Servidor Proxy-Catxé (Squid)

"Proxy" significa intermediari. És a dir, un servidor proxy és un programa que fa d'intermediari entre un client i el servidor desitjat. Aquesta intermediació pot servir per diferents coses, com ara per controlar centralitzadament el tràfic d'una xarxa (i aplicar les polítiques d'accés convenients) o també per actuar de catxé de continguts.

Hi ha tres nivells de catxé: el que proporciona (si ho fa) el propi client que estem fent servir (en el cas concret de comunicacions de tipus HTTP, que és la gran majoria, aquí estaríem parlant d'un navegador), el que proporciona el nostre servidor proxy de la LAN o el que proporciona un altre servidor proxy "pare" o "germà"(sibling).

> NOTA: El proxy "pare" és l´encarregat de rebre la petició d´un proxy "fill", i en cas que no tingui els objectes demanats, els anirà a buscar a Internet per donar-los al fill.Els proxys pares s´utilitzen per aprofitar des d´un sol punt diferents línies d´accés a Internet.
> NOTA: El proxy "germà" també rep peticions d´un altre proxy B, però en el cas que no tinguin els objectes demanats, haurà de ser el proxy B qui ho hagi d´anar a buscar a Internet. Els proxys germans s´utilitzen per compartir el contingut de la catxé en disc entre diferents ordinadors de la mateixa xarxa.

Un tamany molt gran de catxé ocupa molt d´espai en disc, i suposa actualitzar moltes pàgines visitades per evitar inconsistències. Un tamany petit suposa un baix percentatge d´encerts en visitar una pàgina i per tant, poca eficiència. Hi ha diferents algoritmes possibles per actualitzar la catxé (a escollir mitjançant determinades directives del fitxer de configuració de l'Squid) , com ara sobreescriure els objectes menys recentment usats, o els objectes menys freqüentment usats, o els menys freqüentment usats o grans,etc

Per instal.lar el servidor proxy Squid (http://www.squid-cache.org) a Ubuntu, simplement cal fer *apt install squid* . Ha de quedar clar que Squid és només un proxy pels protocols HTTP,HTTPS i FTP. Si es vol controlar altres protocols, s´haurà de fer servir altres programes, com ara servidors SOCKS (un servidor Socks és un servidor proxy que funciona per tots els protocols de la capa d'aplicació) tals com Dante (http://www.inet.no/dante) o un servidor OpenSsh configurat adientment, entre d'altres.

Per iniciar i aturar el servei Squid, farem com sempre: *systemctl {start|stop|restart} squid* . Igualment, també podrem utilizar *systemctl status squid* per veure el seu estat.

Configuració del proxy als clients:

Per a què els clients HTTP (navegadors o altres) facin servir el proxy que hi hagi configurat a la nostra LAN, cal indicar-ho explícitament a cada màquina on s'utilitzin aquests clients. Això es pot fer de forma general a nivell de sistema (afectant així a tots els programes clients de cop) o bé individualment només per als programes de la màquina client que desitgem que "passin pel proxy" (si és que aquests ofereixen aquesta opció, que no tots els programes la ofereixen).

En el cas particular d'usar l'escriptori Gnome, per exemple, per fer que tots els seus programes "passin" per un determinat proxy (HTTP/S, FTP i SOCKS), cal anar al "Centre de Control->Xarxa->Servidor intermediari" i indicar allà manualment la IP d'aquest servidor proxy i el port per on estarà escoltant. El mateix es podria aconseguir executant les següents comandes (caldrà canviar el nom del servidor proxy indicat i el seu port d'escolta):

```
gsettings set org.gnome.system.proxy mode 'manual'
gsettings set org.gnome.system.proxy.http host 'proxy.localdomain.com'
gsettings set org.gnome.system.proxy.http port 8080
gsettings set org.gnome.system.proxy.https host 'proxy.localdomain.com'
gsettings set org.gnome.system.proxy.https port 8080
gsettings set org.gnome.system.proxy ignore-hosts "['localhost', '127.0.0.0/8']"
```

En el cas particular de voler configurar només el proxy pel navegador Firefox, caldrà llavors anar al seu menú "Preferències->General->Servidor intermediari de xarxa->Paràmetres" i allà indicar si es vol

utilitzar el proxy configurat al sistema general (opció per defecte), o bé si es vol configurar-ne un d'específic per ell (o bé si no se'n vol especificar cap explícitament, que també es pot).

També es pot indicar el servidor proxy a utilitzar per les comandes de terminal (*wget, curl, etc*) gràcies a l'establiment de determinades variables d'entorn específiques per aquesta tasca. Concretament:

export http_proxy=http://proxy.localdomain.com:8080
export https_proxy=$http_proxy
export no_proxy="localhost,127.0.0.0/8"

Un programa que pot ajudar a configurar totes aquestes possibilitats és https://github.com/himanshub16/ProxyMan

Arxius i directoris importants:

*/etc/squid/squid.conf : Arxiu de configuració del proxy, on modificarem les entrades adients

*/var/log/squid/access.log : Arxiu de registre de les peticions de pàgines del clients, amb informació diversa i mostrant si s´han bloquejat o no (es pot canviar a squid.conf).

*/var/log/squid/cache.log : Arxiu de registre de funcionament de la catxé, amb informació sobre errors, missatges d´inici,relacions entre catxés, etc (es pot canviar a squid.conf).

*/var/log/squid/store.log : Arxiu de registre de l'activitat del disc dur (del gestor d'emmagatzematge). Mostra quins objectes són expulsats de la catxé, quins són admesos i durant quant de temps, etc. És un fitxer opcional i es pot canviar a squid.conf (de fet, allà es pot deshabilitar tranquilament)

*/usr/share/squid/errors : Directori on es troben diferents subdirectoris per a cada idioma amb què volem mostrar els missatges d´error predefinits (es pot canviar a squid.conf).

*/usr/share/squid/mime.conf : Arxiu on es troben els tipus Mime reconeguts pel proxy.

*/var/spool/squid : Directori on es troba la catxé (si està activada la de disc), dividida en un seguit de subdirectoris interns (es pot canviar a squid.conf) i l'arxiu "swap.state", que manté un índex de tots els objectes que hi ha emmagatzemats a dins

Directives bàsiques de l´arxiu squid.conf:

A continuació es presenten les directives de configuració més importants d'Squid. En qualsevol cas, però, recomano llegir la documentació d'Squid, disponible a http://www.squid-cache.org/Doc/config

cache_dir ufs /var/spool/squid 100 16 256 Especifica l´espai del disc dur que s´utilitzarà de magatzem de la catxé. "Ufs" és el mecanisme de I/O del disc; altres mecanismes són “Aufs” (ufs asíncron amb fils, amb menys riscs d´accés a disc) ó “Diskd” (usa més memòria que “aufs”). Després ve el directori on es guarda la catxé. El 100 indica el tamany de la catxé en Mb. El 16 és el número de subdirectoris de primer nivell i el 256 és el número de subdirectoris de segon nivell. Aquests subdirectoris els utilitza l´squid a nivell intern per organitzar la catxé i millor no tocar els numerets aquests. De totes formes, per defecte aquesta directiva està comentada; això vol dir que l'Squid no utilitza catxé de disc i només n'utilitza en memòria
cache_mem 256 MB La quantitat de catxé que estarà a la memòria RAM.

http_port ip:3128

El port per defecte on escolta el Squid és el 3128. La Ip és opcional,i indica a quina tarja està vinculat squid.

ACLs:

Al quadre següent es mostren les directives que defineixen "grups d'elements" per tal d'indicar-los posteriorment en directives que denegaran (o no) el seu accés a serveis HTTP externs .

acl nom src ip/mascara ...
Direcció i màscara de l´ordinador o xarxa que realitza la petició.
Els punts suspensius indiquen que es poden posar varis valors ip/mascara (separats per espais) per definir l'acl com el conjunt de tots aquests valors
Ej: acl mired src 192.168.0.0/255.255.255.0 o també acl mired src 192.168.0.0/24
També s´accepten rangs d´ip: acl mired src 192.168.0.10-192.168.0.25/32 (¡no posar mask a la 1ª ip!)
O ips concretes: acl mired src 192.168.0.3/32 192.168.0.6/32 192.168.0.68/32
O especificar un fitxer on hi hagi en línies diferents les ips adequades: acl mired src "/ruta/fitxer"
Ej: acl tothom src 0.0.0.0/0.0.0.0
Ej: acl localhost src 127.0.0.1/255.255.255.255

acl nom dst ip/mascara ...
Direcció i màscara de l´ordinador o xarxa on va destinada la petició. Funciona igual que el cas anterior.

acl nom port nº nºinicial-nºfinal ...
Número de port (d'un servidor web, se suposa) on el client vol dirigir la seva petició.
Per indicar el port per on el propi Squid rep les peticions dels clients es pot usar l'acl *localport*

acl nom srcdomain .un.domi.ni ...
Domini origen de la petició (es determina per resolució Dns inversa de la ip de la màquina, així que s´haurà de tenir ben configurat el dns de la xarxa local).

acl nom dstdomain .un.domi.ni ...
Domini on va destinada la petició
Ej: acl google dstdomain .google.com .yahoo.es .info

acl nom srcdom_regex [-i] exprReg ...
Especifica una expressió regular que verifica el domini d´origen
L´expresió regular distingeix majúscules de minúscules a no ser que s´afegeixi l´opció -i
Ej: acl totgoogle dstdom_regex -i google\..*

acl nom dstdom_regex [-i] exprReg ...
Especifica una expressió regular que verifica el nom DNS de destí.
L´expresió regular distingeix majúscules de minúscules a no ser que s´afegeixi l´opció -i

acl nom urlpath_regex paraulaconcretadinslurl ...
Permet especificar expressions regulars per comprovar tot allò que segueix al nom DNS en una url
També es pot crear un arxiu de texte que contingui les paraules concretes dins d'aquesta "cua" de la url que es volen controlar, cadascuna en una línia diferent, fent: *acl nom urlpath_regex "rutaarxiutexte"*

acl nom url_regex [-i] exprReg ...
Permet especificar expressions regulars per comprovar una url completa, incloent el http:// inicial
També es pot crear un arxiu de texte que contingui les expressions regulars corresponents a les diferents Urls, separades per salts de línia i fer *acl nom url_regex "rutaarxiutexte"* . Aquesta seria una manera per exemple de tenir en un fitxer una llista negra de paraules, que es prohibirien amb la corresponent línia

http_access (veure més endavant).
Ej: acl publicitat url_regex ^http://adserver.* ^http://popup.*
Ej: acl peticionsmp3 url_regex -i mp3$ ogg$ wma$

L´acl dstdom_regex es refereix només al domini, urlpath_regex a tot el que ve després del domini, i url_regex a tota la url sencera. És a dir, si tenim: http://www.yahoo.com/sites/index.php?user=on , dstdom_regex només miraria www.yahoo.com, urlpath_regex miraria /sites/index.php?user=on i url_regex tot plegat.

acl nom referer_regex [-i] exprReg ...
Comprova la direcció de l´enllaç que s´ha pulsat per accedir a una determinada pàgina.
Ej: acl vedegoogle referer_regex http://www.google.*

acl nom time diasetmana horainici-horafinal
Els dies de la setmana venen donats per les inicials dels dies en anglès: S-Sunday M-Monday T-Tuesday W-Wednesday H-Thursday F-Friday A-Saturday . Les hores s´escriuen en format 00:00-24:00, i la primera ha de ser menor que la segona. Es pot posar només dies (tot el dia sencer), només rang d´hores (per tots els dies) o tots dos.
Ej: acl horari time MTWHF 8:00-15:00

acl nom req_mime_type -i ^tipusMime$ ...
Comprova el tipus de petició Mime que realitza un client, i es pot utilitzar per detectar certes descàrregues de fitxers.
Ej: acl subida req_mime_type -i ^multipart/form-data$
Ej: acl javascript req_mime_type -i ^application/x-javascript$
Ej: acl estilos req_mime_type -i ^text/css$
Ej: acl audiompg req_mime_type -i ^audio/mpeg$

acl nom rep_mime_type -i ^tipusMime$ ...
Comprova el tipus de resposta Mime del servidor que rep el proxy. Aquest tipus de acl analitza una resposta del servidor, i per tant només li afecten les regles definides a http_reply_access, no les regles http_access que s´apliquen a les peticions.
Ej: els anteriors
Ej: acl executables rep_mime_type -i ^application/octet-stream$

acl maxconn nº
S'aplica quan el client (identificat per la seva ip) té més del màxim de conexions Http especificades
També està l'acl “max_user_ip nº” que indica que en el cas de què existeixi autenticació per usuaris (veure més endavant), des de quantes ips diferents com a màxim es pot autenticar a la vegada (veure també authenticate_ip_ttl).

acl nom method GET POST ...
Comprova el mètode HTTP que utilitza la petició. D'altra banda, també està l'acl “proto HTTP FTP” per distingir el protocol que utilitza la petició. Com a protocol també es pot especificar “cache_object” (és un protocol propi de squid -basat en http- que retorna informació sobre com està configurada la catxé o com s´està executant).

acl nom http_status 200 404 500- 300-400...
Comprova el codi de resposta provinent del servidor web que contesta al client

acl nom browser [-i] exprRegqueindicaelValordelUserAgentdelclient ...
Comprova quin tipus de client és el que fa la petició, a partir de la capçalera User-Agent

Veure les acl ja predefinides a squid.conf, sota el comentari “Recommended minimum configuration”

A partir d'aquí, establirem quins "grups d'elements" poden accedir a l'exterior i quins no amb la directiva *http_access*, així:

http_access {deny|allow} [!] unaacl|otra...

Crea una regla que denega o permet l´accés a fora del proxy (normalment, a Internet via port 80) a una acl concreta. Si la acl ve precedida d´un ! indica que la regla s´aplica a tot allò que no estigui contingut en aquella acl.
Ej: http_access allow !horari -Permet accés fora de l´horari laboral (especificat a l'acl "horari")-
Ej: http_access deny horari -Denega accés dins de l´horari laboral (és a dir, el mateix que l´anterior)-

Les regles http_access es verifiquen en l´ordre que estan escrites, i es deixen de verificar quan la petició coincideix amb alguna regla.Si no es compleix cap regla, l´acció per defecte és fer el contrari que la darrera regla de la llista (per això és millor ser explícit:millor posar sempre al final un deny all o un allow all)

És important molt recalcar que si es posen diferents acl en la mateixa regla, el que estem fent es denegar/permetre SIMULTANIAMENT a tots els ens representats per cadascuna de les acl l´accés a fora. És a dir, en una regla, posar vàries acl implica escriure un AND entre aquestes acl. En canvi, les diferents regles tenen una relació OR entre sí . L´Squid mira si es compleix la primera regla OR la segona OR la tercera...fins que la troba.

Un nom d´acl pot estar a vàries regles.

Alguns exemples:

Ej:Permetre a uns ordinadors determinats l´ús d´Internet només durant un període de temps, fora del qual està prohibit:
acl xarxa src 192.168.0.0/255.255.255.0
acl horaripermes time MTWHF 10:00-16:00
http_access allow xarxa horaripermes
http_access deny xarxa

Ej: Compte amb escriure el que es vol...
acl red1 src 192.168.0.0/24
acl red2 src 192.168.1.0/24
http_access allow red1 red2
permetria l´accés a totes aquelles conexions que procediran a la vegada de red1 i red2 (impossible). Si el que volíem era permetre l´accés a les dues xarxes, s´hauria de fer
acl red1 src 192.168.0.0/24
acl red2 src 192.168.1.0/24
http_access allow red1
http_access allow red2
o bé, de forma més simple:
acl redes src 192.168.0.0/24 192.168.1.0/24
http_access allow redes

Ej: denegar tots els fitxers de tipus exe que provinguin de virusfijo.com
acl antiexe url_regex -i exe$
acl vfijo dstdomain virusfijo.com
http_access deny antiexe vfijo

Ej: denegar l'accés a llocs web especificats a l'arxiu "bansit.txt"
acl xarxalocal src 192.168.0.0/255.255.255.0
acl bansit url_regex "/etc/squid/bansit.txt"
http_access allow localhost
http_access deny bansit
deny_info pagina bansit
http_access allow xarxalocal
http_access deny all
error_directory /usr/share/squid3/errors/personalitzat

Ej: restringir l´accés a cert tipus de contingut a certs horaris per una determinada xarxa
acl classe src 192.168.9.0/255.255.255.0
acl mati time MTWHF 09:00-15:00
acl musica urlpath_regex \.mp3$
http_access allow mati classe !musica

Ej: allow requests FROM the za domain UNLESS they want to go to \.com or \.net
acl bad_dst_TLD dstdom_regex \.com$ \.net$
acl good_src_TLD srcdom_regex \.za$
http_access deny bad_dst_TLD
http_access allow good_src_TLD

Ej: no permetre que només puguin accedir a la nostra catxé els ordinadors de la intranet, no des d´Internet
acl pepe proto HTTP
http_access deny !pepe
acl pepa method GET POST
http_access deny !pepa
acl mired src 192.168.0.0/24
http_access allow mired

Ej proposat: denegar Internet a un ordinador concret només, dins d´un horari determinat
Ej proposat: denegar l´accés a tots els dominis acabats en .com per tota la xarxa local interna.

En el cas d'una petició denegada, es produeix un error. Els missatges d'error mostrats a l'usuari es poden configurar amb les següents directives:

error_directory "/ruta/directori/onestan/elsmissatges/derror/predefinits" Típicament és la carpeta /usr/share/squid/errors/ca o bé /usr/share/squid/errors/personalitzat.
deny_info nomarxiudinsdeldirectoriderrorsambelmissatgederrorHTMLpersonalitzat unaaclconcreta Quan l´acl d´aquesta línia obté una denegació (per exemple, això seria un error) de l´Squid, es mostrarà a la pantalla del navegador l´arxiu HTML especificat, el qual ha d´estar a la carpeta especificada a la directiva error_directory.

Es poden consultar unes quantes configuracions d´acl ja predefinides que poden ser útils per casos concrets a http://wiki.squid-cache.org/ConfigExamples

Altres directives:

*Relacionades amb els tamanys dels objectes transferits i guardats:

Per controlar el tamany dels fitxers o pàgines que es baixen: *request_header_max_size 10KB* *reply_header_max_size 10KB* *request_body_max_size 512K #El tamany màxim d'una capçalera PUT ó POST . Si es posa a 0 és il.limitat* *reply_body_max_size 512KB [unaacl|otra...] #Molt interessant per limitar el tamany de les descàrregues.*
També estan les directives següents per controlar el tamany dels objectes emmagatzemats a la catxé: *maximum_object_size 4096KB* (en disc) *maximum_object_size_in_memory 4096 KB* (en memòria)

*Relacionades amb la gestió de la catxé:

Per especificar el tipus d´objectes que s´emmagatzemaran a la catxé, i amb quina freqüència es refrescaran *refresh_pattern [-i] \.wav$ 10080 90% 10080* *refresh_pattern ^http: 2880 40% 4320* *refresh_pattern . 240 20% 4320 #Qualsevol fitxer* on el primer paràmetre és una expressió regular que fa referència a l'objecte a catxejat, el segon és el número mínim de minuts que l´objecte es considera actualitzat (i per tant no es pot esborrar) el tercer és un valor pels casos en què el servidor remot no dóna informació sobre si l'objecte ha expirat o no; llavors la vida de l'objecte es calcula com el % de la diferència entre el temps actual i el last modified time dels headers HTTP, el quart és el número màxim de minuts que l´objecte es considera actualitzat (passats-els es podria esborrar) Altres directives relacionades són *cache_swap_low 90* i *cache_swap_high 95*
cache_replacement_policy algoritmeperactualitzarcontingutdelacatxededisc Afecta a la catxé de disc. Hi ha tres algoritmes possibles bàsics: LRU (menys recentment usats, fora) , LFUDA (menys freqüentment usats, fora), GDSF (menys freqüentment usats o grans, fora).
memory_replacement_policy algoritmeperactualitzarcontingutdelacatxealaram Hi ha tres algoritmes possibles bàsics: LRU, LFUDA o GDSF

Autenticació:

Mitjançant Squid es pot aconseguir que només un grup d´usuaris puguin accedir a Internet (des de qualsevol màquina). És a dir: autenticar l'accés a l'exterior a l'estil dels "portals captius" dels aeroports, hospitals, etc. Per fer això es necessita un programa extern que sigui l'encarregat d'obtenir (i validar) el nom d´usuari i la contrasenya emmagatzemats en alguna font externa (com l'arxiu /etc/passwd, un arxiu alternatiu, una base de dades SQL, un arbre LDAP, etc). Molts d'aquests programes externs (anomenats "helpers") ja es proporcionen per defecte amb la instal.lació estàndar de l'Squid (es troben ubicats a /usr/lib/squid) i n'hi ha força, com ara:

"basic_pam_auth" (obté els logins del sistema Linux)
"basic_smb_auth" (obté els logins del sistema Windows/Samba)
"basic_ncsa_auth" (obté els logins de l´arxiu htpasswd típic de l´Apache)
"basic_db_auth" (obté els logins d'una base de dades relacional)
"basic_ldap_auth" (obté els logins d´un servidor LDAP/AD)

Les directives que s´ha de posar al squid.conf per configurar tot plegat és...:

auth_param basic program /ruta/modul [paràmetres_extra]

...i al lloc habitual on s'ubiquen dins del fitxer les regles http_access, cal afegir:

acl nomaclqualsevol proxy_auth REQUIRED
http_access {allow|deny} nomaclqualsevol ...

Opcionalment, es poden afegir, després de la línia *auth_param* anterior, altres línies que especifiquen amb més exactitud el comportament del mòdul utilitzat, com per exemple:

auth_param basic children nºprocessosdautentificacio
auth_param basic realm Missatge a mostrar
auth_param basic credentialsttl tempsqueesrecordaralacontrasenyavalida {hours | minute}
auth_param basic casesensitive {on|off}

*Per utilitzar els usuaris del sistema Linux, s´hauria d´escriure:

```
auth_param basic program /usr/lib/squid/basic_pam_auth
#auth_param basic children 5
#auth_param basic realm Squid
#auth_param basic credentialsttl 2 hours
 ...
acl hola proxy_auth REQUIRED
http_access allow hola mired    #per exemple
```

A més, fer una altra cosa: crear un arxiu anomenat “squid” dins la carpeta /etc/pam.d amb el següent contingut (o equivalent):

```
auth            required            pam_unix.so
account         required            pam_unix.so
```

I finalment, cal aplicar el “bit setuid” al programa extern per tal de què pugui llegir el fitxer de contrasenyes del sistema (/etc/shadow) Això es fa executant la comanda *chmod u+s /usr/lib/squid/basic_pam_auth*

*Per utilitzar els usuaris presents dins d'un arxiu htpasswd típic de Apache, s´hauria d´escriure:

```
auth_param basic program /usr/lib/squid/basic_ncsa_auth  /ruta/arxiu/htpasswd
 ...
acl hola proxy_auth REQUIRED
http_access allow hola mired   #per exemple
```

I ja està. L´usuari propietari squid ha de poder llegir i escriure en l´arxiu de contrasenyes.

*Per utilitzar els usuaris presents dins d'una base de dades MySQL, s'hauria d'escriure...:

```
auth_param basic program /usr/lib/squid/basic_db_auth \
--dsn "DSN:mysql:host=x.y.z.w;port=3306;database=squid" \
--user usuariBD --password contrasenyaBD --plaintext --persist
...
acl hola proxy_auth REQUIRED
http_access allow hola mired    #per exemple
```

...on "x.y.z.w" és la direcció IP del servidor de bases de dades i on s'ha suposat que allà s'han executat prèviament les següents sentències (el nom de la BDD i el de la taula han de ser el indicats exactament):

```
mysql> create database squid;
mysql> grant select on squid.* to usuariBD@localhost identified by 'contrasenyaBD';
mysql> CREATE TABLE passwd (
            user varchar(32) NOT NULL default '',
            password varchar(35) NOT NULL default '',
            enabled tinyint(1) NOT NULL default '1',
            fullname varchar(60) default NULL,
            comment varchar(60) default NULL,
            PRIMARY KEY  (user)
);
mysql> insert into passwd values('pepito','1234',1,'Pepito Pérez','Un usuari vàlid');
```

Trobareu més informació sobre l'autenticació d'usuaris a http://wiki.squid-cache.org/SquidFaq.

El fet d'haver d'obligar a l'usuari a validar-se per tal de tenir accés a “Internet” és el que se'n diu un portal captiu, tal com hem dit. Hi han programes específics especialitzats en mantenir un portal d'aquest tipus. Exemples en teniu a http://es.wikipedia.org/wiki/Portal_cautivo#Portales_Cautivos_por_software

Proxy transparent:

Un servidor proxy transparent és aquell que obliga als clients HTTP (el navegador o altres) a utilitza-lo, encara que aquests clients tinguin configurat explícitament que no volen utilitzar cap servidor proxy! El truc està en fer què el servidor proxy sigui la porta d'enllaç per defecte de les màquines clients i a més que estigui escoltant al port 80, (que és on els clients HTTP envien les peticions per defecte). Si la petició és acceptada, és llavors quan el servidor proxy la redirecciona allà on realment ha d´anar (normalment, el router d'accés a l'exterior).

En aquesta configuració, el servidor proxy sol disposar de dues tarjes de xarxa: una connectada (mitjançant un switch) a la xarxa dels clients i una altra connectada al router d'accés a l'exterior, tal com es mostra a la figura (suposarem, en aquest cas, que el router té -per exemple- la IP 192.168.0.1, que el servidor proxy té la tarja anomenada *wan0* connectada al router amb -per exemple- la IP 192.168.0.2 i una tarja connectada al switch de la LAN anomenada *lan0* amb -per exemple- la IP 192.168.33.254 i que els clients d'aquesta xarxa pertanyen a la xarxa 192.168.33.0/24 i tenen com a porta d'enllaç la IP que veuen de l'Squid (molt important això!!):

Amb aquesta configuració, el proxy anirà rebent les peticions del client per una tarja, les processarà i, si s'escau, les reenviarà internament a l'altra tarjeta per entregar-les al router efectiu. Per a què Squid pugui funcionar així, però, primer cal configurar adientment el sistema operatiu subjacent; concretament, cal activar l'"Ip-forwarding" i definir una regla mínima pel tallafocs de Linux (IPtables):

1.-Activar l'"IP-forwarding" . Això vol dir activar l'"entroncament" entre les dues tarjes del proxy de manera que hi hagi conexió entre elles perquè per defecte cadascuna funciona de forma independent i no hi ha cap comunicació entre elles. La forma més senzilla de fer-ho és editant (o afegint) la següent línia dins del fitxer /etc/sysctl.conf i reiniciar tot seguit:

net.ipv4.ip_forward = 1

NOTA: Si es vol desactivar l'"IP-forwarding", només cal substituir l'1 per un 0 a la línia anterior (i reiniciar)
NOTA: Per veure el valor actual de l'"IP-forwarding" es pot executar la comanda *sysctl -p*

Some routers are configured with reverse path forwarding, which means the source ip has to make sense; it can't come in an interface that wouldn't route to that source address.

2.-Definir una regla (al menys) del tallafocs, similar a aquesta:

iptables -t nat -A POSTROUTING -o wan0 -j MASQUERADE

Aquesta regla el que fa és "recordar" quin client fa cada petició, de manera que quan sigui l'Squid qui la faci al router, en rebre la resposta la pugui reenviar al remitent original. No obstant, per a què aquesta regla sigui permanent als següents reinicis del sistema, es poden utilitzar els scripts *iptables-save* (per guardar la configuració actual -que se suposa que és la ja definitiva que volem- en un determinat fitxer) i *iptables-restore* (per llegir aquest fitxer i aplicar la configuració allà escrita al sistema; aquest script es pot indicar, per exemple, a la línia "up" dins de l'arxiu /etc/network/interfaces (hi pot haver una per cada tarja diferent).

NOTA: Ara per ara se surt del nostre tema la teoria d'Iptables. Si algú vol profunditzar en aquest tema, llegiu http://www.linuxhomenetworking.com/wiki/index.php/Quick_HOWTO_:_Ch14_:_Linux_Firewalls_Using_iptables

Un cop configurat el tallafocs i l'IP-Forwarding del sistema, per poder treballar com a proxy transparent l'Squid ha de també configurar-se de forma adient. Això consisteix, bàsicament en:

3.-Afegir a la seva directiva *http_port* , després del número de port (que hem dit que haurà de ser el 80) la paraula *intercept*

El fitxer “proxy.pac” i el protocol Wpad:

El fitxer “proxy.pac” és un fitxer que conté una seqüència d'ordres en Javascript que el navegador llegeix cada cop que es posa en marxa i que estableix el servidor proxy que utilitzarà. Aquest fitxer no es troba físicament a la màquina del client, sinó que s'obté d'algun servidor HTTP remot que ha d'estar accessible a la LAN. Els navegadors hi han de poder accedir ma aquest fitxer mitjançant una URL de tipus http que cal incloure dins la seva configuració (al Firefox, això es fa sel.leccionant l´última opció que apareix al quadre de configuració dels proxy a usar, anomenada "URL de configuració automàtica del servidor intermediari"). Per exemple, així: *http://ip_del_servidor_web/proxy.pac*

També existeix la possibilitat d'oferir el fitxer “proxy.pac” sense haver d'explicitar a la configuració del navegador la seva ubicació, sinò oferint aquest fitxer automàticament en l'arranc de la màquina client via un servidor Dhcp que tinguem a la nostra xarxa; és el que se'n diu el protocol d'autodescoberta “WPAD”. Al Firefox pot utilitizar aquest protocol si sel.leccionem l'opció "Detecta automàticament els paràmetres del servidor intermediari per aquesta xarxa" que apareix al seu quadre de configuradió del proxy. El problema és que aquest protocol WPAD no és estàndar i no tots els navegadors el suporten.

En el cas de fer servir un servidor Dhcp, haurem d'escriure les següents línies al punt adequat de l'arxiu dhcpd.conf:

-A la secció de configuració global:
option wpad-url code 252 = text;

-A la secció de configuració de la subxarxa asignada:
option wpad-url = "http://ip_del_servidor_web/wpad.dat";

En qualsevol dels casos anteriors (escrivint la ubicació directament a la configuració del navegador o via Dhcp), a més necessitarem configurar un servidor Http incloent les següents línies o bé a l'arxiu de configuració d'un virtualhost concret o bé dins de l'arxiu de configuració /etc/apache2/mods-available/mime.conf (o bé, més general, al fitxer del sistema /etc/mime.types):

```
AddType application/x-ns-proxy-autoconfig .pac
AddType application/x-ns-proxy-autoconfig .dat
```

I aquest fitxer, ¿quin contingut té? Un exemple podria ser:

```
function FindProxyForURL (url,host)  {
 if ((url.substring(0,5)!= "http:") &&  (url.substring(0,4) != "ftp:"))
        return "DIRECT";
 if (isPlainHostName(host) || shExpMatch(host,"192.168.*") ||
 shExpMatch(host,"127.*") ||  (url.substring(11,23) == "xtec.es:8800"))
        return "DIRECT";
 else
        return "PROXY 192.168.0.2:8080; PROXY proxy.xtec.es:8080; DIRECT";
 }
```

El primer 'if' d'aquest fitxer obliga al navegador a accedir directament a Internet sense passar per cap proxy si a la URL no es demana cap protocol que l'Squid reconegui.El segon bloc 'if - else', evita que es demanin al proxy continguts que no val la pena desar a la cache, per exemple si accedim al contingut del propi servidor, a pàgines de les màquines de la xarxa local, a serveis 'especials' com per exemple el correu

web de la XTEC pel port 8800, etc. La resta de continguts que no compleixen cap dels punts del primer bloc, es direccionen directament a l'Squid, seguint la pauta indicada dins de l'"else".En el cas que el proxy 192.168.0.2 no respongui, el navegador intentarà utilitzar el proxy.xtec.es, si aquest també falla canviarà el tipus d'accés, connectant directament a Internet sense que l'usuari hagi canviat cap opció del navegador.

El avantatges més clars d'aquest sistema respecte la configuració manual són:

- Permet direccionar l'accés directe a dominis que així ho requereixin de forma més fàcil que configurant internament el servidor proxy.

- Permet configurar l'accés a través del proxy més òptim per a cada connexió, encaminant els clients de forma transparent. Si es volgués fer amb la configuració manual l'usuari hauria de saber quin és el proxy més adient i caldria canviar el 'host' i potser també el 'port' dins les opcions del navegador.

- En cas de fallada del proxy principal, permet definir altres equips de reserva i si aquests també 'cauen', estableix la connexió de forma directa a Internet sense cap intervenció de l'usuari. Amb la connexió manual, si el proxy falla l'usuari ha de desactivar-lo o si coneix les dades del proxy 'de reserva' ha de reflectir-ho dins la configuració del navegador.

No obstant, és evident que aquesta manera de fer requereix tenir un servidor dhcp/dns+http funcionant a la xarxa interna. Per a més informació sobre aquest tema, consultar http://en.wikipedia.org/wiki/Proxy_auto-config o, més específic, http://findproxyforurl.com

Ús d'Squid per filtrar, registrar o monitoritzar tràfic HTTPS:

Fins ara només hem considerat el tràfic HTTP. No obstant, avui dia la majoria de pàgines web són oferides mitjançant una connexió HTTPS. Desgraciadament, Squid per defecte és incapaç d'entendre,catxejar o analitzar aquest tipus de tràfic justament perquè va encriptat. Això fa que la seva utilitat com a proxy-catxé o com a censurador-monitoritzador sigui nul.la en aquest tipus de context. Afortunadament, existeix una manera de configurar l'Squid per fer-li actuar com un atacant de tipus "man in the middle", trencant així la seguretat TLS (no cal dir que aquesta pràctica s'hauria de notificar als usuaris clients perquè s'estaria violant la seva intimitat!). Per tenir clar en quines circumstàncies aquest atac pot tenir èxit hem de tenir clar, però, la diferència entre les directives *http_port* i *https_port*

Si els clients tenen indicat manualment que el seu servidor proxy per HTTP (i HTTPS!) és un ordinador executant Squid configurat de la forma que ja hem vist (bàsicament, escoltant amb la directiva *http_port*) , aquest ordinador podrà catxejar i monitoritzar tot el tràfic que sigui HTTP però en el cas de que sigui HTTPS no. No obstant, el que sí que fa Squid és funcionar "d'entroncament tonto" per aquest tipus de comunicació: és a dir, el client sempre li demanarà explícitament via HTTP (mitjançant una petició especial anomenada CONNECT) poder-se connectar al servidor HTTPS remot i el que farà Squid és establir llavors una connexió TCP entre aquest client i el servidor remot demanat de forma que la informació que passi per allà sigui completament opaca pel proxy (perquè de fet, tècnicament serà una conexió TLS directa entre client i servidor remot). És per això que la navegació HTTPS funciona a pesar de què Squid només és un proxy HTTP.

En el cas de tenir configurat un proxy transparent, amb el paràmetre *intercept* de la directiva *http_port* es pot seguir actuant com a proxy transparent de tipus HTTP (i HTTPS actuant com a "entroncador") però, en aquest cas, s'afegeix la possibilitat de què Squid pugui rebre, entendre i manipular directament peticions HTTPS de forma "nativa". Per això cal afegir i configurar adientment una nova directiva (entre d'altres) anomenada *https_port nºport intercept* No obstant, per a què aquesta directiva es tingui en compte, cal que l'Squid hagi sigut compilat amb les opcions –enable-ssl i –enable-ssl-crtd . Desgraciadament, tal com es pot veure si executem *squid -v*, aquestes opcions no estan presents en la versió del programa empaquetada en els repositoris oficials d'Ubuntu.

La seguretat TLS es pot trencar utilitzant la mateixa tècnica (anomenada "SSL-Bump") en els dos casos, ja sigui amb *http_port* (si el proxy està configurat manualment als clients) com amb *https_port* (si el

proxy és transparent) . No obstant, Squid ha d'estar compilat amb les opcions –enable-ssl i –enable-ssl-crtd obligatòriament. Per solucionar aquest problema, actualment hi ha dues opcions: o bé compilar l'Squid a mà a partir del seu codi font indicant les opcions desitjades (aquesta via es troba detallada en aquest post: https://www.chasewright.com/install-squid-from-source-on-ubuntu-16-04/) o bé afegir un repositori de tercers anomenat "Diladele" (compatible amb Ubuntu 16.04) que ofereix l'Squid empaquetat i ja compilat de la forma desitjada. Aquesta segona manera és, òbviament, molt més directa i fàcil de fer; només cal realitzar els següents passos:

#Afegir la clau del repositori Diladele
wget -qO - http://packages.diladele.com/diladele_pub.asc | sudo apt-key add -
#Afegir el repositori Diladele (i, en el cas d'estar usant Ubuntu 17.10, el repositori d'Ubuntu 16.04 també)
echo "deb [arch=amd64] http://ubuntu16.diladele.com/ubuntu/ xenial main" > /etc/apt/sources.list.d/di.list
echo "deb http://archive.ubuntu.com/ubuntu/ xenial main" >> /etc/apt/sources.list
#Actualitzar la llista de repositoris reconeguts pel sistema
apt update
#Fer la instal.lació (indicant que volem els paquets per la versió Xenial, que és l'oferida per Diladele)
apt -t xenial install libecap3 squid-common squid

La configuració necessària consisteix en realitzar els passos següents:

1.-Crear una clau privada i un certificat, tal com ja vam estudiar amb l'Apache. Igual que en aquell cas, el certificat podria ser signat per una CA (que podríem ser nosaltres mateixos) mitjançant l'enviament d'un fitxer .csr o bé pot ser autosignat. Escollirem aquesta darrera opció. En aquest cas, però, a l'Squid li agrada més el format PEM, el qual incorpora en un mateix fitxer tant el certificat com la clau privada, així que generarem tot plegat amb la següent comanda (com a root):

openssl req -newkey rsa:4096 -x509 -nodes -keyout /etc/squid/squid.pem -out /etc/squid/squid.pem

NOTA: Si es fes servir la llibreria GnuTLS en comptes d'OpenSSL, la comanda seria certtool

És molt important no afegir CAP "Common Name" quan es respon interactivament. Això és perquè aquest certificat serà usat per crear dinàmicament al seu torn un altre certificat per cada petició del client amb el nom indicat a la petició. Aquests certificats dinàmics són els que Squid retorna al client, el qual, com veurà que no està signat per una CA confiable reaccionarà en conseqüència (en el cas dels navegador, apareix la típica pantalla de "Conexió no segura"). A partir d'aquí, serà l'Squid qui iniciarà una sessió HTTPS amb el servidor remot, desencriptarà el contingut rebut i el tornarà a encriptar amb el seu certificat "ad-hoc" per reenviar-lo al client.

NOTA: Els navegadors moderns poden emprar tècniques com HPKP (https://tools.ietf.org/html/rfc7469) o "pins de certificats" (https://www.imperialviolet.org/2011/05/04/pinning.html) que poden mitigar aquest tipus d'atac "man in the middle".

Cal ajustar els permisos del fitxer generat: ***chmod 400 /etc/squid/squid.pem***

2.-Ja que Squid generarà dinàmicament molts certificats "ad-hoc" (un per cada petició dels clients), cal crear una catxé de certificats (la crearem a /var/lib/ssl_db). Això es fa amb la comanda (com a root):

/usr/lib/squid/ssl_crtd -c -s /var/lib/ssl_db

Cal ajustar el propietari del fitxer generat: ***chown squid: /var/lib/ssl_db***

3.-Configurar les directives *http_port* i/o *https_port* (segons el cas de tenir un proxy manual o transparent, respectivament) per a què faci servir la tècnica "SSL-Bump". Això consisteix en afegir les següents línies a l'arxiu squid.conf:

http_port 3128 ***ssl-bump cert=/etc/squid/squid.pem dynamic_cert_mem_cache_size=4MB***
sslcrtd_program /usr/lib/squid/ssl_crtd -s /var/lib/ssl_db -M 4MB
ssl_bump bump all

NOTA: En comptes de *ssl_bump bump* all es podria escriure *ssl_bump terminate all* (tanca totes les connexions HTTPS) o bé *ssl_bump splice all* (es comporta com a simple "entroncador" HTTPS sense cap manipulació en la comunicació) o altres

NOTA: En el cas de voler deshabilitar el "SSL-Bump" per determinades destinacions (per problemes de funcionalitat o per qüestions ètiques), es pot fer així (en aquest cas no es fa SSL-Bump a tots els dominis *.hotmail.com i *.dropbox.com). ¡Atenció a l'ordre de les línies!:

```
acl hola dstdomain .dropbox.com .hotmail.com
ssl_bump none hola
## Ahora va el resto de reglas ssl-bump
## ssl_bump bump all
## etc
```

NOTA: Una altra directiva interessant és *sslproxy_cert_error {allow|deny] all* ... la qual serveix per establir si s'accepten o no certificats autosignats del servidor remot: si és que sí (allow), es passarà al client per a què aquest decideixi; si és que sí (deny), es talla la connexió.

4.-Opcional. Caldria instal.lar el certificat PEM creat al primer pas als navegadors de totes les màquines de la nostra LAN per tal de no obtenir l'error de "Conexió no segura" que apareix en navegar quan es detecta un certificat no reconegut (per ser autosignat). És important instal.lar aquest certificat com a de tipus "CA certificat" per a què l'error no surti cada vegada que es vol navegar a un domini diferent. En el cas del Firefox això es faria anant a "Preferències"->"Privadesa i seguretat"->"Certificats"->Veure" i sota la pestanya "Entitats" utilitzar el botó "Import".

Analitzadors de logs de l´Squid i generadors d´estadístiques:

L´ arxiu access.log té les peticions dels clients, l´activitat, les consultes HTTP i ICMP, les IPs dels clients, les URLs demanades...Consultar a pèl aquest arxiu és una tasca quasi impossible, i sobretot molt farragosa. Per això existeixen aplicacions que llegeixen aquest arxiu i generen gràfiques de diferents tipus (peticions acceptades i denegades, horaris d´ús, etc, etc). Per trobar la majoria d´ells, es pot anar a la web oficial de l´Squid, http://www.squid-cache.org/Misc/log-analysis.html .

També val la pena comentar que existeixen les eines CacheManager i el paquet “squidclient”,tots dos oficials d'Squid, (https://wiki.squid-cache.org/Features/CacheManager) les quals són eines permeten obtenir dades internes sobre seu funcionament i comportament. Un altre sistema alternatiu és fer servir el protocol SNMP (https://wiki.squid-cache.org/Features/Snmp)

Si consultem el format per defecte de l´arxiu access.log, podem deduir els camps més importants:

-Camp 1: Marca de temps de la petició (segons el format de temps Unix)
-Camp 2: Temps que triga l'Squid en donar servei a una petició (en milisegons)
-Camp 3: IP del client
-Camp 4: Si el seu valor és "TCP_MISS/...", vol dir que la petició no ha ha sigut trobada a la catxé . El valor de "HIT" vol dir que la pàgina ha sigut trobada a la catxé perquè ha sigut prèviament emmagatzemada. El valor de “TCP_DENIED” denota una denegació d´accés a aquesta pàgina per part d ´alguna regla http_access.
-Camp 5: Tamany de la pàgina retornada al client
-Camp 6 : Mètode HTTP emprat per la petició (GET, HEAD, etc)
-Camp 7: Url demanada
-Camp 8: Usuari autentificat per l'Squid (si n'hi ha)
-Camp 9: IP del servidor de destí. si el proxy conecta directament al servidor de destí sense més proxies intermediaris pel mig, afegeix la paraula "DIRECT"
-Camp 10: Tipus mime del recurs demanat pel client

Si es vol reenviar la informació de log a un altre destí (com ara el journalctl, via UDP o TCP a un altre ordinador remot, etc) es pot consultar com fer-ho a https://wiki.squid-cache.org/Features/LogModules

---