GOBIERNO DE ESPAÑA
MINISTERIO DE EDUCACIÓN, POLÍTICA SOCIAL Y DEPORTE
SECRETARÍA DE ESTADO DE EDUCACIÓN Y FORMACIÓN
DIRECCIÓN GENERAL DE FORMACIÓN PROFESIONAL
INSTITUTO SUPERIOR DE FORMACIÓN Y RECURSOS EN RED PARA EL PROFESORADO

REDES DE ÁREA LOCAL: APLICACIONES Y SERVICIO EN LINUX

Enrutamiento y Proxy

Formación en Red

Índice de contenido

Enrutamiento

Se puede definir el enrutamiento como la capacidad de transmitir datos entre redes interconectadas. Al agente encargado de realizar este encaminamiento de información entre redes se conoce como **enrutador o router** pudiendo ser de tipo hardware si es un dispositivo físico dedicado al encaminamiento y de tipo software en caso de ser un PC que ejecuta una aplicación que realice las funciones propias del enrutamiento.

Con el software adecuado, nuestro servidor Linux podrá actuar de enrutador en nuestra red de manera que permitirá que los equipos de la red local se conecten a Internet como si lo hicieran a través de un router.

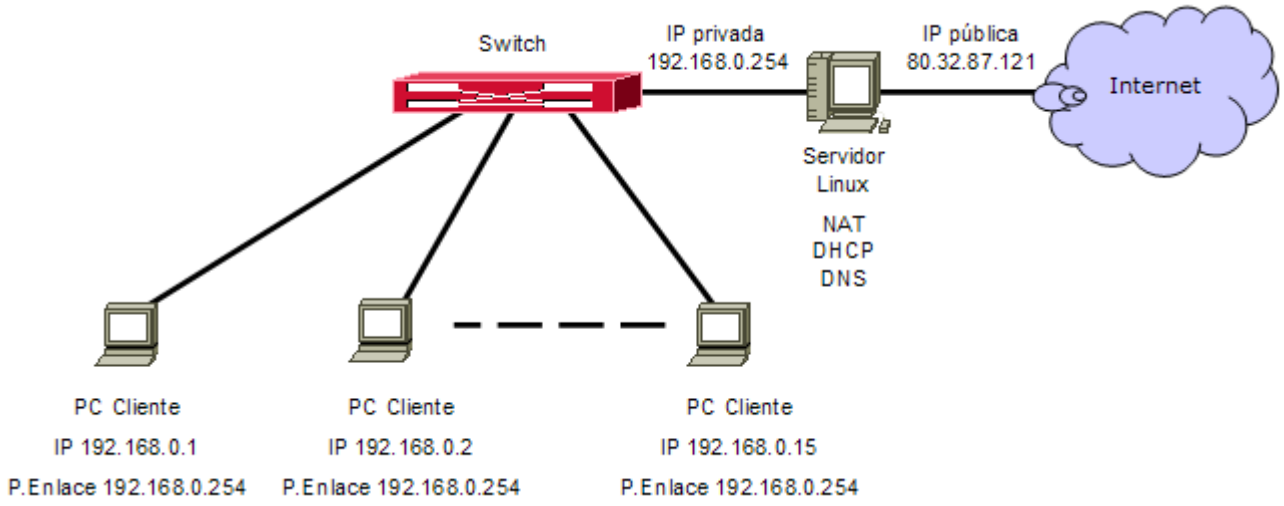
La tecnología empleada para permitir que los equipos de la red local se conecten a Internet a través de nuestro servidor Linux se denomina NAT - Network Address Traslation (Traducción de Direcciones de Red). El software NAT que se ejecuta en nuestro servidor permite, que con una única dirección IP pública en el servidor, tengan acceso a Internet el resto de PCs de la red.

En los PCs de la red local se deberá configurar como puerta de enlace (gateway) la dirección IP interna del servidor para que sea éste quien reciba y procese los paquetes provenientes de la red interna y con destino hacia Internet.

Cuando desde un PC de la red local se quiere acceder a Internet, el paquete de datos se enviará al servidor linux ya que es la puerta de enlace. El software NAT del servidor cambiará en el paquete de datos la dirección IP de origen del PC de la red local por la dirección IP pública del servidor y lanzará el paquete de datos hacia Internet. En una tabla interna almacenará el puerto de salida del paquete junto con la IP del PC de la red local

con la finalidad de que cuando llegue la respuesta desde Internet, realizar el proceso inverso y poder redirigirlo hacia el PC que lanzó la petición.

Si nuestro servidor Linux, dispone además de servidor DHCP, la configuración de las direcciones IP, la puerta de enlace y el servidor DNS de nuestros PCs, podrá ser establecida automáticamente por el servidor DHCP.



Una alternativa podría ser instalar en el servidor un proxy como **squid**, de esa forma las páginas accedidas por los clientes serían cacheadas en el servidor con lo cual se aceleraría la conexión a Internet, especialmente cuando son muchos los clientes que acceden a los mismos sitios. Un proxy facilita también el control de la conexión impidiéndola o restringiéndola a medida de nuestras necesidades. El inconveniente de compartir una conexión a Internet con un proxy es que trabaja a nivel de aplicación y por tanto del protocolo de cada aplicación (HTTP, FTP, SMTP, etc...). Esto obliga a configurar las aplicaciones (navegador, clientes de correo, clientes ftp, etc...) para que utilicen el proxy, cosa que no es necesario hacer cuando se dispone de un router ya que el router NAT trabaja a nivel de red TCP/IP y es totalmente trasparente a las aplicaciones.

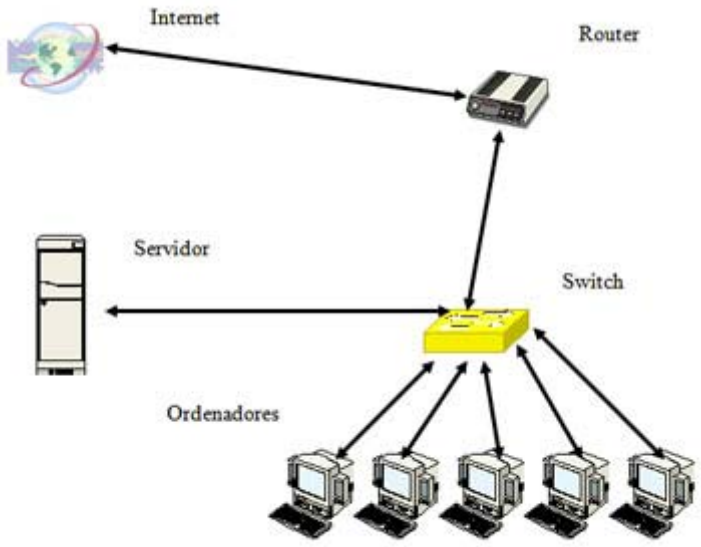
Otro servicio que se podría disponer en el servidor es un cortafuegos como **iptables** que permite filtrar qué paquetes de datos pueden entrar y qué paquetes de datos pueden salir, con la finalidad de controlar el acceso a Internet y ganar en seguridad frente a ataques externos.

Más adelante veremos una configuración básica de **iptables** que nos permitirá permitir o denegar las conexiones a diferentes redes y puertos, así como una configuración básica de **squid** para poder compartir y controlar la conexión a Internet mediante el proxy.

Enrutamiento en Linux

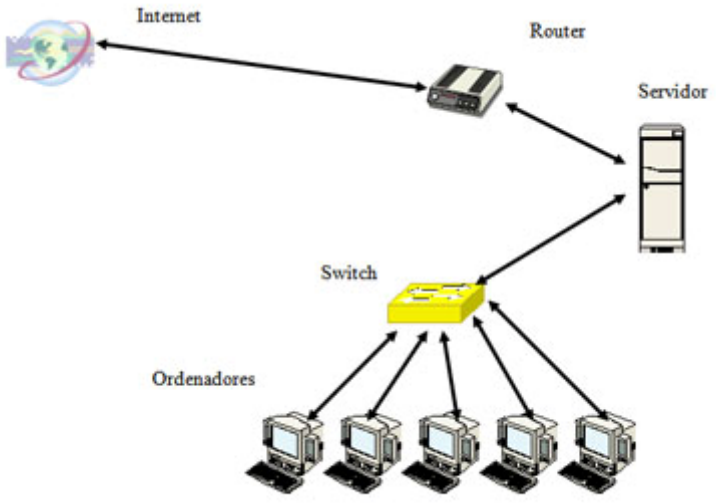
En nuestro Centro Educativo hemos venido detectando problemas de saturación de la línea de conexión a Internet sin motivo justificado. Hemos detectado que en algún ordenador de la sala de profesorado y de algún departamento hay instalados programas de P2P (descarga masiva) y somos conscientes de que estos programas saturan el canal de salida a Internet del centro, además sospechamos que el alumnado también utiliza este tipo de programas.

El router ADSL está conectado a un switch y por lo tanto a través de múltiples utilidades es fácil conocer su dirección IP y configurar nuestro equipo como puerta de enlace, con el consiguiente acceso libre a Internet y a la descarga masiva. Nos encontramos con un esquema del tipo:



Este esquema no permite controlar el tráfico de red puesto que los PCs tienen acceso directo al router.

Situando el servidor entre la red y el router, todo el tráfico hacia Internet pasa por el servidor lo que nos permitirá analizarlo, generar estadísticas, filtrar accesos, instalar un proxy-caché, etc., de forma sencilla y centralizada.



Activación del enrutamiento en Linux

Las funciones de enrutamiento mediante NAT son realizadas por el cortafuegos que analizará los paquetes provenientes de la red local interna cuyo destino sea Internet y los modificará convenientemente para que salgan hacia Internet como si fueran emitidos por el servidor. A partir del nucleo 2.4 de Linux, el cortafuegos empleado es **iptables**.

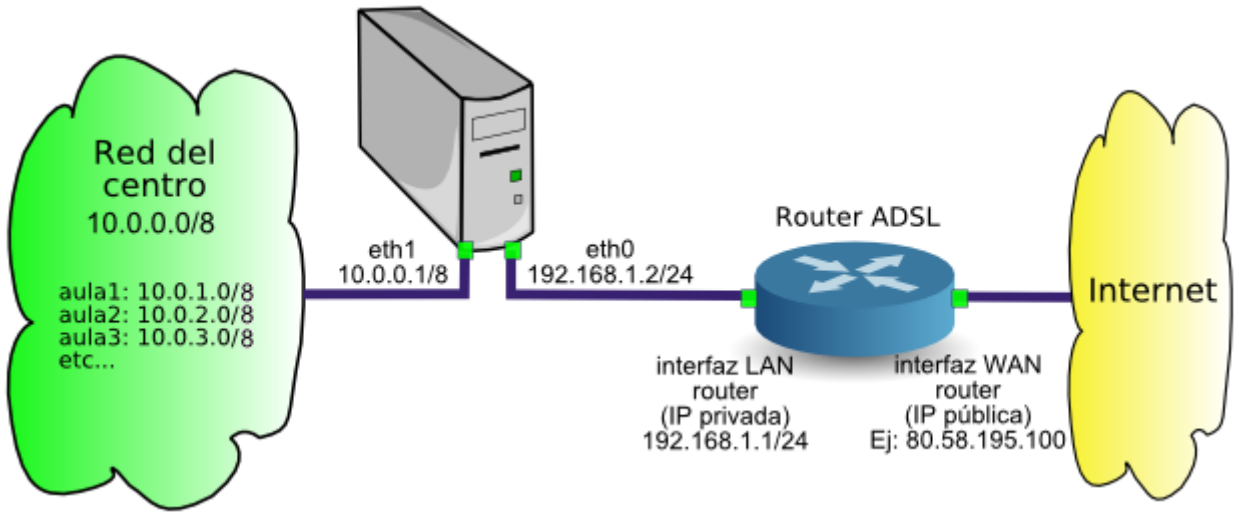
Para posibilitar que nuestro servidor Linux sea capaz de comportarse como un router y hacer de puerta de enlace para los PCs de nuestra red local, será necesario crear un script que configure el cortafuegos iptables para que realice NAT desde dentro de la red local hacia Internet.

Creación del script para activar enrutamiento

Para activar el enrutamiento en un sistema Linux, tan solo basta con poner a '1' la variable ip_forward del sistema, es decir, basta con ejecutar desde una consola de root:

```
// Activar el enrutamiento en un sistema Linux
# echo "1" > /proc/sys/net/ipv4/ip_forward
```

Posteriormente tendríamos que configurar el filtrado para que acepte el redireccionamiento de paquetes desde dentro hacia fuera de nuestra red y mediante NAT permita que los PCs de la red interna naveguen con la dirección IP 'publica' del servidor. Supongamos que el router Linux tiene una tarjeta (eth0) configurada con la IP 192.168.1.2/24 y conectada al router, cuya IP es 192.168.1.1/24, y por otro lado, tenemos otra tarjeta (eth1) configurada con la ip 10.0.0.1/8 y conectada al switch para dar servicio a nuestra red interna que utiliza el rango 10.0.0.0/8. Nuestro esquema sería como el que vemos en la siguiente figura:



Router Linux

Tendríamos que indicar que se acepten todos los paquetes que son para reenviar, es decir, aquellos que llegan a nuestra máquina pero que no es ella la destinataria. Para ello, tendríamos que aceptar los paquetes de tipo FORWARD, como veremos en la siguiente sección. Por otro lado, tendríamos que indicar que los paquetes que llegan desde nuestra red interna (-s 10.0.0.0/8) y que salgan por la interfaz eth0 hacia el router (-o eth0), después de enrutarlos en nuestra máquina (POSTROUTING), debemos enmascararlos (MASQUERADE), es decir, hacer NAT. Los comandos a ejecutar serían:

```
// Haciendo NAT en el servidor
# iptables -A FORWARD -j ACCEPT
# iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -o eth0 -j MASQUERADE
```

Podríamos realizar un script que activara el enrutamiento y el NAT y otro para desactivarlo:

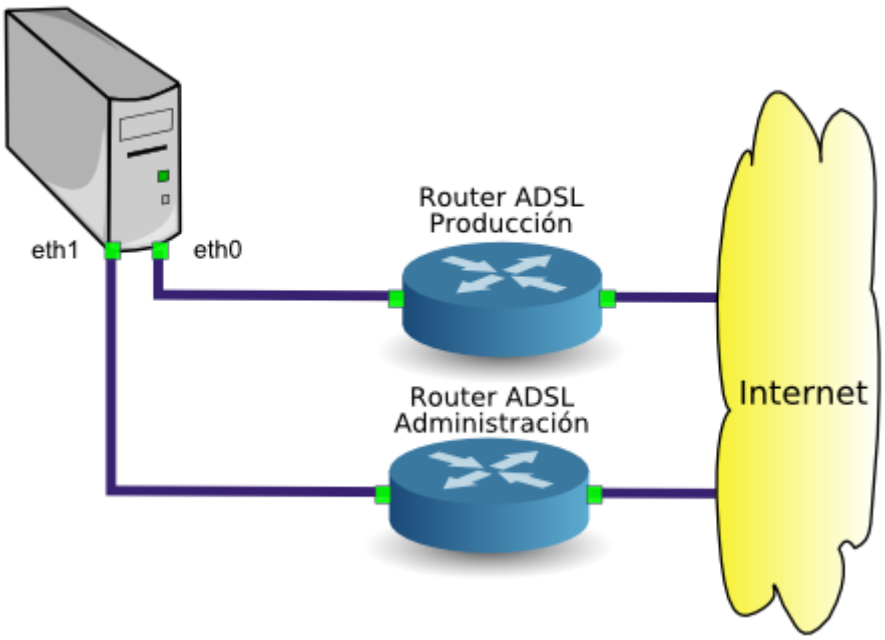
```
// activar-enrutamiento.sh
echo "1" > /proc/sys/net/ipv4/ip_forward
iptables -A FORWARD -j ACCEPT
iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -o eth0 -j MASQUERADE

// desactivar-enrutamiento.sh
echo "0" > /proc/sys/net/ipv4/ip_forward
```

Así, nuestro servidor se convertiría en un router. Si todas las comunicaciones de la red pasan por nuestro servidor, podremos tenerlas controladas, como veremos en las siguientes secciones.

Crear y eliminar rutas fijas

Cuando activamos el enrutamiento en Linux, nuestra máquina se convierte en un router automático, de forma que todo lo que entre por la interfaz eth0 con destino a una red diferente de la definida en eth0, lo reenviará por la interfaz eth1 y de igual forma, todo lo que entre por la interfaz eth1 con destino a una red diferente de la definida en eth1, lo reenviará por la interfaz eth0. Es el funcionamiento normal de un router, enrutar todo.

En algunos casos, puede que nos interese que ciertos paquetes salgan por una interfaz concreta. Por ejemplo, supongamos que en nuestra red disponemos de dos conexiones ADSL independientes, una para dar servicio de conexión a Internet al servidor (interfaz de producción) y otra, para conectarnos desde nuestra casa al servidor, para realizar tareas de administración (interfaz de administración).Supongamos que la interfaz **eth0** está conectada al router ADSL de **producción** y la interfaz **eth1** está conectada al router ADSL para realizar tareas de **administración**.



Rutas fijas

Lo normal es que la interfaz eth0 tenga configurada como puerta de enlace la IP del router de conexión a Internet, pero la interfaz eth1 no debería tener configurada la puerta de enlace, para que no exista tráfico hacia Internet por dicha interfaz. Si en el ADSL de nuestra casa tenemos IP fija, podemos crear una ruta para que cuando la IP destino sea la **IP fija** de nuestra casa, los paquetes se enruten por eth1 en lugar de hacerlo por eth0. Ejemplo, si nuestra IP de casa es 80.58.12.27, el comando a ejecutar será:

```
//Crear una ruta para una IP concreta
# route add 80.58.12.27 eth1
```

En lugar de una IP concreta, quizás nos interese crear una ruta para toda una **red**. Supongamos que queremos que cuando la IP destino sea una IP del CNICE, salga por la interfaz eth1. Teniendo en cuenta que el rango de IPs públicas del CNICE es 192.144.238.0/24, el comando a ejecutar sería:

```
//Crear una ruta para una red concreta
# route add -net 193.144.238.0/24 eth1
```

Si queremos **eliminar** una ruta, utilizaremos el parámetro 'del' seguido de la IP o la red destinataria. Ejecutaríamos el siguiente comando:

```
//Eliminar una ruta
# route del -net 193.144.238.0/24
```

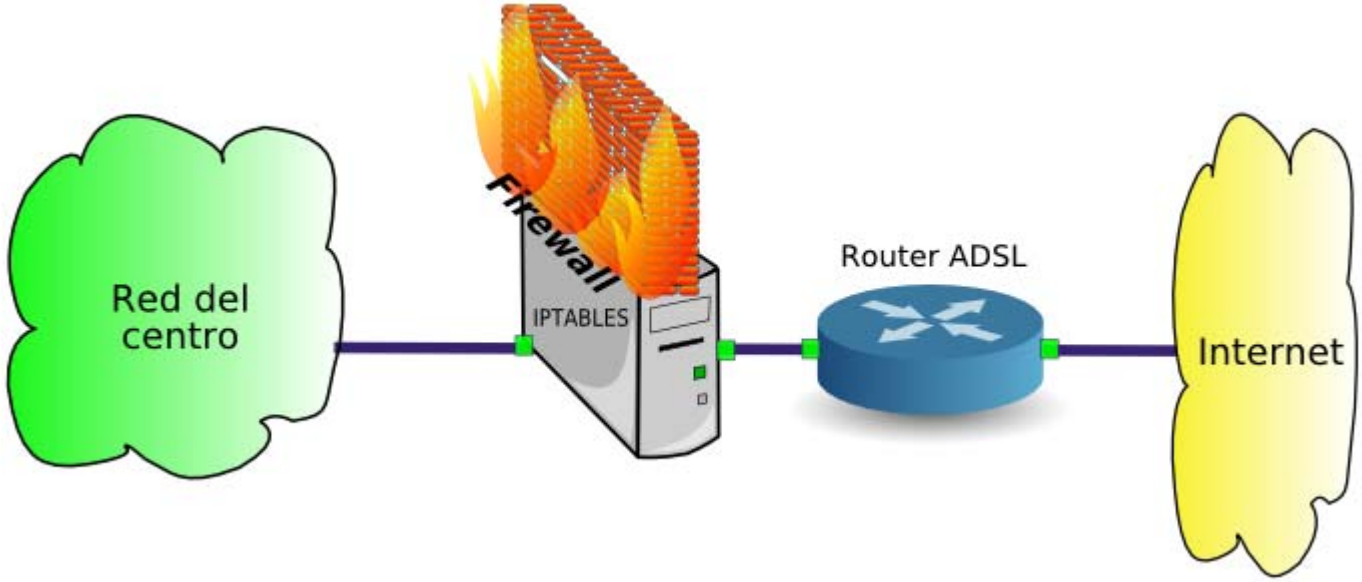
Si queremos **ver** la configuración de la tabla de rutas, debemos ejecutar el comando route sin parámetros:

```
//Ver rutas
# route
```

Establecer rutas puede ser muy interesante cuando queremos dividir nuestra red en diferentes subredes y disponemos de un servidor con varias tarjetas de red.

Cortafuegos iptables

Desde la versión 2.4 del núcleo de linux, el cortafuegos utilizado para gestionar las conexiones es **iptables**. Las posibilidades de iptables son prácticamente infinitas y un administrador que quiera sacarle el máximo provecho, puede realizar configuraciones extremadamente complejas. Para simplificar, diremos que básicamente, iptables permite crear reglas que analizarán los paquetes de datos que entran, salen o pasan por nuestra máquina, y en función de las condiciones que establezcamos, tomaremos una decisión que normalmente será permitir o denegar que dicho paquete siga su curso.



El cortafuegos controla las comunicaciones entre la red y el exterior

Para crear las reglas, podemos analizar muchos aspectos de los paquetes de datos. Podemos filtrar paquetes en función de:

Tipo de paquete de datos:
- Tipo INPUT: paquetes que llegan a nuestra máquina
- Tipo OUTPUT: paquetes que salen de nuestra máquina
- Tipo FORWARD: paquetes que pasan por nuestra máquina

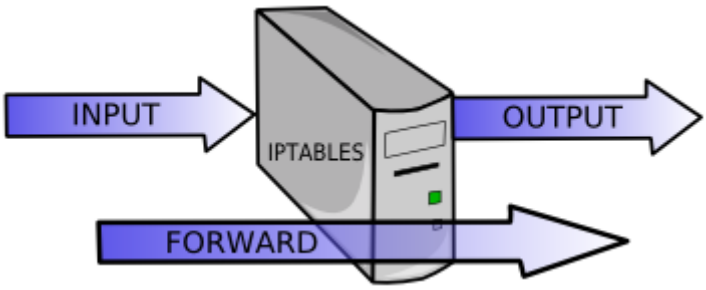
Interfaz por la que entran (-i = input) o salen (-o = output) los paquetes
- eth0, eth1, wlan0, ppp0, ...

IP origen de los paquetes (-s = source)
- IP concreta, ej: 10.0.1.3
- Rango de red, ej: 10.0.1.0/8

IP destino de los paquetes (-d = destination)
- IP concreta, ej: 10.0.1.3
- Rango de red, ej: 10.0.1.0/8

Protocolo de los paquetes (-p = protocol)
- tcp, udp, icmp...

Hacer NAT (modificar IP origen y destino para conectar nuestra red a otra red o a Internet) y...
- Filtrar antes de enrutar: PREROUTING
- Filtrar después de enrutar: POSTROUTING



Los paquetes pueden entrar, salir o pasar

Una forma sencilla de trabajar con iptables es permitir las comunicaciones que nos interesen y luego denegar el resto de las comunicaciones. Lo que se suele hacer es definir la política por defecto aceptar (ACCEPT), después crear reglas concretas para permitir las comunicaciones que nos interesen y finalmente, denegar el resto de comunicaciones. Lo mejor será crear un script en el que dispondremos la secuencia de reglas que queremos aplicar en nuestro sistema. Un ejemplo típico podría ser el siguiente:

```
#!/bin/sh
# Script cortafuegos.sh para la configuración de iptables
#
# Primero borramos todas las reglas previas que puedan existir
iptables -F
iptables -X
iptables -Z
iptables -t nat -F

# Después definimos que la politica por defecto sea ACEPTAR
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT

# Para evitar errores en el sistema, debemos aceptar
# todas las comunicaciones por la interfaz lo (localhost)
iptables -A INPUT -i lo -j ACCEPT

# Aceptamos las comunicaciones que nos interesan y luego denegamos el resto.

# Ejemplo: Denegamos acceso al aula 1
iptables -A FORWARD -s 10.0.1.0/24 -j DROP

# Aceptamos SMTP, POP3 y FTP (correo electrónico y ftp)
iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 25 -j ACCEPT
iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 110 -j ACCEPT
iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 20 -j ACCEPT
iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 21 -j ACCEPT

# HTTP y HTTPS no es necesario porque nuestro servidor será servidor proxy
# Dejamos comentadas las líneas, por si algún día las necesitamos
#iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 80 -j ACCEPT
#iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 443 -j ACCEPT

# DNS no es necesario porque nuestro servidor será servidor DNS
# Dejamos comentadas las líneas (tcp y udp), por si algún día las necesitamos
#iptables -A FORWARD -s 10.0.0.0/8 -p tcp --dport 53 -j ACCEPT
#iptables -A FORWARD -s 10.0.0.0/8 -p udp --dport 53 -j ACCEPT
```

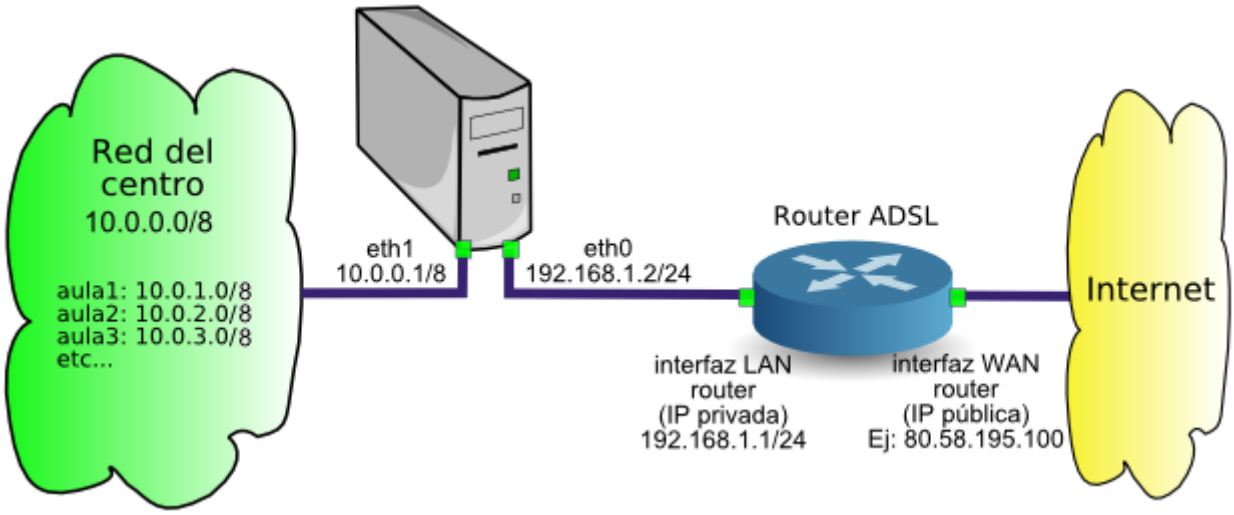
```
# Al PC del Director le damos acceso a todo (cliente VIP)
iptables -A FORWARD -s 10.0.0.7 -j ACCEPT

# Denegamos resto de comunicaciones (no funcionará el p2p)
iptables -A FORWARD -s 10.0.0.0/8 -j DROP

# Hacemos NAT si IP origen 10.0.0.0/8 y salen por eth0
iptables -t nat -A POSTROUTING -s 10.0.0.0/8 -o eth0 -j MASQUERADE

# Activamos el enrutamiento
echo 1 > /proc/sys/net/ipv4/ip_forward

# Comprobamos cómo quedan las reglas
iptables -L -n
```

En el script anterior vemos una serie de reglas que se van a ir ejecutando secuencialmente conformando la configuración del cortafuegos iptables. Cuando indicamos "-A FORWARD" nos referimos a paquetes que van a pasar por nuestro servidor. Otras opciones son "-A INPUT" y "-A OUTPUT". Acto seguido ponemos las condiciones. Si ponemos "-s 10.0.0.0/8" nos referimos a paquetes cuya IP origen está en el rango 10.0.0.0/8. Cuando ponemos "-p tcp" nos referimos a paquetes que utilizan el protocolo tcp. Cuando indicamos "--dport 25" nos referimos a paquetes cuyo puerto de destino es el 25, es decir, protocolo SMTP (correo saliente). Si en una regla no ponemos la condición -p ni la condición --dport, significa que no nos importa el protocolo (cualquier protocolo) ni nos importa el puerto destino (cualquier puerto destino). Por ejemplo, en la regla donde damos acceso al PC del Director, no hemos indicado ni protocolo ni puerto, por lo que dejará pasar todos los protocolos y todos los puertos.

Proxy Squid

Introducción

Un proxy de conexión a Internet es un servidor que hace de **intermediario** entre los PCs de la red y el router de conexión a Internet, de forma que cuando un usuario quiere acceder a Internet, su PC realiza la petición al servidor Proxy y es el Proxy quien realmente accede a Internet. Posteriormente, el Proxy enviará los datos al PC del usuario para que los muestre en su pantalla. El PC del usuario no tendrá conexión directa con el router, sino que accederá a Internet por medio del proxy.



El proxy es un intermediario

Ventajas de disponer de un proxy:

-Los PCs de los usuarios **no tienen acceso al router**, todas las comunicaciones exteriores pasarán por el Proxy, lo que nos permitirá tener las comunicaciones bajo control. Podemos permitir o denegar el acceso web, ftp, email, messenger, p2p, etc...

-Las páginas se **cachean** en la memoria temporal del proxy, lo cual, acelera la descarga cuando varios usuarios acceden a las mismas páginas a la vez. Esta circunstancia se da mucho en los centros educativos cuando el profesor está explicando un tema y todos los alumnos acceden a la vez a la misma página.

-Es fácil crear una lista de **urls prohibidas** a las que el proxy denegará el acceso.

-Es fácil permitir o denegar el acceso a **subredes** o a PCs concretos. Si diseñamos la red de forma que cada aula del centro tenga un rango determinado, por ejemplo 10.0.X.Y donde X es el número de aula e Y el número de PC, sería posible permitir o denegar la conexión a Internet aula por aula.

-El proxy guarda **informes** de todas las conexiones que hacen los usuarios. Al principio puede ser interesante ver a qué páginas de contenido inadecuado acceden nuestros alumnos, para agregarlas a la lista de urls prohibidas.

-Los PCs de nuestra red están más **seguros** de ataques externos ya que el proxy hace de barrera cortafuegos.

Inconvenientes de la utilización de un Proxy:

No todo son ventajas, también hay algún inconveniente en la utilización de un Proxy:

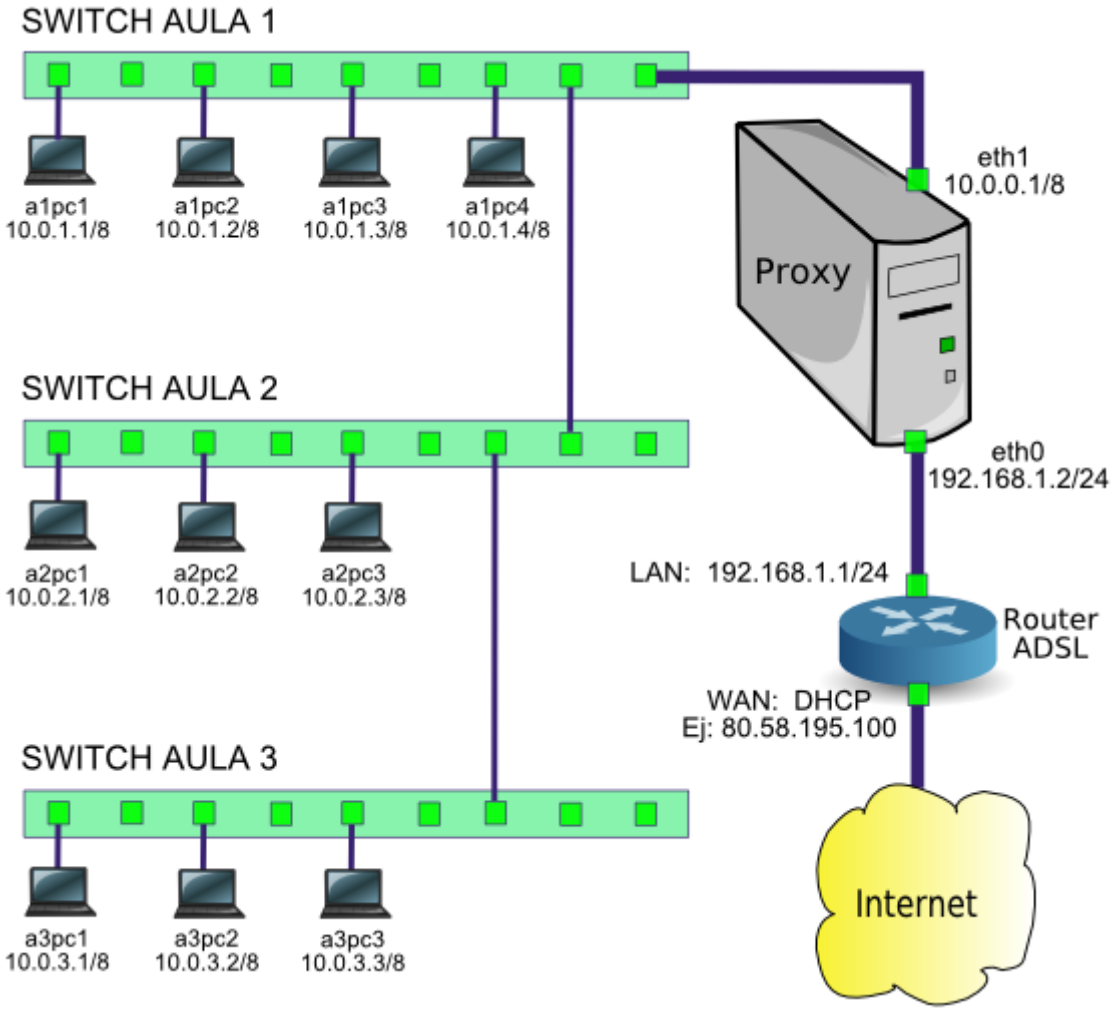
-Para que las aplicaciones accedan a Internet a través del proxy, es **necesario configurar** cada aplicación: navegador web, cliente ftp, cliente de correo, etc...

-Todas las comunicaciones con el exterior pasarán por el servidor. Si el proxy **falla**, la red se quedará sin conexión a Internet. Para subsanar lo más rápidamente posible el problema ante un fallo del Proxy, será conveniente disponer de un proxy de repuesto.

-El proxy requiere **mantenimiento**. Para que todo funcione, es necesario que exista un administrador de la red que se encargue de actualizar, revisar, mantener y reparar el proxy cuando deje de funcionar.

Diseño recomendado de la red del centro

Para facilitar la gestión del acceso a Internet en el centro, se recomienda diseñar la red de forma que cada aula tenga un rango de IPs determinado. Para no quedarnos cortos, lo mejor es utilizar el rango 10.0.0.0/8 siguiendo el esquema 10.W.X.Y donde W sería el número de edificio, X el número de aula e Y el número de PC, que nos permitiría tener un máximo de 254 edificios con 254 aulas cada uno y 254 PCs por aula. Si disponemos de un único edificio con tres aulas, un sencillo esquema de direccionamiento IP podría ser el siguiente:



Direccionamiento de nuestra red

Direccionamiento IP recomendado para nuestra red:

-Utilizar el rango **10.0.0.0/8** para el direccionamiento de red del centro educativo.

-Utilizar la IP **10.0.0.1** para el servidor proxy. Conviene que dicho servidor sea también servidor DNS.

-Las aulas usarán la dirección **10.0.X.Y** donde X sea el número de aula e Y sea el número de PC. Ejemplo, si en la aula 1 hay 4 PCs, en el aula 2 hay 3 y en el aula 3 hay 3, el direccionamiento sería:

Aula	PC	Nom.	IP	Máscara	P.Enlace	DNS
1	1	a1pc1	10.0.1.1	255.0.0.0 sin configurar	10.0.0.1	
1	2	a1pc2	10.0.1.2	255.0.0.0 sin configurar	10.0.0.1	
1	3	a1pc3	10.0.1.3	255.0.0.0 sin configurar	10.0.0.1	
1	4	a1pc4	10.0.1.4	255.0.0.0 sin configurar	10.0.0.1	
2	1	a2pc1	10.0.2.1	255.0.0.0 sin configurar	10.0.0.1	
2	2	a2pc2	10.0.2.2	255.0.0.0 sin configurar	10.0.0.1	
2	3	a2pc3	10.0.2.3	255.0.0.0 sin configurar	10.0.0.1	
3	1	a3pc1	10.0.3.1	255.0.0.0 sin configurar	10.0.0.1	
3	2	a3pc2	10.0.3.2	255.0.0.0 sin configurar	10.0.0.1	
3	3	a3pc3	10.0.3.3	255.0.0.0 sin configurar	10.0.0.1	

Instalación del Proxy squid

Linux dispone del Proxy **squid**. Se trata de una aplicación de gran éxito que se lleva utilizando muchos años y dispone de cientos de posibilidades para personalizar su funcionamiento a nuestras necesidades. Para instalar la última versión de squid, podemos hacerlo con apt-get desde una consola de root:

```
// Instalación del servidor Proxy squid
# apt-get install squid
```

De esta forma instalaríamos los programas necesarios para disponer de un completo servidor Proxy en nuestra red. Tan sólo será necesario configurarlo y ponerlo en marcha.

Arranque y parada del proxy squid

El servicio squid, al igual que todos los servicios, dispone de scripts de arranque y parada en la carpeta /etc/init.d. Debemos ejecutarlos desde una consola de root.

```
// Arrancar o reiniciar el servidor squid
# /etc/init.d/squid restart

// Parar el servidor squid
# /etc/init.d/squid stop

// Recargar configuración del servidor squid
# /etc/init.d/squid reload
```

Para un arranque automático del servicio al iniciar el servidor, debemos crear los enlaces simbólicos correspondientes tal y como se indica en el apartado Arranque automático de servicios al iniciar el sistema.

Configuración básica del proxy squid

El archivo de configuración del proxy es el archivo **/etc/squid/squid.conf**. Si le abrimos con un editor de textos, veremos que es un archivo muy extenso en el que hay cientos de

parámetros que podemos establecer, pero para una utilización básica, son unos pocos los parámetros que debemos configurar. De todos los apartados que dispone el archivo /etc/squid/squid.conf, sólo destacaremos los siguientes:

OPTIONS FOR AUTHENTICATION (Opciones de autentificación)
Aquí se establecen las opciones de autentificación del Proxy. Aunque en este artículo no vamos a hablar de ello, existe la posibilidad de configurar squid para que solicite usuario y contraseña para poder navegar por Internet. Si se quiere hacer uso de esta funcionalidad, lo normal sería tener almacenados los usuarios y las contraseñas en un servidor LDAP y en función de los grupos a los que pertenezcan los usuarios, podríamos habilitar o deshabilitar el acceso. Esto puede ser interesante en empresas, donde el administrador de red da acceso a Internet solo a los usuarios que lo necesitan. En un centro educativo supondría bastante trabajo llevar una administración de este tipo ya que habría que crear y gestionar un usuario para cada alumno y para cada profesor. Es más fácil administrar por redes y por aulas.

ACCESS CONTROL (Control de Acceso)
En esta sección estableceremos los permisos de acceso, es decir, quien puede navegar y quien no. Lo primero que tendremos que hacer es crear listas de control de acceso (Access Control List - ACL) y luego dar permisos a dichas listas.

Una lista de control de acceso (acl) se crea utilizando la palabra acl seguido del nombre que queramos dar a la lista y seguido de una condición que cumplirán los miembros de la lista. Entre las condiciones más utilizadas destacamos: src (IPs o URLs origen), dst (IPs o URLs destino), port (puertos) y proto (protocolos). Ejemplos:

Si en mi red local utilizo el direccionamiento 10.0.0.0/8, puedo crear una lista para definir a toda mi red:

```
//acl para definir toda mi red
acl todos src 10.0.0.0/8
```

Si en mi red local utilizo el direccionamiento 10.0.X.0/24, para el aula X, puedo crear una lista para cada aula:

```
//Una acl para cada aula
acl aula1 src 10.0.1.0/24
acl aula2 src 10.0.2.0/24
acl aula3 src 10.0.3.0/24
acl aula4 src 10.0.4.0/24
acl aula5 src 10.0.5.0/24
```

Luego tendría que dar permiso a las listas. Para ello se utiliza la palabra clave **http_access** seguido del permiso allow (permitir) o deny (denegar) y seguido del nombre de la lista. Ejemplos:

Si quiero dar permiso a toda mi red para que navegue por Internet:

```
//Permiso para que navegue toda mi red
http_access allow todos
```

Si quiero dar permiso a las aulas 1, 2 y 3 para que navegue por Internet pero no quiero que naveguen las aulas 4 y 5:

```
//Permiso para que naveguen las aulas 1, 2 y 3 y no naveguen las aulas 4 y 5
http_access allow aula1
http_access allow aula2
http_access allow aula3
http_access deny aula4
http_access deny aula5
```

Por defecto, squid viene configurado para actuar como **caché** de acceso a Internet, pero no tiene creadas listas de control de acceso. Si configuramos el navegador de Internet de los PCs cliente para que utilicen el Proxy, veremos que tenemos denegado el acceso al Proxy. Para empezar a disfrutar del Proxy, tendremos que crear una lista de control de acceso con el rango de nuestra red y darla permiso. Si en nuestra red utilizamos el rango 10.0.0.0/8, deberíamos añadir en /etc/squid/squid.conf:

```
//Permiso para que navegue toda mi red.
acl todos src 10.0.0.0/8
http_access allow todos
```

Cuando creamos acls, podemos sustituir el rango de IPs por el nombre de un archivo externo, y de esa manera podemos indicar el en archivo externo el rango o los rangos de IPs a los que queremos referirnos, sin necesidad de estar continuamente modificando el archivo squid.conf. Más adelante veremos un ejemplo cómo tener un archivo externo con las urls prohibidas a las que no podrán navegar nuestros alumnos.

NETWORK OPTIONS (Opciones de red)
En esta sección estableceremos con el parámetro http_port, el puerto en el que escucha el Proxy. Lo mejor es dejar el valor por defecto que es el puerto 3128:

```
//Configurar squid en el puerto 3128
http_proxy 3128
```

Squid puede trabajar en modo **transparente**. La ventaja de configurar squid en dicho modo de trabajo, es que no sería necesario configurar el navegador de los PCs clientes para trabajar con el proxy, sino que simplemente configuramos la puerta de enlace del PC cliente con la IP del servidor proxy. Posteriormente tendremos que configurar el cortafuegos del servidor para que redirija las peticiones al puerto 80 hacia el puerto 3128 y así las reciba squid. Si deseamos poner el Proxy en modo transparente, deberemos indicarlo después del puerto. En tal caso, el parámetro http_port quedaría así:

```
//Configurar squid en el puerto 3128, en modo transparente
http_proxy 3128 transparent

//Redirigir las peticiones al puerto 80 hacia el puerto 3128. Ejecutar como root:
# iptables -t nat -A PREROUTING -i eth1 -p tcp --dport 80 -j REDIRECT --to-port 3128
```

MEMORY CACHE OPTIONS
En esta sección estableceremos la memoria RAM utilizada para la caché. Una buena

opción es utilizar sobre un tercio de la memoria RAM del sistema. Ejemplo, si nuestro sistema tiene 512 MB de memoria RAM, una buena opción sería:

```
//RAM utilizada por squid
cache_mem 192 MB
```

DISK CACHE OPTIONS
En esta sección estableceremos el espacio de disco duro utilizado para la caché. Una buena opción es utilizar el 50% de la capacidad total del disco duro. Ejemplo, si nuestro disco tiene sistema tiene 80 GB de memoria RAM, una buena opción sería utilizar 40 GB. Deberemos utilizar la palabra clave cache_dir seguida de la palabra ufs que es el formato utilizado por squid, de la carpeta donde queremos que se almacene la cache, el tamaño de la caché en MB, el número de subdirectorios de primer nivel y el número de subdirectorios de segundo nivel. Ejemplo, si queremos que la caché se guarde en /var/spool/squid, que utilice 40 GB y que cachee hasta 16 subdirectorios de primer nivel y hasta 256 subdirectorios de segundo nivel, escribiremos:

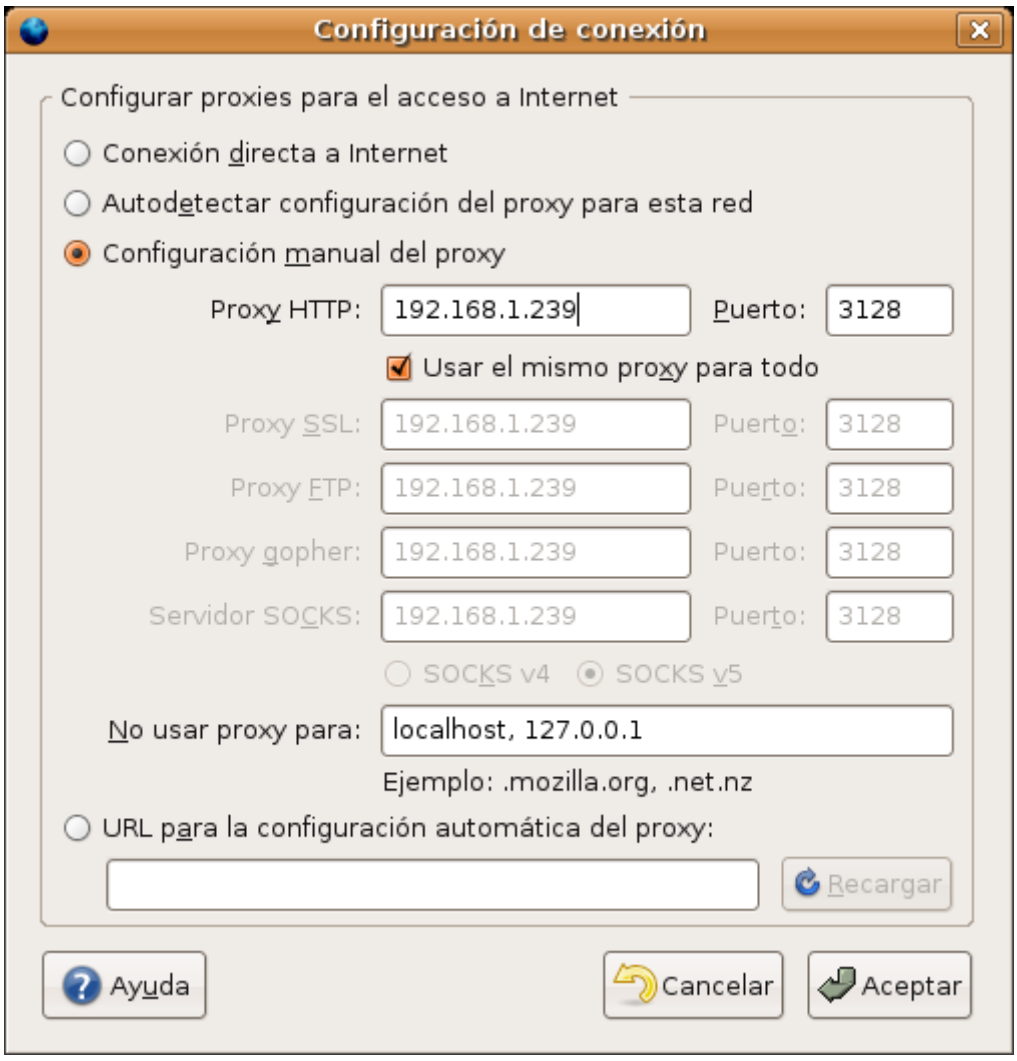
```
//Espacio en disco utilizado por squid
cache_dir ufs /var/spool/squid 40000 16 256
```

Configuración del navegador de los PCs clientes, para que utilicen el Proxy

Supongamos que nuestro servidor Proxy tiene la IP 192.168.1.239 y el servidor squid está escuchando en el puerto 3128 que es el puerto que utiliza por defecto. Con estos dos datos, la IP y el puerto, ya podemos configurar el navegador de Internet de los PCs clientes.

Mozilla Firefox

Para que Firefox utilice nuestro Proxy en sus conexiones, debemos ir a Herramientas > Opciones > Avanzado > Red y en el apartado Conexión, hacer clic en el botón Configuración. En la ventana que aparece, debemos configurar la IP y el puerto de nuestro servidor Proxy:



Configuración del Proxy en Firefox

A partir de este momento, Firefox enviará a nuestro Proxy cualquier consulta web que realice, y será nuestro Proxy quien realizará la conexión en caso necesario.

Internet Explorer

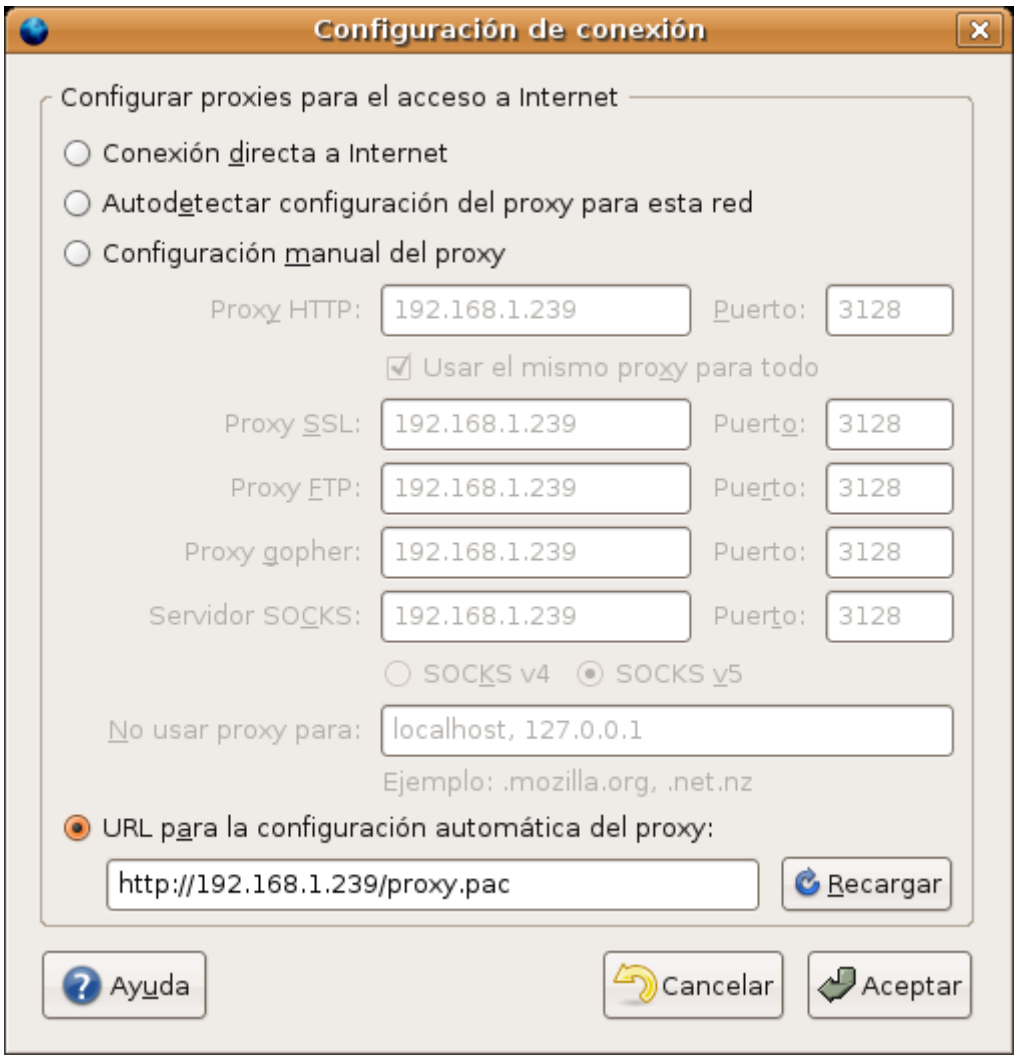
Para indicar a Internet Explorer que debe utilizar un Proxy para realizar conexiones, debemos ir a Herramientas > Opciones de Internet > Conexiones > Configuración de LAN y activar la casilla 'Usar un servidor proxy para la LAN'. En la casilla 'Dirección' pondremos la IP de nuestro Proxy y el 'Puerto' el puerto, tal y como se muestra en la siguiente ventana:

Configuración del Proxy en Internet Explorer

Archivo de configuración automática del proxy

Para no tener que recordar la dirección del proxy y facilitar la tarea a la hora de configurar el proxy en los PCs clientes, existe la posibilidad de crear un archivo de configuración automática del proxy. Dicho archivo indicará al navegador, en función de la url a la que quiera conectarse, si debe hacerlo directamente o debe hacerlo a través del proxy. En un direccionamiento como el que tenemos en nuestro centro, cuando accedemos a nuestra red 10.0.0.0/8 o a la dirección de localhost 127.0.0.1, la conexión debe ser directa, en cambio, cuando accedemos a cualquier otra dirección, deberá ser a través de proxy.

```
//Archivo de configuración automática del proxy
//Archivo /var/www/proxy.pac
function FindProxyForURL(url,host){
        if (isInNet(host, "10.0.0.0", "255.0.0.0"))
                return "DIRECT";
        else if (isInNet(host, "127.0.0.1", "255.255.255.255"))
                return "DIRECT";
        else return "PROXY 192.168.1.239:3128";
}
```



Configuración del Proxy a través de un archivo de configuración

Permitir o denegar el acceso desde ciertos rangos de IPs

Tal y como se ha comentado anteriormente, con squid es sencillo permitir o denegar el acceso a Internet por rangos de IPs. Si tenemos nuestra red diseñada de forma que cada aula utiliza un rango concreto, podremos permitir o denegar el acceso a un aula de forma sencilla.

Para no tener que tocar el archivo squid.conf, lo mejor es crear una acl que cargue las aulas desde un archivo externo. Podemos crear con un editor de texto el archivo /etc/squid/aulas-prohibidas.txt en el que indicaremos los rangos de IPs que no queremos que naveguen. Por ejemplo, si no queremos que naveguen las aulas 2 y 3, el contenido del archivo /etc/squid/denegar-aulas.txt deberá ser:

```
//Archivo /etc/squid/aulas-prohibidas.txt
10.0.2.0/24
10.0.3.0/24
```

Después tendremos que editar squid.conf para crear una acl que cargue los rangos desde el archivo /etc/squid/daulas-prohibidas.txt y deniegue el acceso a dichos rangos.

```
//Archivo externo para indicar las aulas a las que no las permitimos navegar
//Editar squid.conf e introducir estas dos líneas:
acl aulas-prohibidas src "/etc/squid/aulas-prohibidas.txt"
http_access deny aulas-prohibidas
```

Por último, tan solo tenemos que recargar la configuración de squid para que entre en funcionamiento la nueva configuración:

```
//Recargar la configuración de squid
# /etc/init.d/squid reload
```

Igualmente podemos crear una acl para indicar las urls prohibidas desde un archivo externo:

```
//Archivo externo para indicar las urls prohibidas
//Editar squid.conf e introducir estas dos líneas:
acl urls-prohibidas dst "/etc/squid/urls-prohibidas.txt"
http_access deny urls-prohibidas
```

Si no queremos que nuestros alumnos accedan a www.sex.com ni a www.misvecinitas.com, el contenido del archivo /etc/squid/urls-prohibidas.txt debería ser:

```
//Archivo /etc/squid/urls-prohibidas.txt
www.sex.com
www.misvecinitas.com
```

La filosofía sería denegar las aulas prohibidas, denegar las urls prohibidas y luego permitir todo lo demás. Resumiendo, nuestro archivo squid.conf será como el original con las siguientes modificaciones, justo después de la línea # INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS que podríamos traducir como: Inserte sus propias reglas para permitir acceso a sus clientes:

```
//Resumen de modificaciones en squid.conf
# INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS
acl aulas-prohibidas src "/etc/squid/aulas-prohibidas.txt"
http_access deny aulas-prohibidas
acl urls-prohibidas dst "/etc/squid/urls-prohibidas.txt"
http_access deny urls-prohibidas
http_access allow all
```

Así, editando los archivos /etc/squid/aulas-prohibidas.txt y /etc/squid/urls-prohibidas.txt y recargando la configuración de squid ejecutando /etc/init.d/squid reload, podemos reconfigurar squid sin necesidad de tocar el archivo de configuración squid.conf.

El inconveniente es que cada vez que queremos permitir o denegar el acceso a Internet a un aula, tenemos que andar editando el archivo aulas-prohibidas.txt lo que puede resultar un poco engorroso. Podemos crear dos scripts de unix que hagan el trabajo por nosotros y solamente tengamos que ejecutar los scripts indicando el número de aula que queremos prohibir o permitir:

```
Nombre del script: prohibir-aula.sh
```

```
#/bin/bash
#
# Script para prohibir la navegación de un aula
# Se creará el rango del aula en /etc/squid/aulas-prohibidas.txt
# Indicar el número de aula al ejecutar el script

if [ $# -ne 1 ]; then
        echo "Es necesario introducir el numero de aula a prohibir"
        exit -1
fi
echo Prohibir navegar aula $1, subred 10.0.$1.0/24
echo 10.0.$1.0/24 >> /etc/squid/aulas-prohibidas.txt
/etc/init.d/squid reload
echo subredes denegadas:
cat /etc/squid/aulas-prohibidas.txt
//Nombre del script: permitir-aula.sh
#/bin/bash
#
# Script para permitir la navegación de un aula
# Se eliminará el rango del aula de /etc/squid/aulas-prohibidas.txt
# Indicar el número de aula al ejecutar el script

if [ $# -ne 1 ]; then
        echo "Es necesario introducir el numero de aula"
        exit -1
fi
subred=10.0.$1.0/24
echo Permitir navegar aula $1, subred $subred
patron=`echo /10.0.$1.0/d`
cat /etc/squid/aulas-prohibidas.txt | sed -e $patron > /tmp/temp.txt
cat /tmp/temp.txt >  /etc/squid/aulas-prohibidas.txt
/etc/init.d/squid reload
echo Subredes denegadas:
cat /etc/squid/aulas-prohibidas.txt
```

Si deseamos que el aula 1 no navegue, deberíamos ejecutar: prohibir-aula 1. Si luego deseamos permitir que el aula 1 navegue, tendríamos que ejecutar: permitir-aula 1.

Aún con los scripts prohibir-aula.sh y permitir-aula.sh, sigue siendo engorroso realizar cambios ya que el profesor tendría que iniciar sesión en el servidor por ssh y lanzar el script. Lo mejor será crear una página en PHP con botones de comando, donde con un simple clic podamos ejecutar los scripts cómodamente desde el navegador.

Análisis de conexiones

Una de las funcionalidades principales que nos ofrece squid es que registra todos los accesos a Internet. Cada vez que un PCs accede a Internet, squid registrará en el archivo /var/log/squid/access.log la fecha y hora, el PC y la url a la que ha accedido.

```
//Archivo de registro de squid
/var/log/squid/access.log
```

Analizar el archivo /var/log/squid/access.log nos va a resultar de gran ayuda ya que podemos ver a qué páginas web no permitidas se están conectando los alumnos, lo que nos permitirá ir recopilándolas en nuestro archivo urls-prohibidas.txt.