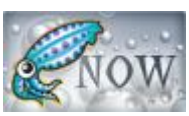
Configuración de Proxy transparente

por Alejandro Moreno
amperisblog[@]gmail.com
http://amperis.blogspot.com

26 de enero de 2008



Introducción

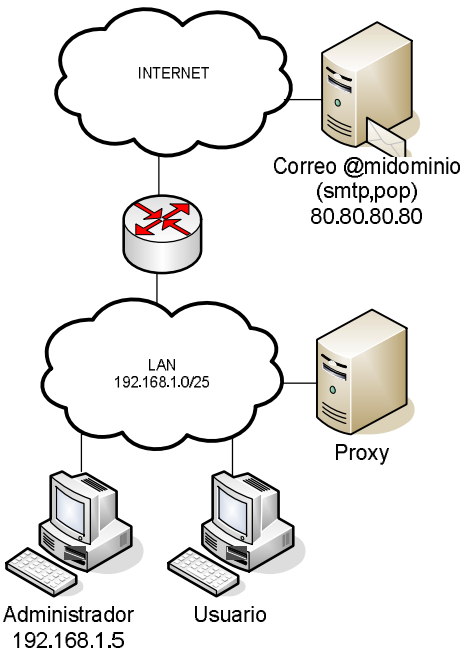
Existen cientos de manuales y howto's de cómo montar un servidor Proxy en producción para una empresa. En este manual se hará hincapié sobre todo en la manera de filtrar el contenido Web a ciertas paginas, el filtrado de las descargas por Internet y la generación de reports diarios para el control.

Como todos sabemos un Proxy es un software que realiza las peticiones a un servidor en nombre de un cliente. Aunque la configuración básica de un Proxy es la de cachear el contenido Web este concepto para mi ya no tiene mucho sentido dado las velocidades que tenemos hoy en dia para el acceso a Internet y sobre todo el aspecto cambiante que tienen las paginas Web. De que sirve cachear una pagina Web si a los 5 minutos ya ha cambiado su aspecto.

En este manual voy a explicar como montar un Proxy transparente para ponerlo en producción para unos 200 usuarios, con filtro a paginas Web, filtro de descarga de archivos, reports diarios, etc.
Al ser un Proxy transparente lo que quiero hacer es que el usuario no tenga conciencia de su existencia y es por eso que el Proxy será su puerta de enlace TCP/IP.

Para implementar esto vamos a utilizar un servidor que será un gateway y allí instalaremos todo el software necesario. Como siempre en mí (y vosotros ¿no?), si se puede hacer gratis lo haremos gratis, así que instalaremos un Linux. Para este servidor utilizaremos la versión 8 de Fedora junto con su última versión para Squid. Dado que tendremos bastantes conexiones este servidor tendrá que tener una Ethernet a 1Gbps.

Todas las estaciones de trabajo tendrán como puerta de enlace el servidor Proxy. Lógicamente aunque el router (o lo que sea) sea alcanzable, los cliente no tienen permisos para cambiar su configuración TCP/IP. Recuerda que estamos hablando de 200 estaciones que seguramente estarán ligadas a un dominio NT o Active Directory.



Instalación

Una vez que tenemos el servidor ya podemos comenzar la instalación del SO. Para la instalación de Fedora 8 no explicaré nada especial salvo elegir el minino numero de paquetes para posteriormente arrancar el mínimo numero de servicios posibles (si que instalaremos las herramientas de desarrollo y las herramientas del sistema). Solo necesitamos lo necesario. Otra cosa importante para cualquier instalación es seleccionar la instalación en inglés ya que es en este idioma donde salen los primeros parches y sobre todo para facilitar la búsqueda de errores por Internet.

Nota: respecto a la BIOS sería aconsejable en los servers activar la opción de arrancar el servidor automáticamente cuando haya tensión eléctrica. De esta forma cuando se va la corriente y vuelve nos aseguramos que el Server arranca. Ya se que todos tenemos un SAI en la empresa. Pero el SAI se termina y no todos tenemos generadores. ☹

Una vez terminada la instalación debemos hacer una actualización completa del sistema y comenzar a bajar los 300Mb que seguramente abran en actualizaciones.

Luego instalaremos todos los paquetes (si no están ya instalados) que necesitamos para nuestra implantación.

```
# yum upgrade
# yum install squid.i386 httpd.i386 mrtg.i386 iptables.i386
```

Después del upgrade reiniciaremos el server (manias mias) para ver que todo ha ido bien y le echaremos un vistazo al /var/log/messages.

También nos faltarán un par de paquetes no oficiales de Fedora (una lastima que no sean ya oficiales). Estos son Sarg y Webmin. Los dos son proyectos libres en Sourceforge. El Webmin es opcional pero lo conocí hace mucho tiempo y para mi viene de serie en todos los servidores Linux. Una vez descargados empezamos a instalar.

```
# cd /root/instalacion
# rpm -i webmin-1.390-1.noarch.rpm
# tar -xzvf sarg-2.2.3.1.tar.gz
# cd sarg-2.2.3.1
# cat README
# ./configure --enable-sysconfdir=/etc/sarg --enable-htmldir=/var/www/html/sarg
# make
# make install
# make clean
```

Nota: Yo tengo por costumbre guardarme en /root/instalacion todos los paquetes que voy instalado. De esta forma si tuviera que montar otra maquina igual en otro sitio simplemente tendría que seguir los mismos pasos y con las mismas versiones de paquetes.

Una vez que tenemos todo el software instalado arrancaremos "`ntsysv`" para deshabilitar y habilitar todos los servicios necesitemos. Tenemos que habilitar iptables, httpd y squid.

Ahora solo queda ir paquete por paquete configurándolo.

Configuración del gateway

Por defecto Linux no realiza funciones de ruteo (forwarding), por tanto no es capaz de aceptar un paquete que no sea para el y rutarlo hacia la mejor ruta. Como nuestro objetivo es que este servidor también haga de gateway tendremos que activar esta opción.
Editaremos "`/etc/sysctl.con`" y cambiaremos "`net.ipv4.ip_forward=0`" por "`net.ipv4.ip_forward=1`". Después solo tenemos que reiniciar los servicios de red con "`service network restart`".

Si configuramos una estación y le ponemos como puerta de enlace este servidor ya deberíamos poder navegar por Internet (si no navegamos tendríamos que parar el iptables ya que aun no lo hemos configurado correctamente).

Configuración de iptables

Lo primero que vamos hacer es asegurar el servidor configurando el firewall. Para ello editaremos la configuración de iptables desde "`/etc/sysconfig/iptables`". La configuración sería algo parecida a esta:

```
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
:LOGDROP - [0:0]

-A LOGDROP -j LOG --log-prefix "IPTABLES "
-A LOGDROP -j DROP

-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT


# Filtro de paquetes entrantes al servidor.
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp -s
192.168.1.5/255.255.255.255 --dport 22    -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp -s
192.168.1.5/255.255.255.255 --dport 3128  -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp -s
192.168.1.5/255.255.255.255 --dport 10000 -j ACCEPT

# Filtramos donde pueden conectarse las estaciones

# podemos hacer Pings a Internet
-A RH-Firewall-1-INPUT -p icmp -m icmp -s 192.168.1.0/255.255.255.0 --icmp-type echo-
request   -j ACCEPT

# Acceso al correo
-A RH-Firewall-1-INPUT -m tcp -p tcp -s 192.168.1.0/255.255.255.0 -d 80.80.80.80  --
dport 25   -j ACCEPT
-A RH-Firewall-1-INPUT -m tcp -p tcp -s 192.168.1.0/255.255.254.0 -d 80.80.80.80  --
dport 110  -j ACCEPT

# Acceso a Web seguro
-A RH-Firewall-1-INPUT -m tcp -p tcp -s 192.168.1.0/255.255.255.0                 --
dport 443  -j ACCEPT

# permitimos hacer consultas DNS
-A RH-Firewall-1-INPUT -p udp -s 192.168.1.0/255.255.255.0                        --
dport 53  -j ACCEPT

# Denegamos el resto de paquetes
-A RH-Firewall-1-INPUT -m tcp -p tcp -j LOGDROP
-A RH-Firewall-1-INPUT -j DROP

# Prohibimos el ICMP Redirect para evitar el cambio de ruta de los paquetes.
-A OUTPUT -p icmp -m icmp --icmp-type redirect -j DROP

COMMIT
*nat
:PREROUTING ACCEPT [2:557]
:POSTROUTING ACCEPT [1:108]
:OUTPUT ACCEPT [1:108]

# Redireccionamiento paral el Transparent-Proxy con Squid
-A PREROUTING -i eth0 -p tcp -m tcp -s 192.168.1.0/255.255.255.0 --dport 80 -j REDIRECT
--to-ports 3128

COMMIT
```

Existen multitud de scripts para generar reglas para iptables. Lo que definimos primero son las reglas necesarias para que solo el administrador del sistema (@ip 192.168.1.5) se pueda conectar por SSH y al Webmin.
El siguiente conjunto de reglas definen donde se pueden conectar las maquinas. Por política de empresa los usuarios solo podrán navegar por Internet (puertos 80 y 443 de TCP) y acceder al correo electrónico (110 y 25 de TCP). Según nuestro caso el servidor de correo electrónico se encuentra en la propia empresa, por tanto los usuarios solo pueden leer el correo de la empresa y no se pueden conectar hacia Internet en busca de de un servidor de correo.

Para terminar denegamos todos los paquetes y marcamos con la etiqueta "IPTABLES" todos los paquetes TCP para identificar mejor los paquetes denegados dentro de "`/var/log/messages`".

Para las reglas de NAT introducimos una regla de prerouting por la cual todos los paquetes que vayan al puerto 80 (http) se redireccionan al pueto 3128 (el Proxy squid).

Nota: Para el que no este acostumbrado al funcionamiento de iptables decir que las ACLs se procesan de arriba abajo. Cuando una regla es cierta para un paquete, este es denegado o aceptado y ya no se procesan más reglas. Si al final el paquete no es aceptado por ninguna regla, la ultima regla elimina el paquete.

Para que iptables cargue las nuevas reglas reiniciamos el servicio con "`service iptables restart`".

Configuración del Proxy

La política que seguiremos para esta implementación será la siguiente: queremos que todos los usuarios puedan navegar (a excepción de los que yo considere), queremos que existen ciertas paginas prohibidas, también queremos que cierto tipo de archivos (exes, pif, etc) estén prohibidos al igual que las conexiones a Messenger.

Antes de configurar Squid vamos a crear 5 archivos de configuración en "`/etc/squid`":

- deny_sites.conf: dominios que estarán prohibidos,
- deny_files.conf: archivos que estarna prohibidos,
- accept_sites.conf: dominios que estarán permitidos y
- deny_srchost: usuarios que no tienen acceso a navegar.

El contenido de deny_sites.conf será:

```
playboy.com
yonkis.com
messenger.msn.com
youtube.com
```

El contenido de deny_files.conf será:

```
\.[eE][xX][eE]$
\.[bB][aA][tT]$
\.[pP][iI][fF]$
\.[rR][eE][gG]$
\.[wW][sS][hH]$
\.[dD][lL][lL]$
\.[cC][aA][bB]$
```

El contenido de accept_sites.conf será:

```
microsoft.com
windowsupdate.com
adobe.com
aeat.es
seg-social.es
```

El contenido de deny_srchost.conf será

```
192.168.1.10
192.168.1.11
```

Ahora ya estamos en condiciones de configurar Squid. Partiremos de la configuración estándar de "`/etc/squid`" y la iremos modificando para ajustará a nuestras necesidades.

Lo primero que haremos es que el Proxy trabaje por su puerto estándar de forma transparente. En versiones anteriores a la 2.6 la forma de ponerlo transparente no era tan fácil como simplemente poner "transparent".

```
http_port 3128 transparent
```

El siguiente paso es definir el conjunto de reglas de control de acceso:

```
acl mired src 192.168.1.0/255.255.255.0
acl deny_sites url_regex "/etc/squid/deny_sites.conf"
acl accept_sites url_regex "/etc/squid/accept_sites.conf"
acl deny_srchosts src "/etc/squid/deny_srchosts.conf"
acl deny_files urlpath_regex "/etc/squid/deny_files.conf"
acl messenger rep_mime_type -i ^application/x-msn-messenger$
```

Una vez definidas las ACLs, las aplico para denegarlas o aceptarlas. Estas conjunto de reglas van después del comentario "`INSERT YOUR OWN RULE(S) HERE TO ALLOW ACCESS FROM YOUR CLIENTS`".

```
http_access allow localhost
http_access allow accept_sites
http_access deny deny_files
http_access deny deny_sites
http_access deny deny_srchosts
http_access deny messenger
http_access allow mired
http_access deny all
```

Nota: dado que la ACL accept_sites está antes de deny_files, si intentásemos bajarnos por ejemplo un archivo ejecutable de windows.com o adobe.com podríamos.

Con esta mínima configuración los usuarios ya podrían comenzar a navegar. Reiniciaremos el squid con "`service squid restart`" y echaremos un vistazo a "`/var/log/squid/access.log`" y "`/var/log/squid/store.log`".

Otra prueba interesante es desde una maquina cliente navegar por http://whatismyipaddress.com/ la cual nos informa de los detalles del Proxy que esta haciendo la petición, señal que nos indica que las peticiones las está realizando un Proxy y no la maquina directamente.

Si un usuario debe descargarse algún programa de alguna pagina confiable pondríamos el dominio de esta pagina dentro de accept_sites.conf y si queremos que alguna maquina no pueda navegar, pondríamos su dirección IP dentro de deny_srchosts.

Opciones de tunning y seguridad

Aunque con la configuración estándar junto con la anterior el Proxy ya estaría funcionando correctamente deberíamos mejorarlo si queremos ponerlo en producción para más de 200 usuarios.

En http://www.visolve.com/squid/squid26/contents.php se encuentran las descripciones de todos los parámetros de Squid para la versión 2.6.

```
cache_mem 128 MB
maximum_object_size 80000 KB
maximum_object_size_in_memory 200 KB
cache_dir ufs /var/spool/squid 3072 32 512
cache_mgr admin@midominio.com
visible_hostname proxy01.midominio.com
httpd_suppress_version_string on
```

Como hemos tocado el tamaño de la cache tendremos que volver a crearla.

```
# service squid stop
# squid -z
# service squid start
```

Configuración de Sarg

El siguiente paso será configurar las estadísticas de Sarg por las cuales podremos ver por donde navega la gente, quien es la persona que más navega y por donde.

La configuración de Sarg se encuentra en "`/etc/sarg/sarg.conf`" (tal como hemos compilado la aplicación anteriormente). La configuración es bastante sencilla. Los parámetros que necesitaremos retocar son:

```
language Spanish
access_log /var/log/squid/access.log
title "MIDOMINIO – Acceso diario a paginas Web"
resolve_ip yes
topsites_num 25
show_sarg_info no
show_sarg_logo no
output_dir /var/www/html/sarg
```

Luego crearemos el siguiente script "`/root/util/sarg-diario.sh`" el cual deberá ejecutarse cada dia. Este script creará el report de navegación del dia anterior y lo dejará en "`/var/www/html/sarg`" según le hemos dicho en el archivo de configuración.

```
#!/bin/bash
YESTERDAY=$(date --date "1 day ago" +%d/%m/%Y)
sarg –n -z -d $YESTERDAY-$YESTERDAY
exit 0
```

Al dia siguiente ya podríamos consultar http://prowy.midomincio.com/sarg para ver los reports generados.

Configuración de MRTG

Para poder crear gráficos estadísticos de Squid tendremos que activar el soporte SNMP de Squid. Para ello editaremos "`/etc/squid/squid.con`" y añadiremos los siguientes parametros:

```
acl snmppublic snmp_community public
snmp_port 3401
snmp_access allow snmppublic all
```

Crearemos el archive de configuración "`/etc/mrtg/mrtg_squid.cfg`" con el siguiente contenido:

```
HtmlDir: /var/www/html/mrtg-squid
ImageDir: /var/www/html/mrtg-squid
LogDir: /var/lib/mrtg
ThreshDir: /var/lib/mrtg
LoadMIBS: /etc/squid/mib.txt

Target[proxy-hit]: cacheHttpHits&cacheServerRequests:public@proxy.midominio.com:3401
RouterName[proxy-hit]: cacheUniqName
MaxBytes[proxy-hit]: 100000
Title[proxy-hit]: HTTP Hits
PageTop[proxy-hit]: <H2>proxy Cache Statistics: HTTP Hits/Requests</H2>
Suppress[proxy-hit]: y
LegendI[proxy-hit]:  HTTP hits
LegendO[proxy-hit]:  HTTP requests
Legend1[proxy-hit]:  HTTP hits
Legend2[proxy-hit]:  HTTP requests
YLegend[proxy-hit]: perminute
ShortLegend[proxy-hit]: req/min
Options[proxy-hit]: nopercent, perminute, dorelpercent, unknaszero

Target[proxy-srvkbinout]:
cacheServerInKb&cacheServerOutKb:public@proxy.midominio.com:3401
RouterName[proxy-srvkbinout]: cacheUniqName
MaxBytes[proxy-srvkbinout]: 76800
Title[proxy-srvkbinout]: Cache Server Traffic In/Out
PageTop[proxy-srvkbinout]: <H2>Cache Statistics: Server traffic volume (In/Out)</H2>
Suppress[proxy-srvkbinout]: y
LegendI[proxy-srvkbinout]:  Traffic In
LegendO[proxy-srvkbinout]:  Traffic Out
Legend1[proxy-srvkbinout]:  Traffic In
Legend2[proxy-srvkbinout]:  Traffic Out
YLegend[proxy-srvkbinout]: per minute
ShortLegend[proxy-srvkbinout]: b/min
kMG[proxy-srvkbinout]: k,M,G,T
kilo[proxy-srvkbinout]: 1024
Options[proxy-srvkbinout]: nopercent, perminute, unknaszero
```

Para que MRTG sea capaz de generar los gráficos tiene que saber exactamente como esta dispuesta su información, para ello tiene que leer la base información de administración (MIB) con el parámetro LoadMIBS.

Luego crearemos una tarea programada cada cinco minutos para ejecutar el siguiente comando:

```
LANG=C LC_ALL=C /usr/bin/mrtg /etc/mrtg/mrtg_squid.cfg --lock-file
/var/lock/mrtg/mrtg_squid --confcache-file /var/lib/mrtg/mrtg_squid.ok
```