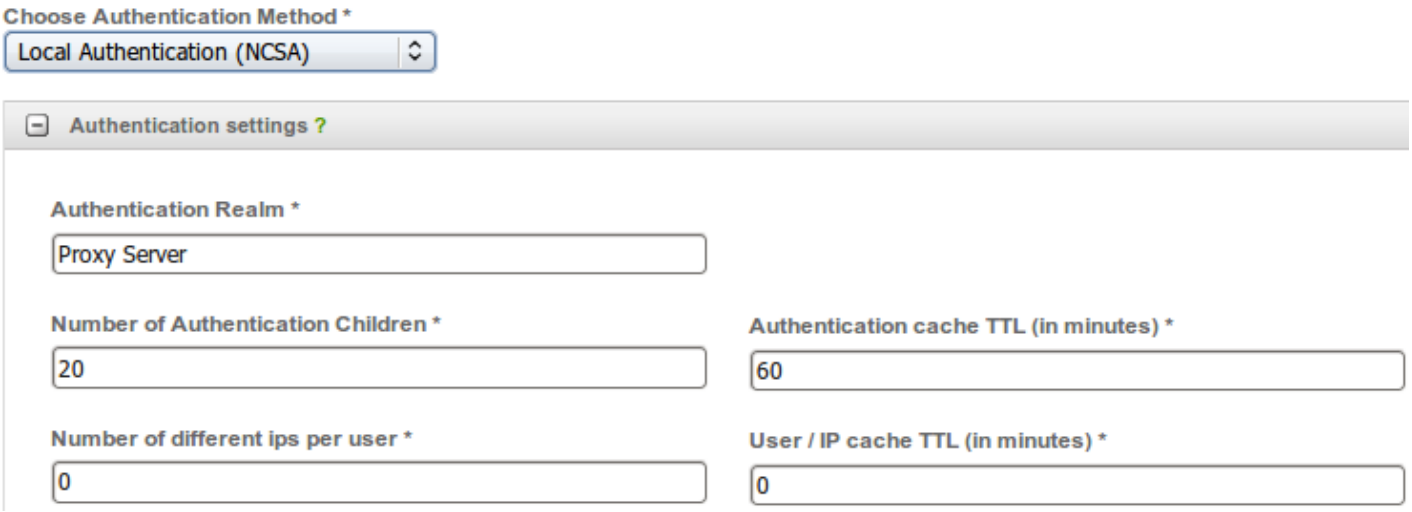
  - El protocolo de nivel de aplicación usado.
- La aplicación utilizada para comunicarse con el exterior (p.e. permitir firefox e I.E. y bloquear el resto de navegadores).

#	Policy	Source	Destination	Authgroup/-user	When	Useragent
1	access denied	GREEN	.facebook.com .facebook.es	not required	Always	ANY
2	filter for virus	GREEN	ANY	not required	MTWHFAS 08:45-14:25	FIREFOX MSIE WINUPD APT

- El contenido de los datos de aplicación. Por ejemplo, impedir el acceso a páginas deportivas o pornográficas, descartar mensajes de correo electrónico con adjuntos de tipo .exe o .pif.



- Autenticación de los usuarios.
  - Contra bases de datos locales del proxy.
  - Contra directorios: hay proxys compatibles con diferentes tecnologías de directorio (Active Directory, LDAP, Radius, …) lo que permite su integración con los servicios de directorio de usuarios de las organizaciones. Esto permite gestionar de forma centralizada los usuarios y sus privilegios, lo que suele representar una ventaja desde el punto de vista de la seguridad y redunda en una mayor comodidad de cara al usuario.

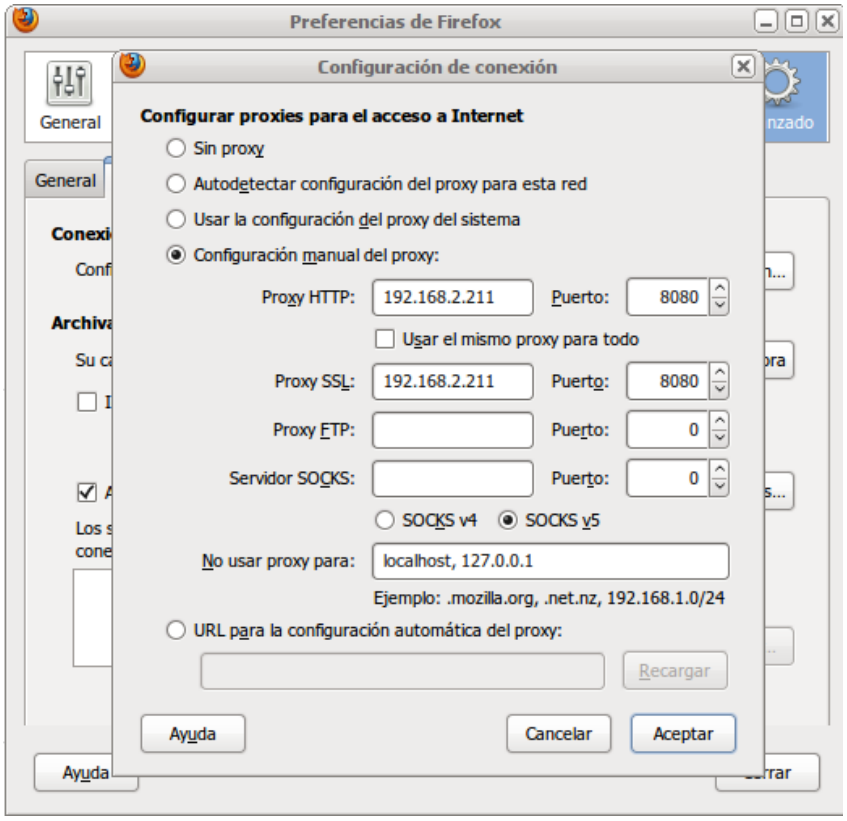


- Servicio de logs a nivel de aplicación, lo que proporciona información para fines de auditoría. Un network level firewall está limitado a la información de las capas de red y transporte (básicamente direcciones IP y puertos); sin embargo, ahora es posible introducirse en un nivel más cercano a los usuarios. Por ejemplo, no se está limitado a saber la IP del servidor web al que se conecta un cliente, se puede saber el sitio web concreto dentro de ese servidor al que accede el cliente, que recurso concreto dentro de ese sitio, que software cliente se está a usar, …

Time	Source IP	Username	URL
2012/Feb/22 21:03:26	192.168.1.1	-	http://pillateunlinux.com/feed/
2012/Feb/22 21:03:28	192.168.1.1	-	http://feeds.feedburner.com/Pillateunlinux
2012/Feb/22 21:03:31	192.168.1.1	-	http://feeds.feedburner.com/KungFooSion
2012/Feb/22 21:03:32	192.168.1.1	-	http://feeds.feedburner.com/LaWeBdeDragoN
2012/Feb/22 21:03:41	192.168.1.1	-	http://feeds.feedburner.com/InformaticoYSegurata
2012/Feb/22 21:03:41	192.168.1.1	-	http://feeds.feedburner.com/Hackplayers
2012/Feb/22 21:03:51	192.168.1.1	-	http://feedproxy.google.com/Spamloco
2012/Feb/22 21:03:51	192.168.1.1	-	http://feeds.feedburner.com/S21sec
2012/Feb/22 21:03:51	192.168.1.1	-	http://feeds.feedburner.com/Spamloco
2012/Feb/22 21:03:52	192.168.1.1	-	http://www.cyberhades.com/feed/
2012/Feb/22 21:04:01	192.168.1.1	-	http://seguridadyredes.wordpress.com/feed/

A pesar de las ventajas que suponen los proxys también tienen inconvenientes:

- Se requiere un servidor proxy para cada servicio, incrementando la complejidad y posible carga del sistema donde se implante.
- Se requiere la reconfiguración de las aplicaciones cliente y en ocasiones, la modificación del software cliente.



- Algunos servicios y protocolos no pueden ser usados con servidores proxy, si bien hoy en día casi todos los principales protocolos son soportados.
- Puede generar una mayor latencia en las peticiones y respuestas a aplicaciones.

3. Proxy Web Caché

Un Proxy Caché almacena en una caché las páginas web visitadas por los usuarios con el objetivo de mejorar la navegación de los usuarios. De esta forma se optimiza el canal de acceso a Internet de las organizaciones. Este tipo de proxy se suele usar buscando:

- Motivos de seguridad:
  - No se permite el libre acceso a Internet a los usuarios.
  - Funciona como un filtro de la información solicitada por los usuarios, bloqueando contenido dañino para los ordenadores e impidiendo el acceso a páginas no deseadas.
- Optimización del ancho de banda:
  - Cuando se visita una página por primera vez (caché MISS), se guarda una copia en caché.
  - La próxima vez que se visite una de estas páginas almacenadas, el proxy-caché la devolverá rápidamente al tener una copia en disco (caché HIT), sin necesidad de acceder al original presente en Internet.

A la hora de configurar el proxy caché hay que tener en cuenta las características del disco duro y la memoria RAM que se empleará para ubicar los elementos almacenados en la caché.

4. Servidor proxy Squid

Squid es un servidor proxy caché que soporta diversos protocolos, destacando http, https y ftp. Se viene desarrollando desde hace varios años y es ampliamente utilizado en la mayoría de sistemas operativos, incluyendo Windows, GNU/Linux y derivados de Unix.

Algunas de las características de Squid son:

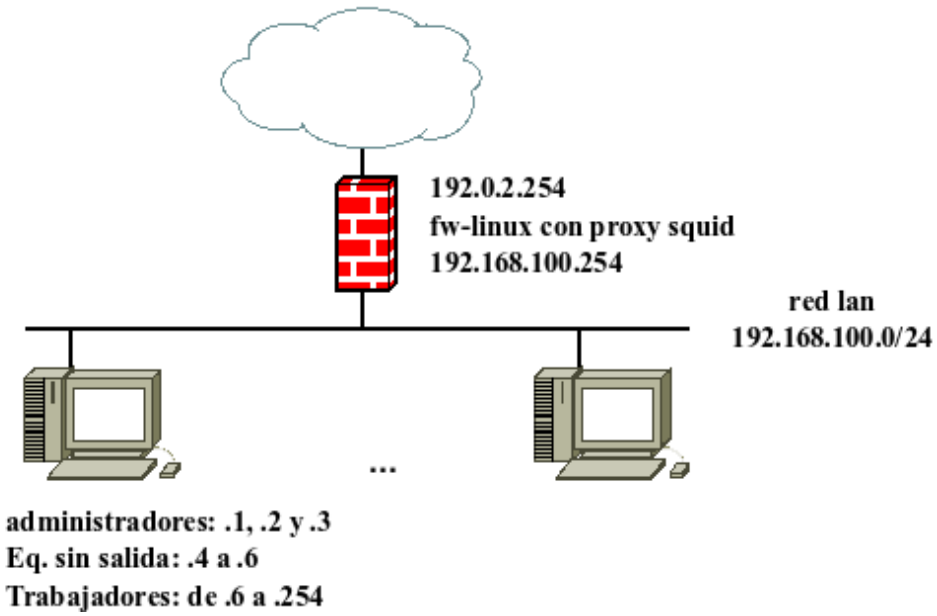
- Almacena en RAM los metadatos y los objetos más frecuentemente consultados.
- Guarda en caché las consultas DNS.
- Soporta SSL: Squid también es compatible con SSL acelerando las transacciones cifradas.
- Políticas de control de acceso: posibilidad de establecer reglas de control de acceso basadas en direcciones IP, dominios, nombre de usuario, puertos tcp, expresiones regulares, métodos http, protocolos, fechas (horas y días), número de conexiones simultáneas desde un cliente, etc.
- Permite reescrituras de consultas.
- Permite la incorporación de módulos o programas externos (plugins) para ampliar sus funcionalidades; como por ejemplo, antivirus, filtro de contidos, controles parentales, etc.
- Soporta distintos tipos de autenticación, permitiendo su integración con dominios del Directorio Activo de Microsoft.
- SNMP: Squid permite activar el protocolo SNMP, proporcionando un método simple de administración de red.
- Puede trabajar como servidor proxy inverso.

Squid en Ubuntu Linux

Dada la organización de la figura instala y configura el servidor proxy squid en el equipo Linux Ubuntu Server para establecer restricciones de acceso en base a los siguientes requerimientos:

- Equipos administradores: .1, .2 y .3
- Equipos que no pueden salir a ver páginas web: .4, .5 y .6
- Equipos trabajadores: resto de Ips de la red lan.
- La caché en disco será como mínimo de 2GBytes y para la caché en memoria RAM se reservarán 150 MBytes. En ambos casos, no se guardarán archivos con un tamaño superior a los 40Kbytes.
- Squid únicamente atenderá peticiones en la interfaz LAN en el puerto 3128/tcp.

- El servidor proxy registrará las solicitudes y rotará los logs cada 7 días.
- Los equipos de los administradores podrán salir a Internet por el proxy sin ningún tipo de restricciones.
- Los equipos de los trabajadores podrán visitar sitios web a través del proxy pero con las siguientes limitaciones:
  - Todas las páginas web de la Xunta de Galicia deben prohibirse, excepto las de la Consellería de Educación.
  - Los sitios web de los periódicos El Mundo, El Pais, La Voz de Galicia y las redes sociales Facebook y Tuenti deben estar bloqueados.



Una vez instaladas y tras hacerlas persistentes, se instala el proxy squid:

```
ubuntu@squid:~$  sudo apt update
ubuntu@squid:~$  sudo apt install squid
```

El comportamiento del servidor proxy squid viene determinado por el valor de las directivas de configuración colocadas en el archivo principal de configuración **`/etc/squid/squid.conf`**. A continuación se muestra una versión reducida del archivo squid.conf con directivas de configuración para cumplir con los requerimientos del enunciado y una explicación de las mismas:

Define en que socket (IP:puerto) atenderá a las solicitudes squid. Los puertos registrados recomendados para Servidores Intermediarios (Proxies) pueden ser el 3128 y 8080 a través de TCP. Para incrementar la seguridad, se vincula el servicio a una IP concreta:

```
http_port 192.168.100.254:3128
```

Nombre del servidor, dirección de email del administrador del proxy (aparecen en los mensajes de error) e idioma por defecto de los mensajes de error:

```
visible_hostname fw-firewall
cache_mgr admin@organizacion.org
error_default_language es-es
```

Usario/grupo con el que correrá el proceso squid después de ser iniciado por el administrador:

```
cache_effective_user proxy
cache_effective_group proxy
```

Localización de los ficheros de log:

- `access_log`: registra las actividades de los clientes (una línea para cada petición http).

- `cache_log`: registra información de carácter general sobre squid.
- `cache_store_log`: para registrar actividades del gestor de almacenamiento (objetos borrados de la caché, objetos guardados y por cuanto tiempo).

```
access_log /var/log/squid/access.log
cache_log /var/log/squid/cache.log
cache_store_log none
```

Con logfile_rotate se configura la rotación de los ficheros de log cuando se ejecute el comando `squid -k rotate`.

```
logfile_rotate 7
debug_options rotate=7
```

Definición del tamaño de la memoria física a usar para objetos en tránsito, *hot objects* y *negative-Cached objects* así como del tamaño máximo de objeto a guardar en la memoria física:

```
cache_mem 150 MB
maximum_object_size_in_memory 40 KB
```

Cuando se habla de la caché en squid hay que tener presente su estructura:

- Caché en disco: la mayor parte de los objetos cacheados por squid se guardan en el disco duro. La tecnología del disco duro y los tiempos de I/O constituyen factores clave para el rendimiento del proxy. Cuanto más rápido sea el acceso y recuperación de los elementos cacheados en el disco, más rápido se devolverán a los clientes que los solicitaron.
- Caché en RAM: squid puede cachear algunos objetos en memoria RAM para mejorar la respuesta del sistema, como por ejemplo '*hot or popular objects*' (recursos muy demandados o populares) y *negatively-cached objects* (respuestas negativas a recursos solicitados por squid).
- Índice de elementos en la caché: squid guarda en memoria RAM información relativa a los objetos cacheados en el disco duro.

Es fundamental asociar el tamaño de la caché en disco duro con el consumo de memoria RAM para mantener localizados los elementos. No se puede especificar una caché en disco muy grande si el equipo no tiene memoria RAM suficiente:

- Con squid de 32 bits, por 1 GByte de caché en disco duro hay un consumo de unos 10MB de memoria RAM.
- Con squid de 64 bits, por 1 GByte de caché en disco duro hay un consumo de unos 14MB de memoria RAM.

A mayores de los índices y la caché en RAM, squid consume memoria RAM para la caché DNS, información de estado para cada solicitud, etc. El proceso squid puede llegar a consumir grandes cantidades de memoria en función de la configuración y de la carga de trabajo; por lo que es importante monitorizar el sistema para detectar carencias de memoria RAM, para cambiar la configuración de squid y/o aumentar la memoria física delo equipo.

En este ejemplo, se va a configurar una caché en disco de 2GByte y de 150 MBytes para la caché en RAM. Estos valores junto con los tamaños máximo y mínimo de los objetos a cachear pueden cambiarse en función de las características del equipo y del tipo de tráfico de los clientes. Por ejemplo, si los usuarios acostumbran a ver recursos de tamaño más grande, se puede aumentar el tamaño máximo del objeto a cachear para ahorrar ancho de banda.

Además del tamaño de las cachés y los objetos, se deben especificar las políticas de substitución que definen que objeto va a ser reemplazado cuando se necesita espacio:

```
memory_replacement_policy heap GDSF
```

```
cache_replacement_policy heap LFUDA
```

Directorio de ubicación de la caché, tamaño y definición de los L1 y L2 de la caché, tamaños mínimo y máximo de objetos a cachear, límites de swap para la caché (valores a partir de los que se empiezan a reemplazar objetos en la caché). Dentro del directorio `/var/spool/squid` se crearán 16 subdirectorios (L1); donde a su vez y para cada uno de ellos, se crearán 256 subdirectorios (L2), que permitirán cachear objetos hasta ocupar un tamaño máximo de 2GB:

```
cache_dir ufs /var/spool/squid 2048 16 256
minimum_object_size 0 KB
maximum_object_size 40 KB
offline_mode off
cache_swap_low 90
cache_swap_high 95
```

Para crear la caché, posteriormente ejecutaremos el comando `squid -z`.

Definición de ACL Elements para indicar equipos, puertos y métodos que se usarán más adelante en las reglas de control para permitir/bloquear tráfico:

```
acl localnet src  192.168.100.0/24
acl allsrc src all
acl safeports port 21 70 80 210 280 443 488 563 591 631 777 901  3128 3127 1025-65535
acl sslports port 443 563
acl connect method CONNECT
```

Definición de ACLs para indicar equipos sin restricciones, baneados, listas blancas y negras. Fijarse que para cada ACLs se indica un fichero donde se pondrán los correspondientes equipos/dominios:

```
acl unrestricted_hosts src "/etc/squid/acls/unrestricted_hosts.acl"
acl banned_hosts src "/etc/squid/acls/banned_hosts.acl"
acl whitelist dstdomain "/etc/squid/acls/whitelist.acl"
acl blacklist dstdom_regex -i "/etc/squid/acls/blacklist.acl"
```

El contenido del fichero **`unrestricted_hosts.acl`** sería:

```
192.168.100.1
192.168.100.2
192.168.100.3
```

El contenido del fichero **`banned_hosts.acl`** sería:

```
192.168.100.4
192.168.100.5
192.168.100.6
```

El contenido del fichero **`whitelist.acl`** sería:

```
.edu.xunta.es
.edu.xunta.gal
```

El contenido del fichero **`blacklist.acl`** sería:

```
xunta.es
xunta.gal
elmundo.es
elpais.com
lavozdegalicia.es
facebook.com
tuenti.com
```

Reglas de control de acceso (*access list*) para permitir/bloquear tráfico. Con `http_access` y las ACLs definidas anteriormente se puede especificar si una request http de un cliente es aceptada

o denegada. Se procesarán por orden hasta que una de ellas determine que hacer con la solicitud (`allow/deny`); es decir, muy parecido a lo visto en los network level firewalls. Si en una regla hay varios *ACL Elements* se usa la lógica AND; es decir, deben coincidir todos para que se considere que el paquete cumple con la regla:

```
http_access deny !safeports
http_access deny connect !sslports
http_access deny banned_hosts
http_access allow unrestricted_hosts
http_access allow whitelist
http_access deny blacklist
http_access allow localnet
```

Bloquear todo el tráfico no autorizado previamente:

```
http_access deny allsrc
```

Para verificar el archivo de configuración puede ejecutarse el comando:

```
ubuntu@squid:~$ sudo squid -k parse
```

Si el comando anterior no da errores, quiere decir que la configuración tiene una sintaxis correcta.

La primera vez que se va a usar squid, y tras comprobar que no hay errores, se crea la caché:

```
ubuntu@squid:~$ sudo systemctl stop squid.service
ubuntu@squid:~$ sudo squid -z
ubuntu@squid:~$ ls -lhF /var/spool/squid/
total 89K
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 00/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 01/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 02/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 03/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 04/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 05/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 06/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 07/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 08/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 09/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0A/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0B/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0C/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0D/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0E/
drwxr-x--- 258 proxy proxy  258 Apr 25 11:32 0F/
-rw-r-----   1 proxy proxy 1.2K Apr 25 11:41 swap.state
```

Como otros servicios, squid se puede controlar a través de los comandos:

```
ubuntu@squid:~$ sudo systemctl start squid.service
ubuntu@squid:~$ sudo systemctl stopt squid.service
ubuntu@squid:~$ sudo systemctl restart squid.service
ubuntu@squid:~$ sudo systemctl status squid.service
```

Para que las modificaciones en el archivo squid.conf se tengan en cuenta se debe indicar a squid que relea el archivo con:

```
ubuntu@squid:~$ sudo squid -k reconfigure
```

Configuración de los clientes

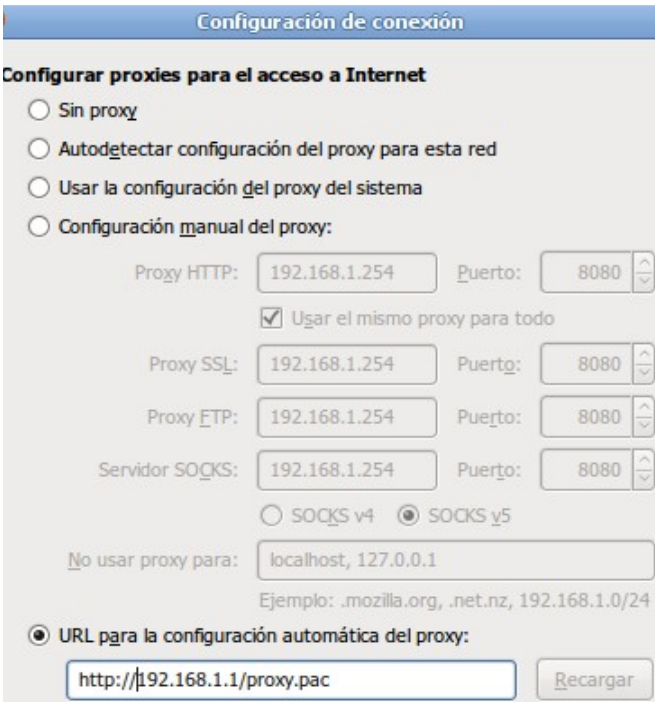
Como se indicó antes, es necesario configurar la aplicación cliente para que use el proxy. Interesa realizar esta configuración de forma automática ya que:

- Puede haber usuarios móviles, donde no interesa tener que configurar el proxy cada vez que cambia de red.
- Configurar de forma manual los clientes puede resultar una tarea costosa y tediosa.

Se han ideado algunos sistemas para configurar los clientes de forma automática:

Proxy Auto-config (PAC)

Se crea y se publica un archivo de configuración que contiene una función JavaScript que indica cual es el proxy adecuado para cada URL. A continuación, se configuran los navegadores a mano, con la URL donde se encuentra este archivo.



En el siguiente ejemplo se puede ver el contenido del archivo proxy.pac, donde se indica que se use el proxy 192.168.2.211, que está a la escucha en el puerto 8080; y en caso de fallar el proxy, que el cliente se conecte directamente.

```
function FindProxyForURL(url, host)
  {
     return "PROXY 192.168.2.211:8080; DIRECT";
  }
```

Web Proxy AutoDiscovery Protocol (WPAD)

Permite configurar los navegadores sin configuración manual, valiéndose para ello de dos métodos:

- DHCP: usando la opción 252 de `auto-proxy-config` de dhcp se informa al cliente de la ubicación del archivo `wpad.dat`.
- DNS: el cliente buscará el archivo `wpad.dat` en `http://wpad.dominio.com/wpad.dat` siendo dominio.com el dominio al que pertenece el cliente.

El fichero `wpad.dat` o `proxy.pac` puede ser más complejo; ya que, es posible definir diferentes proxys en función de los recursos a visitar e incluso varios proxys para obtener balanceo de carga o tolerancia a fallos:

```
   function FindProxyForURL(url, host) {
     // para los dominios locales no se usa proxy
     if (shExpMatch(host, "*.dominio.com"))
     {
        return "DIRECT";
     }
     // Para URLs dentro de esta red se usará el proxy 192.168.2.212:8080
     if (isInNet(host, "172.31.1.0",  "255.255.255.0"))
     {
        return "PROXY 192.168.2.211:8080";
     }
     // Para el resto de solicitudes se podrá usar alguno de los siguientes proxys
     // En caso de no estar operativos, se saldrá directamente.
     return "PROXY 192.168.2.210:8080; PROXY 192.168.2.212:8080; DIRECT";
  }
```