

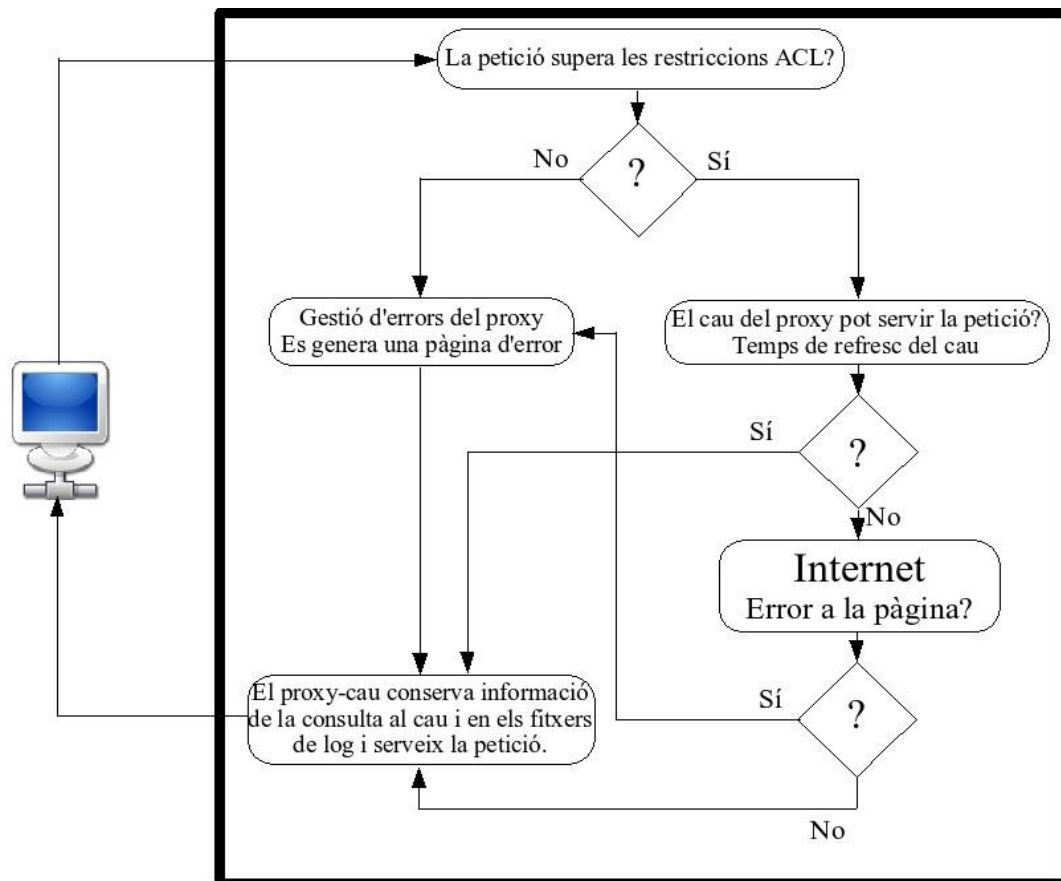
Capítol 6

El Servei PROXY
Configuració del servei
Explotació del servei a l'empresa

6.1 Servidor proxy-cau SQUID

El servei proxy és utilitzat actualment arreu com accelerador del servei de connexió a Internet i navegació web de cara a l'usuari final. Grans distribuïdors de servies d'Internet utilitzen un tipus o altre de proxy (normalment de manera transparent per l'usuari) per evitar col·lapses en el seu servei i aprofitar l'ample de banda que s'ofereix als usuaris.

El funcionament d'un proxy-cau és realment senzill. El següent gràfic ens pot ajudar a entendre la dinàmica del seu funcionament i el perquè ens permet reduir l'ample de banda:



Tot comença amb una petició de connexió des d'un client utilitzant un navegador d'Internet. Aquest client no envia directament la seva petició a Internet sinó que ho fa a través d'un proxy. El proxy té configurades una sèrie de restriccions ACL (Access Control List) que permeten o deneguen la petició del client. Si el client i la seva petició superen les restriccions, el proxy revisa si ja existeix la petició del client al cau i si aquesta no ha superat el temps de refresc del cau la serveix directament al client sense fer cap connexió cap a l'exterior. Si la petició existeix al cau però el temps de refresc del cau ha estat superat, o no existeix cap instància de la petició del client al cau, es fa una consulta a Internet per resoldre la petició del client. Si la consulta està ben resolta el proxy guarda una instància (o renova la ja existent). En canvi, si hi ha un error, perquè no existeix la pàgina o no supera les restriccions ACL, es passa a la gestió d'errors on es genera una pàgina web d'error que es servida al client que ha fet la petició, guardant al cau una instància d'aquesta petició.

En tots els casos hi ha un estricte seguiment de missatges de part del proxy en el que respecta a les peticions dels clients, la informació dels clients (IPs, usuaris, navegadors usats, hora de la petició i altres) i la resolució de les peticions per part del proxy. Tot es guarda en uns fitxers (o logs) dins del sistema i existeixen eines per poder digerir el seu contingut i treure interessants conclusions com veurem més tard.

Està clar que en fer un número de peticions a diverses pàgines web (que es repeteixen molt) per part d'un nombre elevat de clients el cau pot arribar a reduir el tràfic cap a Internet en més d'un 75%, ja que el proxy les serveix directament. Des del punt de vista del client, la velocitat de connexió sembla que augmenti força, ja que les pàgines les serveix directament un “servidor” que en aquest cas està en la mateixa xarxa interna. Passem de 2/4 Mbips, d'una ADSL, a 100 Mbips de velocitat d'una xarxa LAN.

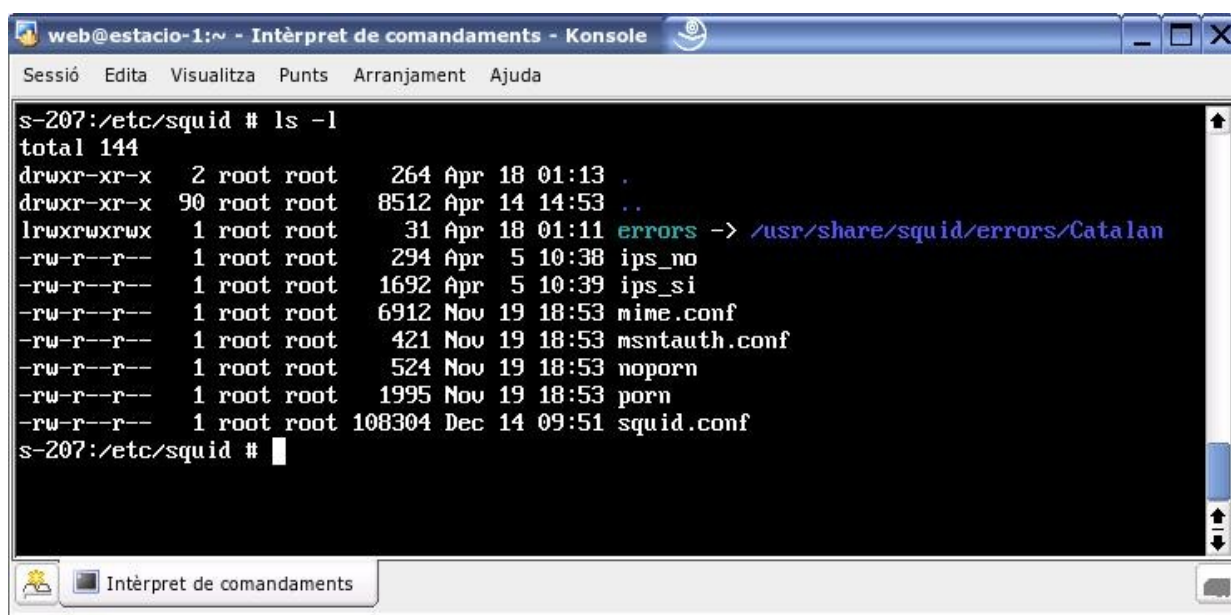
Aquest augment de la velocitat (que no és real) és una de les característiques principals d'un proxy-cau, però en té altres que ens poden ser d'utilitat:

- Pot fer de proxy-cau de les connexions HTTP, però també de FTP i altres URLs.
- Suporta proxy per connexions SSL.
- Permet definir jerarquies de proxy-cau: proxy parents.
- Permet extreure informació “útil” sobre el tràfic de la nostra xarxa.
- Permet definir un proxy de tipus “transparent” per l'usuari final.
- WCCP (Cisco's Web Cache Coordination Protocol).
- Permet un control exhaustiu de les connexions a Internet per mig d'ACLs.
- Permet acceleració sobre el servei HTTP.
- Permet SNMP (Simple Network Management Control) per monitoritzar el servei i el nostre servidor: CPU, memòria, disc dur,
- Permet proxy-cau sobre peticions al DNS.

Per totes aquestes funcionalitats SQUID es presenta com un proxy-cau que pot cobrir moltes de les necessitats d'una petita o mitjana empresa. Actualment SQUID es troba en la versió estable 2.5 i es pot baixar directament des de la web oficial [1]. Com es comenta en aquest enllaç, Squid és un aplicació proxy-cau de funcionalitats complertes dissenyada per executar-se en sistemes de tipus UNIX i que és bàsicament suportada per voluntaris amb finalitats d'aconseguir un producte lliure i de codi obert.

6.2 Configuració del servei proxy-cau SQUID

Els fitxers de configuració de SQUID es poden trobar en la majoria de distribucions sota el directori */etc/squid*. A la següent imatge es pot veure un llistat del contingut del directori:



```

web@estacio-1:~ - Intèrpret de comandaments - Konsole
Sessió  Edita  Visualitza  Punts  Arranjament  Ajuda

s-207:/etc/squid # ls -l
total 144
drwxr-xr-x  2 root root   264 Apr 18 01:13 .
drwxr-xr-x 90 root root  8512 Apr 14 14:53 ..
lrwxrwxrwx  1 root root    31 Apr 18 01:11 errors -> /usr/share/squid/errors/Catalan
-rw-r--r--  1 root root   294 Apr  5 10:38 ips_no
-rw-r--r--  1 root root  1692 Apr  5 10:39 ips_si
-rw-r--r--  1 root root  6912 Nov 19 18:53 mime.conf
-rw-r--r--  1 root root   421 Nov 19 18:53 msntauth.conf
-rw-r--r--  1 root root   524 Nov 19 18:53 noporn
-rw-r--r--  1 root root  1995 Nov 19 18:53 porn
-rw-r--r--  1 root root 108304 Dec 14 09:51 squid.conf
s-207:/etc/squid #

```

Per configurar SQUID podem utilitzar, una vegada més, totes les eines que s'han anat comentant durant el curs: WEBMIN, YAST per els usuaris de SuSE i la línia de comandes.

Com ens passava en el cas del servidor Apache, el servidor proxy està ja pràcticament configurat per ser usat una vegada s'instal·la i ens calen únicament petites modificacions per adaptar el seu funcionament al nostre gust. Les eines com ara WEBMIN, en aquest cas, són més útils per fer un seguiment del servei que no pas per configurar les opcions bàsiques.

Una vegada més optarem per configurar el servei directament des de la línia de comandes. Les opcions que ens presenta el fitxer de configuració (*squid.conf*) són moltes però estan prou explicades dins del mateix fitxer (i als manuals d'SQUID). No faré una descripció detallada de totes però sí de les que ens permeten definir les opcions bàsiques del servei. Abans de començar l'explicació de la posta en marxa del servei és obligat comentar les necessitats de la nostra màquina per suportar el servei SQUID.

En aquest cas, el servei de proxy implica (en xarxes de grans dimensions i amb tràfic important) un moviment “important” de fitxers des del disc dur cap a la targeta de xarxa. Caldrà que tots dos siguin ràpids. Part del cau es sol conservar en memòria, per tant és interessant tenir una bona quantitat de memòria si es vol accelerar al màxim les transferències des del cau als clients. La creació, indexació i manteniment del cau impliquen una activitat important de la CPU cada cert temps, però no cal una potència de càlcul massa important si no estem tractant amb xarxes molt grans o amb configuracions molt específiques (que surten de l'objectiu del curs).

Mirem ara quines són les opcions importants que cal canviar (o interessa conèixer) per ficar en marxa el servei (dins del fitxer *squid.conf*):

<i>Opció</i>	<i>Perquè serveix</i>
http_port (Defecte 3128)	Permet definir el port on espera les connexions el servidor proxy
cache_mem 8M (Defecte)	Permet definir la quantitat de memòria RAM per el cau
cache_dir	Permet definir la quantitat de disc, la situació, el tipus de sistema de fitxers i altres característiques del cau. Les opcions s'expliquen a continuació
cache_log	Permet definir el fitxer de sortida de missatges del servei (logs)
acl/http_access	Permeten definir, conjuntament, les restriccions ACL

Sense cap dubte, la opció més important a modificar sigui *cache_dir*. Com ja s'ha comentat ens permet definir les característiques del cau. Per defecte té les següents opcions:

```
cache_dir ufs /var/cache/squid 100 16 256
```

Que són prou raonables per la majoria d'instal·lacions del servidor proxy. Ens podem trobar el cas, però, de voler utilitzar una zona de disc més ràpida per la situació del cau i millorar així el seu rendiment. També podem augmentar la mida del cau o el tipus de sistema de fitxers a utilitzar. Les opcions possibles són les següents:

```
cache_dir tipus Nom_Directori Mida_MB L1 L2 [opcions] [Q1=n] [Q2=n]
```

Podem utilitzar més d'una instància *cache_dir* per definir més d'un cau.

El significat de cadascuna d'aquestes opcions és:

<i>Opció</i>	<i>Significat</i>
tipus	Tipus de sistema d'emmagatzemar en el cau. <i>ufs</i> és l'estàndard d'SQUID, però podem utilitzar <i>aufs</i> o <i>diskd</i>
Nom_Directori	Directori complet on volem crear el cau
Mida_MB	La mida en MegaBytes del nostre cau (100 MB per defecte)
L1	Nombre de directoris de 1r nivell sota el cau (16 per defecte)
L2	Nombre de directoris de 2n nivell sota cada directori de 1r nivell (256 per defecte)
[opcions]	opcions diverses com ara <i>read-only</i> o <i>max-size=n</i>
[Q1=n]	Nombre màxim de demandes a partir de les quals SQUID no obrirà més fitxers. Per defecte 64.
[Q2=n]	Nombre màxim de missatges a partir de les quals SQUID bloquejarà l'emissió de missatges. Per defecte 72.

Una mida adequada per un proxy-cau pot ser al voltant d'1 GB. Si volem utilitzar tot el disc per cau, caldrà descomptar un 20% de la seva capacitat lliure per altres menesters que necessita el mateix proxy.

L'optimització d'un proxy-cau és una feina “delicada” i que es deriva en gran part de l'observació constant del seu comportament. Cal fer un seguiment durant un temps per poder optimitzar els diferents paràmetres. Com que no volem fer una configuració tancada, deixarem les opcions per defecte tal com venen amb el servei instal·lat i ficarem els proxy-cau en marxa. Abans de continuar cal donar una ullada al directori on es crearà el cau i veure el propietari i els permisos:

```

web@estacio-1:~ - Intèrpret de comandaments - Konsole
Sessió Edita Visualitza Punts Arranjament Ajuda

s-207:/var/cache/squid # ls -l
total 757
drwxr-x--- 18 squid root      504 Apr 15 04:15 .
drwxr-xr-x 10 root  root      240 Nov 19 18:23 ..
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 00
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 01
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 02
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 03
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 04
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 05
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 06
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 07
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 08
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 09
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0A
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0B
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0C
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0D
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0E
drwxr-x--- 258 squid nogroup  6192 Nov 19 19:20 0F
-rw-r--r-- 1 squid nogroup 675360 Apr 15 14:30 swap.state
-rw-r--r-- 1 squid nogroup 0 Apr 15 04:15 swap.state.last-clean
s-207:/var/cache/squid #

```


El propietari del cau és l'usuari *squid* i el *group* per defecte *nogroup*. Únicament *squid* i té permisos totals sobre el cau.

Per ficar en marxa el servidor proxy cal anar al directori `/etc/rc.d` i executar el shell-script corresponent. Les opcions del shell script són les següents:

start stop status try-restart restart force-reload reload probe

Molt similars a les que s'han tractat en altres servidors. La primera vegada que *squid* es fica en marxa crearà el cau amb la definició que hem fet al fitxer de configuració, això pot consumir un temps. Per veure que el proxy estè actiu cal fer una prova senzilla. Amb qualsevol navegador, ja sigui del nostre servidor linux o de qualsevol client de la xarxa, cal definir en les opcions de navegació que utilitzem un servidor proxy, amb la IP corresponent al nostre servidor i el port 3128 que és el que s'ha deixat per defecte. Si la navegació per Internet és normal observarem els següents efectes: la primera vegada que visitem una pàgina (per exemple, <http://www.enxarxa.net>) la navegació es veurà una mica alentida. Qualsevol visita posterior que fem a una pàgina visitada (encara que es buidi el cau local) serà molt més ràpida.

6.3 Adaptació del proxy a necessitats particulars

Ara plantejaré un cas pràctic on el desplegament d'un proxy-cau pren tot el seu sentit. Imaginem un centre d'ensenyament amb un centenar d'ordinadors on els responsables d'informàtica volen millorar la connexió a Internet i controlar l'accés des de les aules a Internet per mig de les IPs dels clients, filtrant al mateix temps els continguts de les pàgines on es connecten els alumnes. Disposen d'un servidor amb dues targetes de xarxa i disc suficient per crear un cau. La primera targeta de xarxa apunta al tram d'IPs de les aules (192.168.1.0/24) i la segona apunta al tram d'IP del router (192.168.0.0/24) que és el que dona la sortida a Internet. Per motius de seguretat, les xarxes dedicades a l'alumnat i a la resta del centre estan física separades i l'únic nexe d'unió entre les diferents xarxes és el servidor. Aquest esquema ja ha sortit anteriorment al curs al capítol 3.

Per evitar tenir que explicar la configuració del proxy als alumnes (i no donar pistes de com funciona la connexió internament al centre) volen que la connexió sigui transparent de cara als usuaris: es a dir que els alumnes únicament es preocupen de navegar per Internet sense saber com. Els responsables d'informàtica han detectat que hi ha algunes assignatures que necessiten connectar a pàgines webs específiques (com ara cursos Cisco, edu365, connexions sftp i altres) les quals necessiten connexions addicionals al port 80, que és el que gestiona (normalment) el proxy. Aquests ports poden ser diversos; no volen haver d'anar modificant la configuració conforme es necessiti un nou port d'accés, pel que demanen que la connexió als ports diferents al port 80 es faci de manera directa o no filtrada a excepció de certs ports particulars com ara els corresponents a *e-mule* i altres programes P2P o servei de IRC i Xat.

Per poder donar el servei que es demana caldrà fer les següents actuacions:

- ✓ Crear un servei proxy-cau amb les característiques següents:
 - × Proxy-cau amb control ACL per IP de clients.
 - × Proxy-cau amb control ACL per paraules clau.
 - × Proxy-cau transparent.
- ✓ Crear regles IPTABLES per satisfer dues necessitats:
 - × Poder crear un proxy-cau transparent.
 - × Emmascarar la xarxa interna i fer que el servidor actuï com un encaminador on es filtraran certs paquets TCP/IP i UDP que estiguin encaminats cap a ports que volem filtrar.

EL fet de crear un proxy transparent vers un proxy manual té les seves avantatges i inconvenients. En primer lloc té la gran virtut de fer la configuració dels clients molts senzilla i sense intervenció directa per part de ningú perquè s'utilitzi el proxy (els clients utilitzaran el proxy “per força”, sense cap altra alternativa). La seguretat i el control sobre les connexions dels clients queda totalment controlada des del servidor. Aquestes virtuts són el seu principal inconvenient, ja que qualsevol mal funcionament del proxy implica que els clients quedin sense connexió a Internet.

Si les regles de filtratge impedeixen que es pugui entrar a una pàgina concreta (encara que no sigui adequat el filtratge en aquest cas) els clients no en tindran accés de cap forma fins que l'administrador del proxy faci les modificacions corresponents. En canvi, si el proxy es pot configurar manualment des dels clients (directament al navegador) sempre es poden tenir les virtuts del proxy evitant el seus inconvenients respecte a les restriccions de connexió. Queda clar que la utilització d'un proxy transparent és molt més “estricta i restrictiva” que la utilització d'una configuració manual.

Mirem ara com podem crear l'escenari que s'ha descrit abans amb el nostre proxy i les regles IPTABLES corresponents. Comencem per configurar SQUID com a proxy transparent, afegirem les següents línies sota l'entrada corresponent del fitxer *squid.conf*:

```
httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_single_host off
httpd_accel_with_proxy on
httpd_accel_uses_host_header on
```

Per aprofitar al màxim la funcionalitat de cau aplicarem la següent directiva, que per defecte està en desactivada:

```
offline_mode on
```

I ens assegurem que ningú (dels nostres usuaris) sàpiga que utilitzem un proxy aplicant les directives següents:

```
header_access X-Forwarded-For deny all
header_access Via deny all
```

Respecte a la directiva sobre la ubicació i mida del cau farem aquesta petita modificació respecte l'estàndard, ja que tenim una zona de disc prou gran (de més d'1 GB) que està separada de la resta del sistema de fitxers (és una partició independent i està muntada sobre el directori cau-squid):

```
cache_dir aufs /cau-squid1000 16 256
```

S'ha utilitzat “*aufs*” que utilitza POSIX-threads i que permet una optimització important dels processos d'entrada i sortida que milloren l'acceleració del proxy.

Com que els centres d'ensenyament estan dins d'una organització més gran que disposa d'un proxy propi ens interessa que la configuració del nostre proxy consulti primer al proxy “superior” per si ens pot servir la demanda dels nostres clients, per fer-ho fem el proxy superior com a parent amb aquesta directiva:

```
cache_peer isis.xtec.es parent 8080 3130 no-query
```

Amb tot això ja tenim el nostre proxy configurat per actuar com a transparent i adaptat a les nostres característiques més o menys ideals. Ara ens cal definir les dues restriccions ACL comentades. La

primera restricció és per adreça IP. El que farem serà primer definir una xarxa per servir el proxy i que qualsevol petició d'altra xarxa sigui desatesa, això ho fem aplicant aquesta ACL:

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

Com es pot veure, l'estructura de les ACL és senzilla, segueix més o menys el següent esquema:

```
acl nom_de_la_regla mètodo actuació expressió coincident
acció allow ó deny nom_ACL
```

El *nom_de_la_regla* pot ser qualsevol que dessitjem i la expressió coincident pot ser una IP, paraules, una xarxa o un fitxer on hi hagi qualsevol paràmetre significatiu, més tard veurem un exemple. Mirem quines són les opcions més importants i el seu significat per a *mètode_actuació*:

<i>Paraula clau</i>	<i>Utilització</i>
<i>src</i>	Direcció d'origen de la petició. Cal que sigui una adreça IP o un conjunt d'adreces IP mai el nom d'un ordinador.
<i>dst</i>	Direcció de destí de la petició rebuda. Com al cas anterior serà una adreça IP.
<i>myip</i>	Adreça IP única de l'ordinador que fa la petició.
<i>srcdomain</i>	Nom del host (no domini) origen que fa la petició.
<i>dstdomain</i>	Nom del host (no domini) destí de la petició que ha rebut el proxy.
<i>srcdom_regex</i>	Avalua el nom del domini d'origen per trobar coincidències
<i>dstdom_regex</i>	Avalua el nom del domini de destí per trobar coincidències
<i>time</i>	Utilitza el temps per avaluar la condició
<i>url_regex</i>	S'utilitza tota l'adreça escrita pel client a la petició per buscar coincidències
<i>urlpath_regex</i>	Idèntica a l'anterior però utilitzant l'adreça a partir del primer /
<i>method</i>	Avalua el mètode utilitzat per la petició per obtenir informació del servidor: GET, POST, CONNECT i PUT i l'utilitza per les coincidències
<i>maxconn</i>	Nombre màxim de connexions per un client
<i>rep_mime_type</i>	Avalua el tipus de dades que es transfereixen a la petició en format <i>mime: application/x-javascript ...</i> tal com es mostra al fitxer <i>/etc/mime.types</i>

Algunes de les accions més importants poden ser les següents:

<i>Paula clau</i>	<i>Significat</i>
<i>http_access</i>	Controla l'accés al proxy de la petició que ha superat la comparativa de l'ACL
<i>no_cache</i>	Serveix per forçar o denegar l'entrada al cau d'una petició concreta
<i>always_direct</i>	Serveix per enviar la petició directament sense passar per proxy (parent)
<i>never_direct</i>	Serveix per forçar que la petició passi pel proxy obligatòriament

Ara que ja coneixem la dinàmica de les llistes de restricció anem a veure com apliquem algunes al nostre cas concret. Recordem que volem limitar l'accés a Internet dels clients segons la seva IP. De vegades voldrem que algunes IPs accedeixin i altres vegades que no accedeixin. Ens caldrà definir dos fitxers amb les IPs permeses i altre amb les IPs restringides i utilitzar aquests fitxers perquè la llista ACL corresponent pugui comparar i decidir si es denega o es dona accés. Ambdues llistes cal que siguin excloents. Aquests fitxers s'anomenen *ips_si* i *ips_no* i estan al mateix directori que el fitxer de configuració d'squid, com es pot veure al llistat que s'ha presentat abans. El primer fitxer conté una llista de les adreces que tenen permès l'accés al proxy i l'altra una llista d'adreces que tindran prohibit l'accés a Internet (via web).

L'altra limitació correspon a una restricció que impedeixi que els usuaris puguin accedir a pàgines web que continguin (en el seu nom mateix) alguna paraula ofensiva, i que podem utilitzar per limitar l'accés a webs de contingut violent o pornogràfic. Per fer-ho definim dos fitxers *porn* i *noporn*, que contenen el primer paraules restringides i el segon URLs permeses. Aquesta restricció ACL (molt dura per cert), impedirà l'entrada a qualsevol adreça que contingui una paraula prohibida. Si per exemple utilitzem com a paraula prohibida *sex* la restricció impedirà entrar a qualsevol URL que contingui aquesta paraula. Així ens impedirà “buscar” al *google* adreces webs d'educació sexual ... i evidentment no és el que volem. Per evitar això ens caldrà veure quines webs que continguin la paraula *sex* han d'estar permeses i incloureu dins del fitxer *no_porn*.

Per escriure aquestes restriccions al fitxer de configuració ens cal ficar el següent:

```
acl ips_si src "/etc/squid/ips_si"
acl ips_no src "/etc/squid/ips_no"
acl porn url_regex "/etc/squid/porn"
acl noporn url_regex "/etc/squid/noporn"
http_access allow ips_si
http_access deny ips_no
http_access allow noporn
http_access deny porn
```

Com que ja estem ficats en el tema, ara introduïrem una regla que ens permetrà evitar que el cau s'ompli amb fitxers que els usuaris es poden baixar de la xarxa (fins i tot podríem impedir que els baixin, però de moment l'únic que farem serà prevenir que no s'ompli el cau). Imaginem que hi ha usuaris que baixen fitxers amb mp3, iso o ISO, gzip, zip, tar.gz, gz i altres terminacions que no volem que es guardin al cau. També volem evitar al mateix temps que qualsevol consulta a una

pàgina dinàmica es guardi al cau. Aplicarem la següent regla:

```
acl NOCACHE urlpath_regex -i cgi-bin \? \.gz$ \.bz2$ \.zip$ \.iso$ \.tar$ \.mp3$
no_cache deny NOCACHE
```

L'aplicació de regles ACL pot arribar a ser molt complexa. Sabent el funcionament ens permet un control pràcticament total de les connexions. Però encara així, moltes vegades és difícil dur un control el prou significatiu com per evitar que els usuaris puguin fer i desfer segons quines coses, degut a l'elevat nombre de pàgines web que són susceptibles de ser visitades. Això ens porta a tenir que controlar les connexions de la nostra xarxa analitzant les sortides del programa. Per fer-ho tenim eines molt interessants que ens faciliten la feina com veurem més tard.

Amb aquestes modificacions ja tenim SQUID configurat per actuar com un proxy transparent. Ens queda però definir el nostre servidor com l'encaminador de la xarxa interna aplicant els filtres corresponents. Per fer-ho aplicarem regles d'IPTABLES.

Al capítol següent definirem amb més deteniment què són i com funcionen aquestes regles. De moment ens preocuparem de veure com les podem aplicar de manera senzilla per tenir un proxy transparent totalment funcional. Crearem un fitxer per introduir les nostres regles IPTABLES al directori */etc/rc.d* (recordem que aquest directori conté els shell-script d'engegada dels serveis). Això ho fem perquè aplicarem aquestes regles en engegar el sistema. El fitxer que creem l'anomenem, per exemple, *iptables_rules*. Perquè el nostre proxy actuï com a proxy transparent únicament ens cal aplicar les següents regles (que escriurem al fitxer *iptables_rules*):

```
#!/bin/sh
## Esborrar regles anteriors
iptables -F
iptables -X
iptables -Z
iptables -t nat -F
## Política per defecte: ACCEPTAR TOT
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
iptables -P FORWARD ACCEPT
iptables -t nat -P PREROUTING ACCEPT
iptables -t nat -P POSTROUTING ACCEPT
## Al localhost tot
iptables -A INPUT -i lo -j ACCEPT
## A la xarxa local tot
iptables -A INPUT -s 192.168.1.0/24 -i eth1 -j ACCEPT
# Enmascarament de la xarxa local
iptables -t nat -A POSTROUTING -s 192.168.1.0/24 -o eth0 -j MASQUERADE
## Activem l'enrutament per la nostra LAN
echo 1 > /proc/sys/net/ipv4/ip_forward
## Reenviem tràfic del port 80 des de la xarxa LAN cap al port 3128 SQUID
iptables -t nat -A PREROUTING -i eth1 -s 192.168.1.0/24 -d ! 192.168.1.0/24 -p tcp --dport 80 -j
REDIRECT --to-port 3128
## S'acabat
```

Les regles anteriors permeten al nostre servidor actuarà d'encaminador per “tot” el tràfic de la nostra xarxa local cap a Internet i en concret reenvia el tràfic del port 80 (les connexions web) cap al proxy SQUID. Aquest esquema és insegur i no tractem de filtrar en cap moment cap port “ben conegut” de la nostra xarxa interna, per la qual cosa poden realitzar-se connexions poc desitjades. L'única mesura aplicada és el filtratge ACL del mateix SQUID.

Al capítol següent veurem amb més detall com configurar el tallafocs per augmentar els nivells de seguretat i evitar connexions “descontrolades” des de la nostra LAN cap a Internet i des de Internet a la nostra LAN.

6.4 Revisió i “digestió” dels logs d'SQUID (i APACHE)

La generació d'informes legibles i dels que es puguin treure profit a a partir dels logs derivats dels serveis és una eina inestimable per un administrador de sistemes. En el cas que ens tracta, el servidor proxy, la seva utilitat radica en que ens permetrà veure si el funcionament del nostre proxy és l'adequat i així poder optimitzar les seves opcions. A més a més, és una eina útil com a control per poder justificar usos fraudulents de les instal·lacions (maquinari i recursos) de la empresa per part de treballadors o altres possibles potencials usuaris. Si parlem del servei Apache, la seva utilitat es reflecteix en poder veure quines visites (qui, des de on i en quin moment) té el nostre servidor web (i les pàgines que són visitades).

Comencem mirant quina informació guarda el servidor SQUID. El proxy utilitza una zona del disc situada en `/var/log/squid` per guardar informació diversa. Aquí es pot veure un llistat del contingut d'aquest directori:

The screenshot shows a terminal window with the title bar "ocastell@lluna:~ - Intèrpret de comandaments - Ksilo". The terminal content displays the command `s-207:/var/log/squid # ls` and its output, which lists numerous log files in a grid-like format. The files are organized into columns, with some files in the first column being truncated by the terminal width. The files listed include:

- `access.log-20050216.gz`
- `access.log-20050217.gz`
- `access.log-20050218.gz`
- `access.log-20050219.gz`
- `access.log-20041124.gz`
- `access.log-20041126.gz`
- `access.log-20041127.gz`
- `access.log-20041130.gz`
- `access.log-20041202.gz`
- `access.log-20041203.gz`
- `access.log-20041204.gz`
- `access.log-20041211.gz`
- `access.log-20041215.gz`
- `access.log-20041216.gz`
- `access.log-20041217.gz`
- `access.log-20041221.gz`
- `access.log-20041222.gz`
- `access.log-20050111.gz`
- `access.log-20050113.gz`
- `access.log-20050115.gz`
- `access.log-20050118.gz`
- `access.log-20050119.gz`
- `access.log-20050120.gz`
- `access.log-20050122.gz`
- `access.log-20050126.gz`
- `access.log-20050128.gz`
- `access.log-20050201.gz`
- `access.log-20050202.gz`
- `access.log-20050204.gz`
- `access.log-20050205.gz`
- `access.log-20050209.gz`
- `access.log-20050210.gz`
- `access.log-20050211.gz`
- `access.log-20050215.gz`
- `access.log-20050216.gz`
- `access.log-20050217.gz`
- `access.log-20050218.gz`
- `access.log-20050219.gz`
- `access.log-20050222.gz`
- `access.log-20050223.gz`
- `access.log-20050224.gz`
- `access.log-20050225.gz`
- `access.log-20050226.gz`
- `access.log-20050301.gz`
- `access.log-20050302.gz`
- `access.log-20050304.gz`
- `access.log-20050305.gz`
- `access.log-20050308.gz`
- `access.log-20050309.gz`
- `access.log-20050310.gz`
- `access.log-20050311.gz`
- `access.log-20050314.gz`
- `access.log-20050315.gz`
- `access.log-20050316.gz`
- `access.log-20050317.gz`
- `access.log-20050318.gz`
- `access.log-20050319.gz`
- `access.log-20050330.gz`
- `access.log-20050331.gz`
- `access.log-20050402.gz`
- `access.log-20050405.gz`
- `access.log-20050406.gz`
- `access.log-20050407.gz`
- `access.log-20050408.gz`
- `access.log-20050409.gz`
- `access.log-20050412.gz`
- `access.log-20050413.gz`
- `access.log-20050414.gz`
- `access.log-20050415.gz`
- `access.log-20050419.gz`
- `cache.log`
- `ctry_usage_200411.png`
- `ctry_usage_200412.png`
- `ctry_usage_200501.png`
- `ctry_usage_200502.png`
- `daily_usage_200411.png`
- `daily_usage_200412.png`
- `daily_usage_200501.png`
- `daily_usage_200502.png`
- `hourly_usage_200411.png`
- `hourly_usage_200412.png`
- `hourly_usage_200501.png`
- `hourly_usage_200502.png`
- `index.html`
- `rcsquid.log`
- `store.log`
- `store.log-20041124.gz`
- `store.log-20041126.gz`
- `store.log-20041127.gz`
- `store.log-20041130.gz`
- `store.log-20041202.gz`
- `store.log-20041203.gz`
- `store.log-20041204.gz`
- `store.log-20041211.gz`
- `store.log-20041214.gz`
- `store.log-20041215.gz`
- `store.log-20041216.gz`
- `store.log-20041217.gz`
- `store.log-20041221.gz`
- `store.log-20041222.gz`
- `store.log-20050411.gz`
- `store.log-20050412.gz`
- `store.log-20050413.gz`
- `store.log-20050414.gz`
- `store.log-20050415.gz`
- `store.log-20050419.gz`
- `store.log-20050113.gz`
- `store.log-20050115.gz`
- `store.log-20050118.gz`
- `store.log-20050119.gz`
- `store.log-20050120.gz`
- `store.log-20050121.gz`
- `store.log-20050122.gz`
- `store.log-20050125.gz`
- `store.log-20050126.gz`
- `store.log-20050128.gz`
- `store.log-20050201.gz`
- `store.log-20050202.gz`
- `store.log-20050203.gz`
- `store.log-20050204.gz`
- `store.log-20050205.gz`
- `store.log-20050209.gz`
- `store.log-20050210.gz`
- `store.log-20050211.gz`
- `store.log-20050215.gz`
- `store.log-20050216.gz`
- `store.log-20050217.gz`
- `store.log-20050218.gz`
- `store.log-20050219.gz`
- `store.log-20050222.gz`
- `store.log-20050223.gz`
- `store.log-20050224.gz`
- `store.log-20050225.gz`
- `store.log-20050226.gz`
- `store.log-20050301.gz`
- `store.log-20050302.gz`
- `store.log-20050303.gz`
- `store.log-20050308.gz`
- `store.log-20050309.gz`
- `store.log-20050310.gz`
- `store.log-20050311.gz`
- `store.log-20050314.gz`
- `store.log-20050315.gz`
- `store.log-20050316.gz`
- `store.log-20050317.gz`
- `store.log-20050318.gz`
- `store.log-20050319.gz`
- `store.log-20050330.gz`
- `store.log-20050331.gz`
- `store.log-20050402.gz`
- `store.log-20050405.gz`
- `store.log-20050406.gz`
- `store.log-20050407.gz`
- `store.log-20050408.gz`
- `store.log-20050409.gz`
- `store.log-20050412.gz`
- `store.log-20050413.gz`
- `store.log-20050414.gz`
- `store.log-20050415.gz`
- `store.log-20050419.gz`
- `usage.png`
- `usage_200411.html`
- `usage_200412.html`
- `usage_200501.html`
- `usage_200502.html`
- `webalizer.current`
- `webalizer.hist`

The terminal window also shows the prompt `s-207:/var/log/squid #` at the bottom.

El primer fitxer, *accres.log*, correspon a les peticions actuals realitzades a SQUID i la resta són fitxers comprimits de sortides anteriors on s'indica la data (20041124 correspon a 24-11-2004). El fitxer *cache.log* correspon a les sortides d'errors de SQUID. El fitxer *store.log* conserva informació d'aquelles peticions que es van renovant, esborrant o actualitzant al cau de SQUID. Finalment, el fitxer *rscsquid.conf* ens informa d'errors en el fitxer de configuració o l'engegada de SQUID.

Els fitxers que ens val la pena analitzar són els que estan catalogats com a *access.log**. Evidentment, seria impensable analitzar fitxer per fitxer (de text) per poder treure alguna conclusió clara i per aquest motiu utilitzarem una eina d'anàlisi.

Aquesta eina, que ens mostra el resultat de l'anàlisi via web, s'anomena WEBALIZER [2]. És programari lliure i es pot instal·lar de manera molt ràpida al nostre sistema (cal primer mirar si ja està instal·lat o si està al CD de la nostra distribució).

En aquest cas farem la instal·lació de la darrera versió. Necessitem baixar dos fitxers: el programa en ell mateix (amb la extensió `tar.gz`) [3] i la llibreria gràfica `gd` [4]. Aquest dos fitxers contenen les fonts dels programes i les eines per obtenir l'executable per la qual cosa ens cal compilar-los.

Primer caldrà descomprimir els paquets amb les comandes *tar* i *gz*. Després entrem a cadascuna de les carpetes que es creen i executem per ordre:

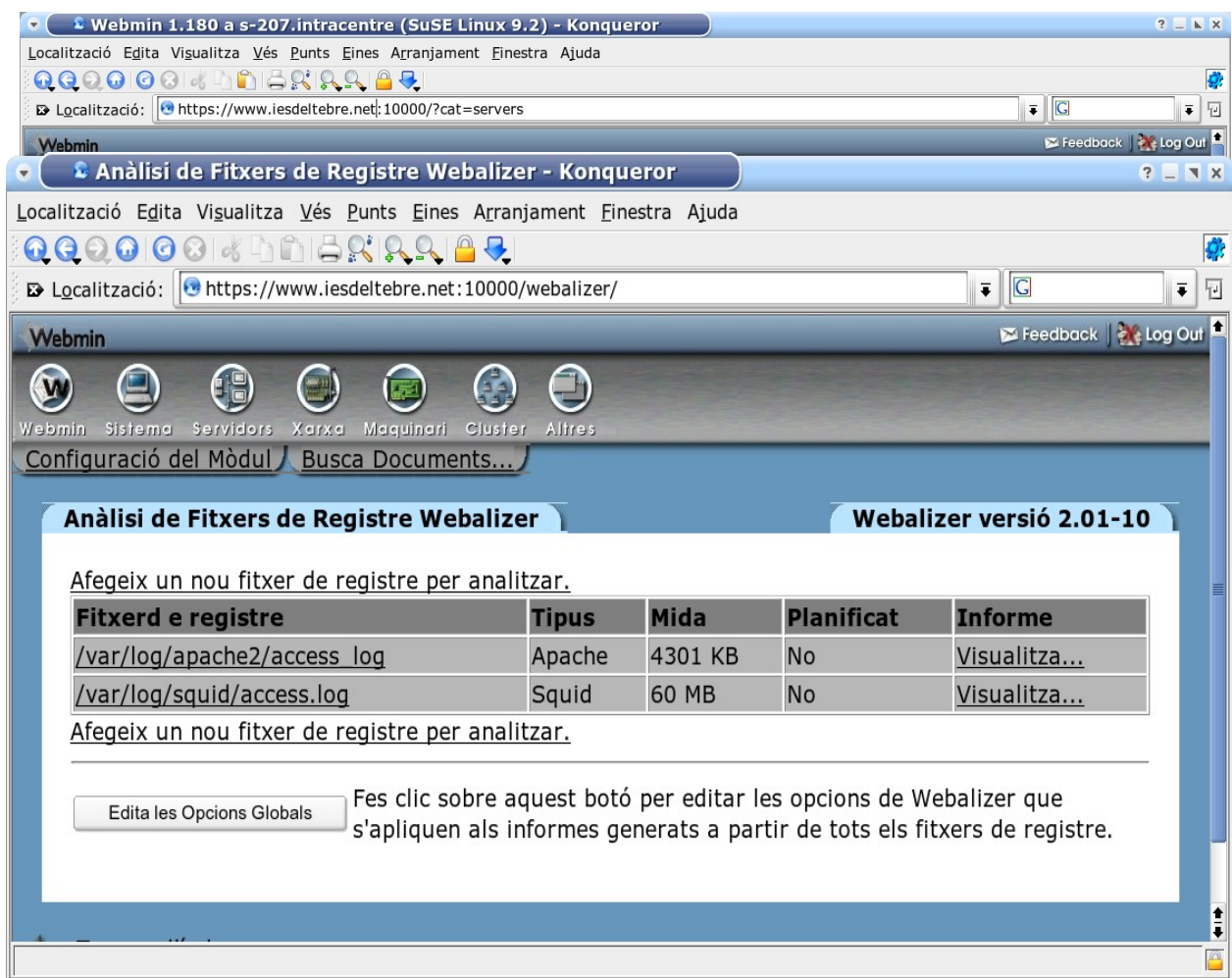
```
./configure
./make
./make install
```

Si tot és correcte tindrem instal·lat WEBALIZER al nostre sistema. Cal començar la instal·lació primer per la llibreria, ja que el programa presenta una dependència d'aquesta llibreria. Una vegada s'ha instal·lat el programa ens cal configurar-lo.

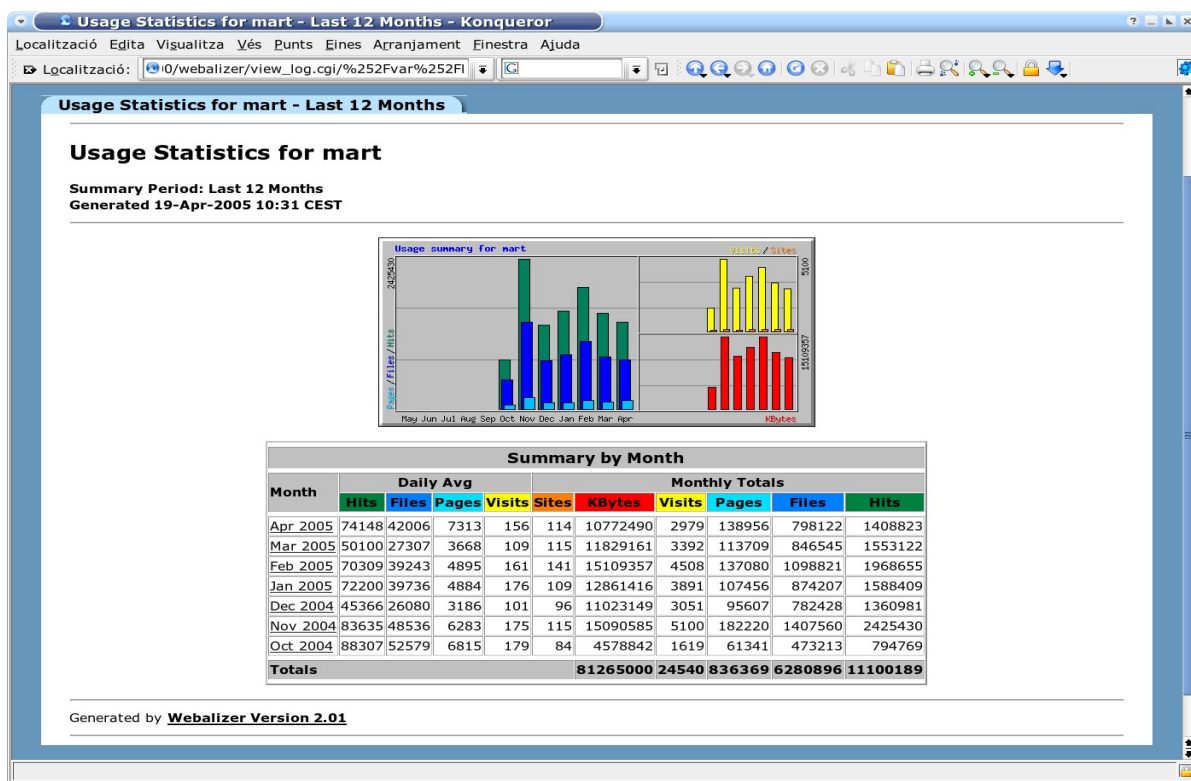
Com sempre, sota el directori */etc* trobarem els fitxers de configuració. En aquest cas és molt senzill, ja que únicament hi ha un fitxer de configuració a manipular que s'anomena */etc/webalizer.conf* i que, com sempre, ja bé configurat i pràcticament el podem usar directament. Únicament cal assegurar-se que existeix, ja que el procés d'instal·lació ens crea un fitxer d'exemple anomenat *webalizer.conf.sample* i que ens cal copiar com *webalizer.conf*.

Ara que ja tenim WEBALIZER instal·lat ens cal configurar una tasca que l'executi cada cert temps per crear l'anàlisi dels logs de SQUID i APACHE cada cert temps. Per fer això utilitzarem la comanda *cron*. Però, millor encara utilitzarem WEBMIN per fer aquesta configuració, ja que en aquest cas ens estalviarà molta feina i ens permetrà visualitzar via web el resultat de l'anàlisi.

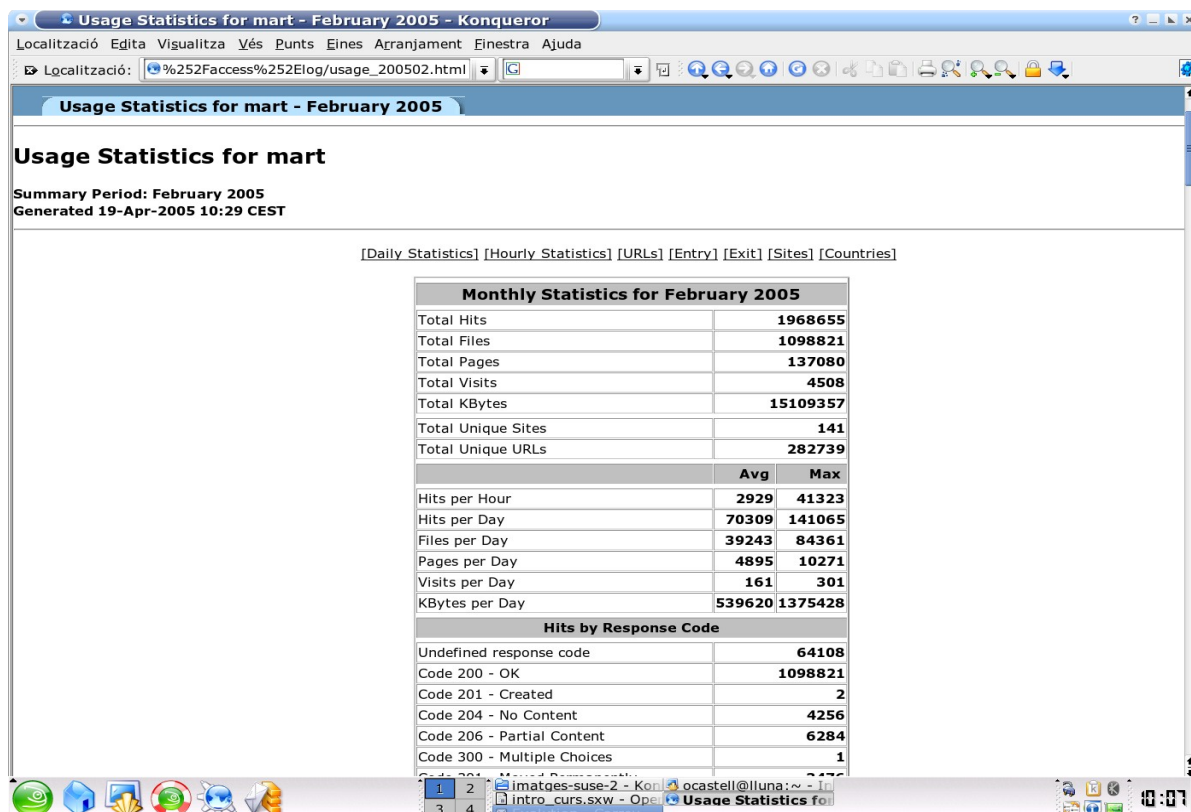
En WEBMIN, a la secció de *Servidors* trobem un apartat que es refereix a *Anàlisi de Registres Webalizer*. Cal configurar l'anàlisi per fer-lo cada cert temps i generar l'anàlisi corresponent, un per SQUID i altre per APACHE (recordem que el fitxer de log d'apache2 es troba a */var/log/apache2*):



Per veure l'informe resultant fem un clic a Visualitza i podrem veure l'informe que WEBALIZER extreu dels fitxers de log d'Apache o de SQUID. Si visualitzem de manera parcial l'informe de SQUID hauríem de veure alguna cosa semblant al següent:



Això és un resum general de l'activitat dels darrers 12 mesos. Si volem aprofundir una mica més ens cal fer un clic sobre el detall d'un més concret i veure la informació que ens proporciona. Per exemple, escollim el mes de febrer:

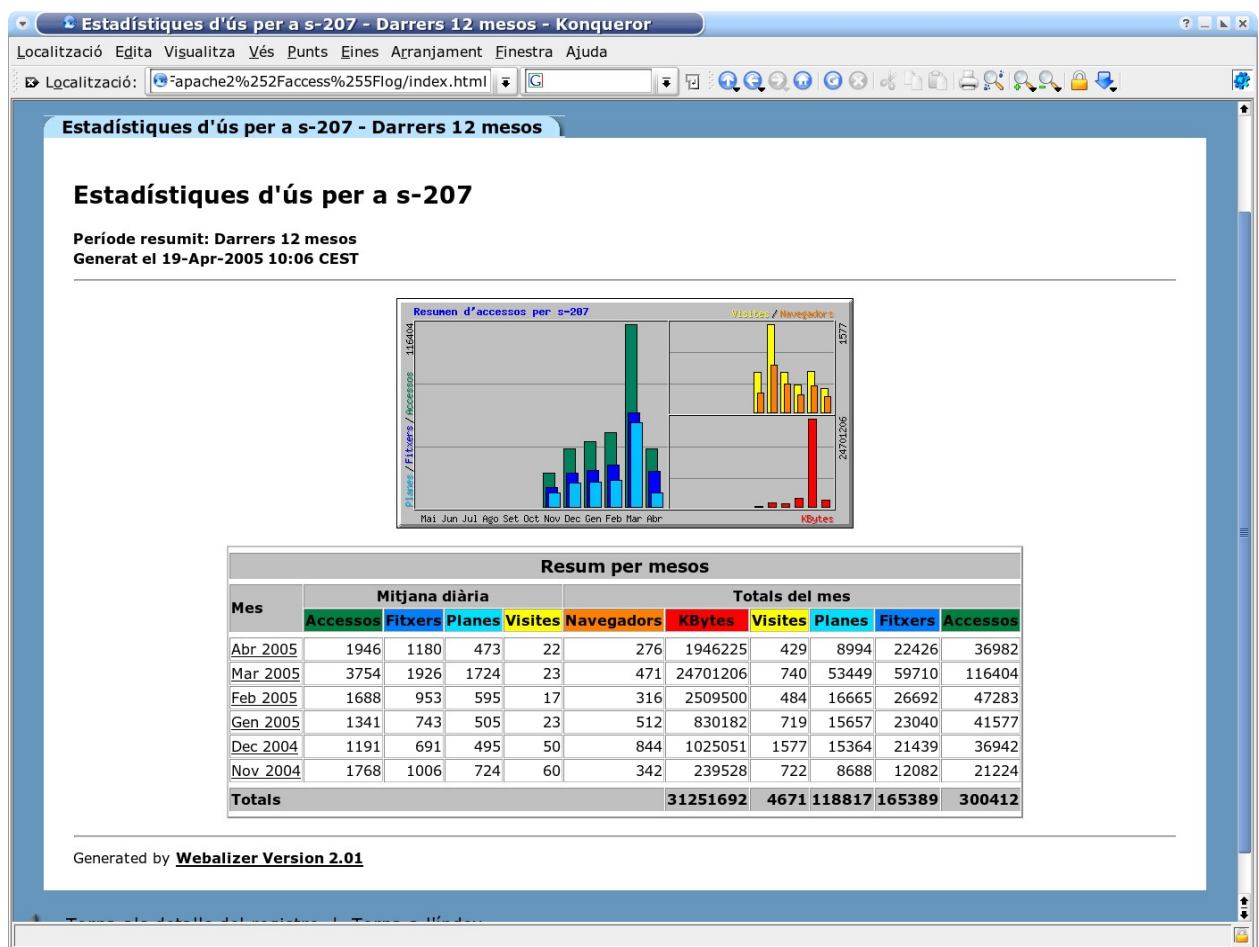


La informació que ens proporciona en aquest cas és molt més detalla (explicaré una part):

- Estadístiques per hores del dia i per dies del mes. Entre aquestes estadístiques ens diu les visites per resposta del servidor: les O.K. i les que estan prohibides, que ens dona una idea del funcionament de les nostres ACL.
- URLs visitades: les més visitades (top 30) i les que més transferència de Kbytes han suposat (top 10) i altres de menys interès.
- Les adreces IP dels clients (de la nostra LAN) que més transferència tenen (top 30) amb el número de visites i els kbytes de transferència, que després ens ho presenta amb un top 10 per als 10 clients que més kbytes han transferit.

Com es veu, ens dona informació molt interessant en un format tabulat i/o gràfic que ens ajuda molt a prendre les decisions corresponents del què fer al nostre servidor proxy si volem millorar l'estat de la navegació de la xarxa. L'estudi d'aquestes dades i estadístiques permeten extreure conclusions molt important i interessants sobre l'ús de la connexió a Internet: les hores de major i menor explotació, les pàgines visitades més usualment, els dominis més visitats, els clients que fan un ús més intensiu de la connexió a Internet i altres.

Per el cas del servidor Apache webalizer ens informa de coses molt semblants, però en aquest cas es refereixen a les visites que rep la nostra pàgina web:



Referències:

- [1] Projecte SQUID. Pàgina oficial: <http://www.squid-cache.org/>
- [2] Projecte Webalizer. Pàgina oficial: <http://www.mrunix.net/webalizer/>
- [3] Baixada del webalizer: <ftp://ftp.mrunix.net/pub/webalizer/webalizer-2.01-10-src.tgz>
- [4] Baixada de la llibreria GD: <http://www.boutell.com/gd/http/gd-2.0.33.tar.gz>