Squid + SquidGuard + Privoxy

Un problema molto sentito sia dalle aziende che dalle famiglie è il filtraggio dei siti e dei contenuti accessibili tramite il protocollo http con un browser di navigazione web. Esistono sul mercato ormai molti prodotti specializzati che offrono soluzioni di vario tipo a questo problema, prodotti sia per la grande azienda che per la famiglia, ed add-on con queste funzioni sono ormai presenti in molti firewall anche di tipo SOHO. In questo e nel prossimo articolo descriviamo una semplice soluzione basata sul ben noto proxy squid, che pertanto richiede una se pur piccola rete interna. Questa è comunque una soluzione abbastanza scalabile, da reti domestiche (sempre che vi sia qualcuno in grado di gestirla) ad aziende medie o grandi. La soluzione si basa sulla presenza di squid come proxy-cache e di due programmi che possiamo considerare suoi add-on, SquidGuard e Privoxy. Questa soluzione è facilmente implementabile su una macchina con OS UNIX/Linux dedicata a questo scopo. Descriviamo prima la configurazione di squid e SquidGuard e passeremo poi a Privoxy. Nel descrivere la configurazione di questa soluzione toccheremo molte delle problematiche comuni a quasi tutte le soluzioni di questo tipo.

Squid

L'installazione di squid segue le usuali procedure per programmi su OS UNIX/Linux e per molte OS/distribuzioni sono disponibili pacchetti pre-compilati. Non possiamo qui descrivere la compilazione ed installazione di squid per la quale facciamo riferimento alla abbondante documentazione. Squid è comunque un programma ragionevolmente grande e complesso il cui scopo principale è quello di funzionare da proxy-cache e gestire i protocolli http, https e ftp. Come ogni proxy, squid riceve le richieste da parte dei client svolgendo le funzioni di server, e se i dati non sono presenti nella sua cache, forwarda le richieste ai veri server verso i quali si comporta come se fosse un client. La configurazione di squid non è semplicissima, per rendersene conto basta guardare le dimensioni del file di configurazione, di solito in `/etc/squid/squid.conf`. In tabella 1 sono riportati alcuni importanti, almeno dal punto di vista della sicurezza, parametri di configurazione di squid, ma una completa configurazione richiede molto più lavoro. Abbiamo configurato squid ad ascoltare le richieste dei client sulla porta 3128 solo sull'indirizzo interno della macchina. Le ACL indicate restringono l'accesso solo alle macchine sulla rete locale (192.168.1.0/24) e per i servizi relativi alle porte 21, 80, 443 e 8080. Il (pericoloso) metodo CONNECT è limitato al solo https sulla porta 443. Le configurazioni per il servizio di acceleratore (`httpd_accel_*`) sono ovviamente opzionali e permettono a squid di funzionare da proxy trasparente sulla porta 80, ovvero la richiesta di un client al server finale può essere ridiretta da un

router o firewall a squid, il quale con questa configurazione la soddisfa senza che il client si renda conto di essere passato attraverso un proxy. Ad esempio, in Linux se squid è sulla stessa macchina che funge da firewall, la seguente regola è sufficiente a ridirigere il traffico per i client che non hanno configurato il proxy:

```
iptables -t nat -A PREROUTING -i eth0 -p tcp --dport 80 \
   -j REDIRECT --to-port 3128
```

Ancora opzionalmente possiamo cambiare l'User-Agent offerto dai browser dei client uniformandoli ad uno particolare. Questa operazione permette di nascondere l'identità delle macchine interne, ma rischia di rendere difficile la navigazione su siti che offrono versioni specializzate delle proprie pagine a seconda del browser che le richiede.

Il problema che volevamo risolvere è però quello di filtrare i siti e le pagine raggiungibili dai nostri client. Squid non offre direttamente alcuna possibilità di eseguire questi filtri. Visto però che tutto il traffico passa attraverso di lui, è ovvio che squid è il punto migliore ove implementare questi filtri. Dividiamo il problema in due parti: il filtraggio degli indirizzi, ovvero le URL, ed il filtraggio dei contenuti delle pagine. Affrontiamo il primo problema con SquidGuard, ed il secondo con Privoxy.

SquidGuard

Squid offre la possibilità di aggiungere un programma esterno detto *Redirector* il cui scopo è quello di modificare le URL provenienti dai client prima che squid richieda la pagina al server finale. E' possibile utilizzare questo servizio per i nostri scopi come segue: squid passa al redirector, in questo caso SquidGuard, tutte le URL di ogni pagina richiesta; SquidGuard controlla le URL rispetto al proprio database ed alla propria configurazione e se la URL è vietata la sostituisce con la URL di una pagina di errore preparata dall'amministratore. Il client che cerca di accedere ad una pagina vietata viene perciò ridiretto ad una pagina di errore.

Come al solito, compilazione ed installazione di SquidGuard sono semplici, e per molti OS vi sono i soliti pacchetti precompilati. La configurazione di SquidGuard è invece un poco più complessa. Prima di tutto, squid deve attivare SquidGuard e questo è fatto con le seguenti istruzioni nel file di configurazione di squid:

```
redirect_program /usr/bin/squidGuard -c /etc/squid/squidguard.conf
redirect_children 5
```

(il numero di children dipende ovviamente dal numero di client ed il traffico che svolgono). La configurazione di SquidGuard richiede sia di modificare il file di configurazione, di solito `/etc/squid/squidguard.conf`, sia di creare il database di indirizzi vietati e/o permessi. In

Tabella 2 riportiamo un semplicissimo file di configurazione. In questo file specifichiamo dove si trova il database e dove vanno scritti i file di log. Poi sono definiti dei gruppi di destinazioni, ad ogni gruppo corrispondono 3 file del database di tipo diverso: i file di `domini`, quelli di `URL` e quelli di `regular-expression`. Esempi di questi file sono nelle Tabelle 3, 4 e 5. Se è specificato un logfile, tutte le volte che c'è un match di un gruppo questo verrà riportato nel logfile. Le regole da applicare sono specificate nella ACL. la regola `pass` dice che devono essere accettate così come sono, ovvero non modificate, le URL che appartengono al gruppo *good*, e tutte quelle che non appartengono ai gruppi *bad*, *adult* e *aggressive*. Le URL che non soddisfano questa regola, ovvero quelle che appartengono ai gruppi *bad*, *adult* e *aggressive* e non appartengono al gruppo *good*, vengono sostituite dalla URL descritta nella regola `redirect`. Il file di configurazione di SquidGuard permette di fare molto di più: ad esempio definire delle regole attive solo in determinati orari, applicare delle regole solo a certi indirizzi IP sorgente o domini o utenti, ovvero solamente a certi client, e così via.
Vi sono database pubblici, anche già pacchettizzati per varie distribuzioni, di URL di siti o pagine che possono essere non gradite, di siti o pagine con eccessiva pubblicità, o siti o pagine con contenuti illeciti, o non accessibili ai minori ecc. Il problema comune a tutte le blacklist è che da una parte queste non sono complete e vanno aggiornate frequentemente, e dall'altra spesso bloccano l'accesso a pagine invece del tutto lecite. Nella configurazione indicata, i primi gruppi definiti nel file di configurazione corrispondono a database pubblici, mentre i gruppi *bad* e *good* sono locali. Utilizzando questi due gruppi (si noti come il gruppo *good* è il primo nella regola `pass`) possiamo sia aggiungere blocchi per destinazioni non presenti nelle liste generiche, che creare delle eccezioni. Bisogna fare comunque attenzione che se una URL è bloccata da una regola ad esempio di tipo `regular-expression`, bisogna creare una regola dello stesso tipo nel gruppo *good* per permetterla. Per maggiore velocità di esecuzione è anche possibile convertire i file di tipo domini e URL in formato DB con il comando `squidguard -C /fullpath/file` . Ogni volta che si modifica un file nel database bisogna aggiornare il processo in esecuzione di SquidGuard, per fare questo basta inviare un segnale di HUP a squid, che forza sia squid che SquidGuard a ricaricare le proprie configurazioni.

Prima di procedere con la configurazione di Privoxy, è necessario riassumere velocemente la struttura del nostro piccolo progetto. Abbiamo sinora parlato di squid + SquidGuard, in particolare il ruolo di SquidGuard è quello di filtrare le richieste dei client, ovvero le URL che i client passano a squid. Questo processo è relativamente semplice, anche perché le richieste dei client sono

composte solamente dagli indirizzi delle pagine cercate, e quindi sono dati molto piccoli e con una struttura molto semplice e limitata. Filtrare le richieste dei client non è però sufficiente, dobbiamo anche filtrare i dati inviati dai server, ovvero le pagine web di ritorno dai server ai client, e questo è il ruolo di Privoxy. La Figura 1 riporta un semplice diagramma del traffico che illustra il flusso dei dati ed i due filtraggi distinti. E' chiaro che il lavoro di Privoxy è molto maggiore di quello di SquidGuard, visto che i dati da analizzare sono molti di più. E' anche chiaro che il lavoro di Privoxy deve essere molto simile a quello di un anti-virus o di un Intrusion Detection. Deve infatti analizzare tutti i dati (le pagine web) e trovare stringhe pericolose o non permesse basandosi su impronte note, ed eliminare i contenuti non permessi dalle pagine. Una conseguenza di ciò è che mentre l'aggiunta di SquidGuard a squid di norma non modifica sensibilmente i tempi di download delle pagine web, l'introduzione di Privoxy può aggiungere dei ritardi alle volte sensibili (dipende anche dall'Hardware utilizzato).

Privoxy

Privoxy è un proxy server web indipendente, non caching, che nella nostra architettura mettiamo tra squid ed i server web finali. Per squid, Privoxy è configurato come *Parent Proxy* diretto, solo per i protocolli http e https. In Tabella 6 indichiamo le aggiunte alla configurazione di squid per indirizzare tutto il traffico tra squid ed internet attraverso Privoxy che assumiamo essere installato sulla stessa macchina. Una volta compilato ed installato, la configurazione di Privoxy è possibile sia via interfaccia Web che tramite file di configurazione. L'accesso via web è di norma possibile solo sulla stessa macchina su cui è installato: basta configurare il proprio browser con `HTTP_PROXY 127.0.0.1:8118` ed indirizzare il browser alla pagina http://p.p . In questa sede descriviamo invece la configurazione manuale escludendo quella via web. Vi sono vari file di configurazione da modificare. Il primo è di solito `/etc/privoxy/config` ed in Tabella 7 riportiamo le principali modifiche che abbiamo apportato. In particolare in questo file definiamo gli altri 4 file di configurazione che useremo, l'indirizzo e la porta sulla quale Privoxy riceve le richieste da squid, escludiamo la possibilità di disabilitazione via web, e fissiamo il limite superiore di dimensione alle pagine da esaminare, di solito 4MB. Se una pagina è più grande di questo limite, la pagina viene inviata al client senza essere verificata. Il problema è che quando Privoxy esamina una pagina, la carica tutta in RAM (e più pagine possono essere esaminate in parallelo), questo richiede ovviamente di imporre dei limiti per evitare peggiori conseguenze.

L'anima di Privoxy è però negli altri file di configurazione, i quali sono di due tipi: gli *Actions* file ed il *Filter* file. Di solito gli Actions file `standard`, `default` ed il Filter file `default.filter` non richiedono di essere modificati, mentre l'Actions file `user` è quello che

deve essere adattato alle proprie esigenze. Cominciamo a descrivere il Filter file, di cui un piccolo esempio è riportato in Tabella 9. La sintassi di questo file è molto semplice: dopo la keyword `FILTER` segue il nome del filtro e la sua descrizione. Nelle righe seguenti sono definite le azioni del filtro in un linguaggio che prende dal programma *sed* con regole di sostituzione e regular-expression simili a quelle di *perl*. Nell'esempio in Tabella 9 tutte le volte che in una pagina viene trovato un oggetto `shockwave-flash` questo viene sostituito con il commento `<!-- Squished Shockwave Flash Object -->` in questa maniera il cliente finale non riceve il contenuto non permesso, ma riceve tutto il resto della pagina. Ovviamente, oltre ai filtri presenti di default alcuni dei quali sono elencati in Tabella 10, possiamo facilmente creare nuovi filtri per bloccare contenuti che non vogliamo arrivino ai nostri client.

Nei file di tipo Actions definiamo quali azioni, tra cui anche i filtri appena descritti, applicare a quali siti e quali pagine. Le azioni non si limitano ai filtri, ma includono tra gli altri anche la possibilità di bloccare o modificare i cookies, bloccare o modificare le immagini, bloccare del tutto l'accesso al sito o pagina (ma questo lo abbiamo implementato con SquidGuard) e così via. In Tabella 8 riportiamo un esempio di file Actions. Il formato è semplice: tra parentesi graffe c'è il nome dell'azione da applicare, se preceduta da +, oppure da non applicare, se preceduta da -. Nelle righe seguenti vi è la lista di URL, come al solito in formato regular-expression, a cui applicare o meno l'azione. La possibilità di non applicare azioni permette di introdurre eccezioni a regole generali, ad esempio da mettere nel file di configurazione locale `user`. Nelle prime due righe della Tabella 8 sono definiti due alias che permettono di raggruppare più azioni e di applicarle insieme ad una lista di URL.

La configurazione di Privoxy è abbastanza potente, e le regular-expression ci permettono di costruire filtri mirati a bloccare i contenuti che non desideriamo arrivino ai nostri client. La configurazione di default di Privoxy contiene molti filtri ed azioni predefiniti, oltre ad un numeroso elenco di URL che possiamo essere interessati a filtrare per vari motivi. In particolare, la configurazione di default di Privoxy permette di filtrare molta della pubblicità, sia come immagini che come pop-ups, che ormai affligge gran parte della navigazione web. Sta ovviamente a chi gestisce il sistema ed ha la responsabilità, decidere cosa è appropriato filtrare, anche nel rispetto delle leggi in vigore sulla Privacy.

Andrea Pasquinucci
Libero Professionista in Sicurezza Informatica
pasquinucci@ucci.it

Riferimenti Bibliografici

[1] http://www.squid-cache.org/

[2] http://www.merlinobbs.net/Squid-Book/HTML/index.html

[3] http://www.squidguard.org/

[4] SquidGuard e suoi database in formato rpm sono ad esempio reperibili su
http://dag.wieers.com/packages/squidguard/,
http://dag.wieers.com/packages/squidguard-blacklists/

[5] http://www.privoxy.org/

```
http_port 192.168.1.1:3128
ftp_user us@us.com
cache_mgr admin@domain.it

acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
acl SSL_ports port 443
acl Safe_ports port 21 80 443 8080 # ftp http https
#acl Safe_ports port 1025-65535  # unregistered ports
acl CONNECT method CONNECT
acl local_net src 192.168.1.0/255.255.255.0

http_access allow manager localhost
http_access deny manager
http_access deny !Safe_ports
http_access deny CONNECT !SSL_ports
http_access deny to_localhost
http_access allow local_net
http_access allow localhost
http_access deny all

httpd_accel_host virtual
httpd_accel_port 80
httpd_accel_with_proxy on
httpd_accel_uses_host_header on

forwarded_for off

header_access User-Agent deny all
header_access Via deny all
header_replace User-Agent Mozilla/5.0 (Windows XP; U)
```

Tabella 1. Parte del file di configurazione di Squid

```
### Path configuration
dbhome /var/lib/squidguard
logdir /var/log/squidguard

### Group 'adult' containing entries for 'adult,porn'
dest adult {
        logfile adult.log
        domainlist      adult/domains
        urllist         adult/urls
        expressionlist  adult/expressions
}

[ ... other groups ... ]

### Define your local blacklists here
dest bad {
        logfile localbad.log
        domainlist      local/bad/domains
        urllist         local/bad/urls
        expressionlist  local/bad/expressions
}

dest good {
        domainlist      local/good/domains
        urllist         local/good/urls
        expressionlist  local/good/expressions
}

### ACL definition
acl {
     default {
               pass good !bad !adult !aggressive any
               redirect 302:http://www.ucci.it/forbidden.html?\
                      clientaddr=%a&clientname=%n&clientident=%i&\
                      srcclass=%s&targetclass=%t&url=%u
     }
}
```

Tabella 2. File di configurazione di SquidGuard

```
# /var/lib/squidguard/local/good/domains
www.ucci.it
www.ictsecurity.it
```

Tabella 3. Domains file

```
# /var/lib/squidguard/local/bad/expressions
\.(ra?m|wma|mp2|mpv2|mp3|asx)($|\?)
\.(mpe?g?|wmv|mov|movie|qt|avi|dvd?|divx)($|\?)
```

Tabella 4. Expressions file

```
# /var/lib/squidguard/ads/urls
www.ucci.it/Pubblicita
www.ictsecurity.it/graphics/Advertisements
```

Tabella 5. URLs file

```
# Define Privoxy as parent proxy (without ICP)
cache_peer 127.0.0.1 parent 8118 7 no-query
# Define ACL for protocol FTP
acl ftp proto FTP
# Do not forward FTP requests to Privoxy
always_direct allow ftp
# Forward all the rest to Privoxy
never_direct allow all
```

Tabella 6. Aggiunta alla configurazione di Squid

```
actionsfile standard  # Internal purpose, recommended
actionsfile default   # Main actions file
actionsfile user      # User customizations
filterfile default.filter
Listen-address  127.0.0.1:8118
enable-remote-toggle 0 # cannot disbale privoxy with web interface
enable-edit-actions 0 # web interface read-only
buffer-limit 4096     # 4MB, larger pages are not checked
```

Tabella 7. Principali elementi del Config file di Privoxy

```
allow-all-cookies = -crunch-all-cookies -session-cookies-only
+block-as-image = +block +handle-as-image


{ +handle-as-image }
/.*\.(gif|jpe?g|png|bmp|ico)$

{ +block-as-image }
ar.atwola.com
.ad.doubleclick.net
.ad.*.doubleclick.net
.a.yimg.com/(?:(?!/i/).)*$
.a[0-9].yimg.com/(?:(?!/i/).)*$

{ allow-all-cookies }
sourceforge.net
sunsolve.sun.com
.slashdot.org
.yahoo.com
.msdn.microsoft.com
.redhat.com

# Your bank is allergic to some filter, but you don't know which,
# so you disable them all:
{ -filter }
.your-home-banking-site.com
```

Tabella 8. Il formato di un Actions file

```
FILTER: shockwave-flash Kill embedded Shockwave Flash objects

s|<object [^>]*macromedia.*</object>|<!-- Squished Shockwave \
   Object -->|sigU
s|<embed [^>]*(application/x-shockwave-flash\|\.swf).*>\
   (.*</embed>)?|<!-- Squished Shockwave Flash Embed -->|sigU
```

Tabella 9. Il formato del Filter file

```
js-annoyances:  Get rid of particularly annoying JavaScript abuse
js-events: Kill all JS event bindings (Radically destructive!)
html-annoyances: Get rid of particularly annoying HTML abuse
content-cookies: Kill cookies that come in the HTML or JS content
refresh-tags: Kill automatic refresh tags (for dial-on-demand)
unsolicited-popups: Disable only unsolicited pop-up windows
all-popups: Kill all popups in JavaScript and HTML
img-reorder: Reorder attributes in <img> tags
banners-by-size: Kill banners by size (very efficient!)
banners-by-link: Kill banners by their links to known clicktrackers
webbugs: Squish WebBugs (1x1 invisible GIFs used for user tracking)
tiny-textforms: Extend tiny textareas to 40x80 and kill the wrap
jumping-windows: Prevent windows from resizing and moving
frameset-borders: Give frames a border
demoronizer: Fix MS's non-standard use of standard charsets
shockwave-flash: Kill embedded Shockwave Flash objects
quicktime-kioskmode: Make Quicktime movies saveable
fun: Text replacements  for subversive browsing fun!
crude-parental: Kill pages that contain the words "sex" or "warez"
```

Tabella 10. Alcuni Filtri predefiniti

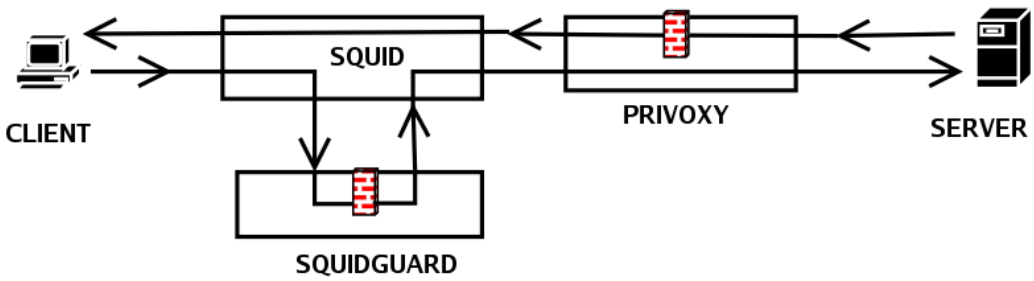

Figura 1. Diagramma di traffico