Implementing a home proxy server with Squid

SAFE HARBOR

A proxy server provides safer and more efficient surfing. Although commercial proxy solutions are available, all you really need is Linux and an old PC in the attic.

BY GEERT VAN PAMEL

I have had a home network for several years. I started with a router using Windows XP with ICS (Internet Connection Sharing) and one multi-homed Ethernet card. The main disadvantages were instability, low performance, and a total lack of security. Troubleshooting was totally impossible. Firewall configuration was at the mercy of inexperienced users, who clicked randomly at security settings as if they were playing Russian roulette.

I finally turned to Linux and set up an iptables firewall on a Pentium II computer acting as a router. The firewall system would keep the attackers off my network and log incoming and outgoing traffic. Along with the iptables firewall, I also set up a Squid proxy server to improve Internet performance, filter out unwanted popup ads, and block dangerous URLs.

A Squid proxy server filters Web traffic and caches frequently accessed files. A proxy server limits Internet bandwidth usage, speeds up Web access, and lets you filter URLs. Centrally blocking advertisements and dangerous downloads is cost effective and transparent for the end user.

Squid is a high performance implementation of a free Open-Source, full-featured proxy caching server. Squid provides extensive access controls and integrates easily with an iptables firewall. In my case, the Squid proxy server and the iptables firewall worked together to protect my network from intruders and dangerous HTML. You'll find many useful discussions of firewalls in books, magazines, and Websites. (See [1] and [2], for example.) The Squid proxy server, on the other hand, is not as

Table 1: Recommended Hardware

Necessary Components	Specifics
Intel Pentium II CPU, or higher - Why not a spare Alpha Server?	350 MHz
80 - 100 MB memory minimum	more is better
1 or more IDE disks (reuse 2 old disks: 1 GB system SW + swap & 3 GB for cache + /home disk)	4 GB minimum
2 Ethernet cards, minihub, fast Ethernet modem, wireless router or hub	100 Mbit/s if possible
CDROM, DVD reader	software is mostly distributed via DVD
Use only normal straight LAN cables [no need for cross cables]	modem and minihub cross themselves!

well documented, especially for small home networks like mine. In this article, I will show you how to set up Squid.

Getting Started

The first step is to find the necessary hardware. Figure 1 depicts the network configuration of the Pentium II computer I used as a firewall and proxy server. This firewall system should operate with minimal human intervention, so after the system is configured, you'll want to disconnect the mouse, keyboard, and video screen. You may need to adjust the BIOS settings so that the computer will boot without a keyboard. The goal is to be able to put the whole system in the attic, where you won't hear it or trip over it. From the minihub shown in Figure 1, you can come "downstairs" to the home network using standard UTP cable or a wireless connection. Table 1 shows recommended hardware for the firewall machine.

Assuming your firewall is working, the next step is to set up Squid. Squid is available from the Internet at [3] or one of its mirrors [4] as *tar.gz* (compile from sources). You can easily install it using one of the following commands:

```
rpm -i /cdrom/RedHat/RPMS/➥
squid-2.4.STABLE7-4.i386.rpm➥
# Red Hat 8

rpm -i /cdrom/Fedora/RPMS/➥
squid-2.5.STABLE6-3.i386.rpm ➥
# Fedora Core 3

rpm -i /cdrom/.../➥
squid-2.5.STABLE6-6.i586.rpm➥
 # SuSE 9.2
```

At this writing, the current stable Squid version is 2.5.

Configuring Squid

Once Squid is installed, you'll need to configure it. Squid has one central configuration file. Every time this file changes, the configuration must be reloaded with the command */sbin/init.d/squid reload*.

You can edit the configuration file with a text editor. You'll find a detailed description of the settings inside the *squid.conf* file, although the discussion is sometimes very technical and difficult to understand. This section summarizes some of the important settings in the *squid.conf* file.

First of all, you can prevent certain metadata related to your configuration from reaching the external world when you surf the Web:

```
vi /etc/squid/squid.conf
...
anonymize_headers deny ➥
From Server Via User-Agent
forwarded_for off
strip_query_terms on
```

Note that you cannot anonymize *Referer* and *WWW-Authenticate* because otherwise authentication and access control mechanisms won't work.

forwarded_for off means that the IP address of the proxy server will not be sent externally.

With *strip_query_terms on*, you do not log URL parameters after the ?. When this parameter is set to *off*, the full URL is logged in the Squid log files. This feature can help with debugging the Squid filters, but it can also violate privacy rules.

The next settings identify the Squid host, the (internal) domain where the machine is operating, and the username of whoever is responsible for the server. Note the dot in front of the domain. Further on, you find the name of the local DNS caching server, and the number of domain names to cache into the Squid server.

```
visible_hostname squid
append_domain .mshome.net
```

```
cache_mgr sysman

dns_nameservers 192.168.0.1
dns_testnames router.mshome.net
fqdncache_size 1024

http_port 80
icp_port 0
```

http_port is the port used by the proxy server. You can choose anything, as long as the configuration does not conflict with other ports on your router. A common choice is 8080 or 80. The Squid default, 3128, is difficult to remember.

We are not using *cp_port*, so we set it to 0. This setting synchronizes proxy servers.

With *log_mime_hdrs on*, you can make mime headers visible in the *access.log* file.

Avoid Disk Contention

Squid needs to store its cache somewhere on the hard disk. The cache is a tree of directories. With the *cache_dir* option in the *squid.conf* file, you can specify configuration settings such as the following:

- disk I/O mechanism – *aufs*
- location of the squid cache on the disk – */var/cache/squid*
- amount of disk space that can be used by the proxy server – *2.5 GB*
- number of main directories – *16*
- subdirectories – *256*

For instance:

```
cache_dir aufs ➥
/var/cache/squid 2500 16 256
```

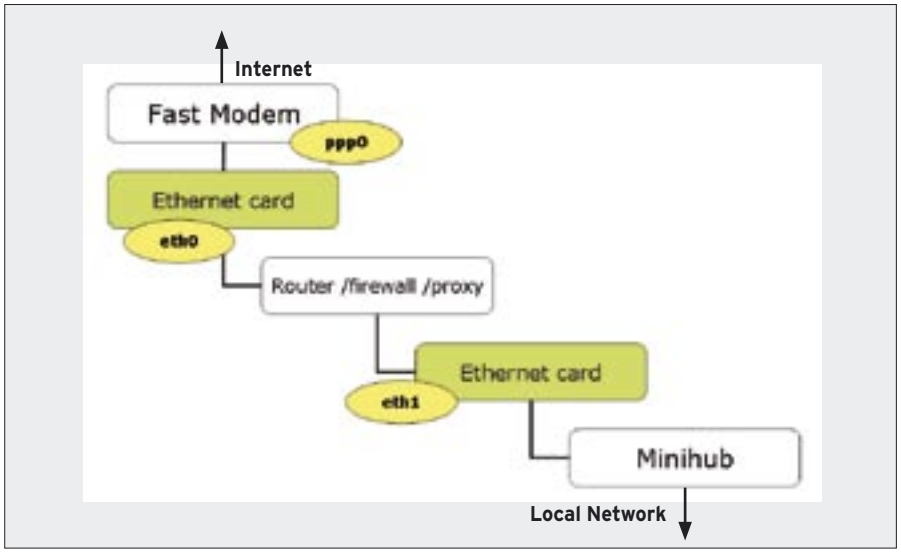

Figure 1: Ethernet basic LAN configuration.

The disk access method options are as follows:

- *ufs* – classic disk access (too much I/O can slow down the Squid server)
- *aufs* – asynchronous UFS with threads, less risk of disk contention
- *diskd* – diskd daemon, avoiding disk contention but using more memory

UFS is the classic UNIX file system I/O. We recommend using *aufs* to avoid I/O bottlenecks. (When you use *aufs*, you have fewer processes.)

```
# ls -ld /var/cache/squid
lrwxrwxrwx    1 root     root➥
 19 Nov 22 00:42 ➥
/var/cache/squid -> ➥
/volset/cache/squid
```

I suggest you keep the standard file location for the squid cache */var/cache/squid*, then create a symbolic link to the real cache directory. If you move the cache to another disk for performance or capacity reasons, you only have to modify the symbolic link.

The disk space is distributed among all directories. You would normally look for even distribution across all directories, but in practice, some variation in the distribution is acceptable. More complex setups using multiple disks are possible, but for home use, one directory structure is sufficient.

Cache Replacement

The proxy server uses an LRU (Least Recently Used) algorithm. Detailed studies by HP Laboratories [6] have revealed that an LRU algorithm is not always an intelligent choice. The *GDSF* setting keeps small popular objects in cache, while removing bigger and lesser used objects, thus increasing the overall efficiency.

```
cache_replacement_policy➥
 heap GDSF
memory_replacement_policy➥
 heap GDSF
```

Big objects requested only once can flush out a lot of smaller objects, therefore you'd better limit the maximum object size for the cache:

```
cache_mem 20 MB
maximum_object_size➥
 16384 KB
maximum_object_size➥
_in_memory 2048 KB
```

Log Format Specification

You can choose between Squid log format and standard web server log format using the parameter *emulate_httpd_log*. When the parameter is set to *on*, standard web log format is used; if the parameter is set to *off*, you get more details with the Squid format. See [7] for more on analyzing Squid log files.

Proxy Hierarchy

The Squid proxy can work in a hierarchical way. If you want to avoid the parent proxy for some destinations, you can allow a direct lookup. The browser will still use your local proxy!

```
acl direct-domain ➥
dstdomain .turboline.be
always_direct allow ➥
direct-domain

acl direct-path urlpath_regex➥
 -i "/etc/squid/direct-path.reg"
always_direct allow direct-path
```

Some ISPs allow you to use their proxy server to visit their own pages even if you are not a customer. This can help you speed up your visits to their pages. The closer the proxy to the original pages, the more likely the page is to be cached. Because your own ISP is more remote, the ISP is less likely to be caching its competitor's contents…

Table 2: ACL Guidelines

- the order of the rules is important
- first list all the deny rules
- the first matching rule is executed
- the rest of the rules are ignored
- the last rule should be an allow all

```
cache_peer  proxy.tiscali.be➥
 parent  3128 3130 ➥
no-query default
cache_peer_domain ➥
proxy.tiscali.be .tiscali.be
```

no-query means that you do not use, or cannot use, ICP (the Internet Caching Protocol), see [8]. You can obtain the same functionality using regular expressions, but this gives you more freedom.

```
cache_peer  proxy.tiscali.be➥
 parent  3128 3130 ➥
no-query default
acl tiscali-proxy ➥
dstdom_regex -i ➥
\.tiscali\.be$
cache_peer_access ➥
proxy.tiscali.be allow ➥
tiscali-proxy
```

Listing 1: Blocking Unwanted Pages

```
01 acl block-ip dst "/etc/squid/block-ip.reg"
02 deny_info filter_spam block-ip
03 http_access deny block-ip
04
05 acl block-hosts dstdom_regex -i "/etc/squid/block-hosts.reg"
06 deny_info filter_spam block-hosts
07 http_access deny block-hosts
08
09 acl noblock-url url_regex -i "/etc/squid/noblock-url.reg"
10 http_access allow noblock-url Safe_ports
11
12 acl block-path urlpath_regex -i "/etc/squid/block-path.reg"
13 deny_info filter_spam block-path
14 http_access deny block-path
15
16 acl block-url url_regex -i "/etc/squid/block-url.reg"
17 deny_info filter_spam block-url
18 http_access deny block-url
```

Listing 2: Making a Page Invisible

```
01 vi /etc/squid/errors/filter_spam
02 ...
03 <script language="JavaScript"
   type="text/javascript">
04  <!--
05  window.status="Filter " +
   document.location; //.pathname;
06  // -->
07 </script>
08 <noscript><plaintext><!--
```

The ACL could also include a regular expression (regex for short) with the URL using an *url_regex* construct.

For Squid, regular expressions can be specified immediately, or they can be in a file name between double quotes, in which case the file should contain one regex expression per line – no empty lines. The *-i* (ignore case) means that case-insensitive comparisons are used.

If you are configuring a system with multiple proxies, you can specify a round-robin to speed up page lookups and minimize the delay when one of the servers is not available. Remember that most browsers issue parallel connections when obtaining all the elements from a single page. If you use multiple proxy servers to obtain these elements, your response time might be better.

```
cache_peer  80.200.248.199  ➥
parent  8080 7 ➥
no-query round-robin
cache_peer  80.200.248.200  ➥
parent  8080 7 ➥
no-query round-robin
...
cache_peer  80.200.248.207➥
 parent  8080 7 ➥
no-query round-robin
```

FTP files are normally downloaded just once, so will not normally want to cache them, except when downloading repeatedly. Also, local pages are not normally cached, since they already reside on your network :

```
acl FTP proto FTP
always_direct allow FTP

acl local-domain dstdomain ➥
.mshome.net
always_direct allow ➥
local-domain

acl localnet-dst dst ➥
192.168.0.0/24
always_direct allow ➥
localnet-dst
```

Filtering with Squid

The preceding sections introduced some important Squid configuration settings. You have already learned earlier in this article that ACLs (Access Control Lists) can be used for allowing direct access to pages without using the parent proxy. In this section, I'll show you how to use ACLs for more fine-grained access control.

Table 2 provides some guidelines for creating ACL lists. It is a very good idea to only allow what-you-see-is-what-you-get (WYSIWYG) surfing. If you do not want to see certain pages or frames, then you can automatically block the corresponding URLs for those pages on the proxy server.

You can filter on:

- domains of client or server
- IP subnets of client or server
- URL path
- Full URL including parameters
- keywords
- ports
- protocols: HTTP, FTP
- methods: GET, POST, HEAD, CONNECT
- day & hour
- browser type
- username

Listing 1 shows examples of commands that block unwanted pages.

The script in Listing 2 will make unwanted pages invisible:

Whenever Squid executes the *deny_info* tag, it sends the file */etc/squid/errors/filter_spam* to the browser instead of the real Web page… effectively filtering away the unwanted object. The trailing *<!--* hides any other Squid error messages in the body of the text.

Squid allows you to block content by IP subnet. For instance, you can block sites with explicit sexual content; you could use whois [9] to help you identify the subnets, then enter the subnets in the */etc/squid/block-ip.reg* file:

```
vi /etc/squid/block-ip.reg
...
64.255.160.0/19
64.57.64.0/19
64.7.192.0/19
66.115.128.0/18
66.152.64.0/19
66.230.128.0/18
```

To block advertisements or sex sites by domain name, you can list regular expressions describing the sites in the file */etc/squid/block-hosts.reg*, as shown in Listing 3.

It is also a good idea to block certain file types. For instance, you do not want to allow *.exe* files, since these are some-

Listing 3: Blocking by Domain Name

```
01 vi /etc/squid/block-hosts.reg
02 ...
03 ^a\.
04 ^ad\.
05 ^adfarm\.
06 ^ads\.
07 ^ads1\.
08 ^al\.
09 ^as\.
10 \.msads\.net$
11 ^ss\.
12 ^sa\.
13 ^sc\.
14 ^sm6\.
15 ^tracking\.
16 adserver.adtech.de
17
18 \.belstat\.be$
19 \.doubleclick\.net$
20 \.insites\.be$
21 ^metrics\.
22 \.metriweb\...$
23 \.metriweb\....$
24
25 \.playboy\.com$
26 \.hln\.be$
27 side6
28 www.whitehouse.com
```

Listing 4: Blocking by Path or Extension

```
01 vi /etc/squid/block-path.reg
02 ...
03 \.ad[ep](\?.*)?$
04 \.ba[st](\?.*)?$
05 \.chm(\?.*)?$
06 \.cmd(\?.*)?$
07 \.com(\?.*)?$
08 \.cpl(\?.*)?$
09 \.crt(\?.*)?$
10 \.dbx(\?.*)?$
11 \.hlp(\?.*)?$
12 \.hta(\?.*)?$
13 \.in[fs](\?.*)?$
14 \.isp(\?.*)?$
15 \.lnk(\?.*)?$
16 \.md[abetwz](\?.*)?
17 \.ms[cpt](\?.*)?$
18 \.nch(\?.*)?$
19 \.ops(\?.*)?$
20 \.pcd(\?.*)?$
21 \.p[ir]f(\?.*)?$
22 \.reg(\?.*)?$
23 \.sc[frt](\?.*)?$
24 \.sh[bs](\?.*)?$
25 \.url(\?.*)?$
26 \.vb([e])?(\?.*)?$
27 \.vir(\?.*)?$
28 \.wm[sz](\?.*)?$
29 \.ws[cfh](\?.*)?$
```

INFO

[1] Presentation for the HP-Interex user group in Belgium on 17/03/2005 about "Implementing a home Router, Firewall, Proxy server, and DNS Caching Server using Linux" *http://users.belgacombusiness.net/linuxug/pub/router/linux-router-firewall-proxy.zip*

[2] Firewalls: *http://www.linux-magazine.com/issue/40/Checkpoint_FW1_Firewall_Builder.pdf http://www.linux-magazine.com/issue/34/IPtables_Firewalling.pdf*

[3] About Squid in general: *http://www.squid-cache.org http://squid-docs.sourceforge.net/latest/book-full.html#AEN1685 http://www.squid-cache.org/FAQ/FAQ-10.html*

[4] Squid mirror sites: *http://www1.de.squid-cache.org http://www1.fr.squid-cache.org http://www1.nl.squid-cache.org http://www1.uk.squid-cache.org*

[5] Suse 9.2 Professional – DVD software distribution *http://www.linux-magazine.com/issue/54/Linux_Magazine_DVD.pdf*

[6] For more information about the GDSF and LFUDA cache replacement policies see: *http://www.hpl.hp.com/techreports/1999/HPL-1999-69.html http://fog.hpl.external.hp.com/techreports/98/HPL-98-173.html*

[7] Reporting and analysing Squid log files: *http://www.linux-magazine.com/issue/36/Charly_Column.pdf*

[8] ICP – Internet Caching Protocol: *http://en.wikipedia.org/wiki/Internet_Cache_Protocol*

[9] The whois database: *http://www.ripe.net/db/other-whois.html*

[10] About Regular Expressions: *http://www.python.org/doc/current/lib/module-re.html*

[11] Example configuration files for Squid: *http://members.lycos.nl/geertivp/pub/squid*

times executable zip files that install software. Squid lets you block files by path, filename, or file extension, as shown in Listing 4.

Squid also lets you filter for regular expressions used in the URL.

Of course, your filter may occasionally turn up a false positive. You can add regular expressions for URLs you specifically don't want to block to */etc/squid/noblock-url.reg*.

```
vi /etc/squid/noblock-url.reg
...
^http://ads\.com\.com/
```

You can find an up-to-date version of those configuration files at [11]

Protect your Ports

For security reasons, you should disable all ports and only allow well known web ports using the syntax shown in Listing 5.

The same can be done for connected ports. You can allow SSL ports when connected, and deny them otherwise. Remember that the normal HTTP protocol is not connected. The client and the browser always establish a new connection for every page visit.

```
acl SSL_ports port 443 563
acl SSL_ports port 1863 ➥
# Microsoft Messenger
acl SSL_ports port 6346-6353 ➥
 # Limewire

http_access allow ➥
CONNECT SSL_ports
http_access deny ➥
CONNECT
```

Do not allow others to misuse your cache! You only want your cache to be used by your own intranet. Users on the external Internet should not be able to access your cache:

```
acl localhost src ➥
127.0.0.1/255.255.255.255
acl localnet-src src ➥
192.168.0.0/24

http_access deny !localnet-src
```

Allowing All the Rest

To allow only the protocols and the methods that you want:

```
acl allow-proto proto HTTP
http_access deny !allow-proto
```

Listing 5: Protecting Ports

```
01 acl Safe_ports port 80   # http
02 acl Safe_ports port 21   # ftp
03 acl Safe_ports port 2020 # BeOne Radio
04 acl Safe_ports port 2002 # Local server
05 acl Safe_ports port 8044 # Tiscali
06 acl Safe_ports port 8080 # Turboline port scan
07 acl Safe_ports port 8081 # Prentice Hall
08
09 # Deny requests to unknown ports
10 http_access deny !Safe_ports
```

```
acl allow-method ➥
method GET POST
http_access deny ➥
!allow-method
```

The last rule should be an allow-all, since the previous rule was a deny…

```
http_access allow all
```

Remember after changing parameters to always restart the Squid server with the following command:

```
/sbin/init.d/squid reload
```

For SuSE the */sbin/init.d* folder is standard. For Fedora, create a symbolic link:

```
cd /sbin
ln -s /etc/init.d
```

When you have finished the configuration, use *setup* (Fedora), *yast2* (SuSE), or an equivalent tool to activate the Squid service. Remember to reload the server when you change a file.

```
/sbin/init.d/squid reload
```

If anything does not work as expected, you can look for a reason in the cache log file */var/log/squid/cache.log*.

Conclusions

This article is the result of a presentation for the Belgium HP-Interex organization. Slides are available from [1] giving more details about the setup of the iptables firewall, the router, the DNS caching server, the DHCP server, and the NTP server.

If you are looking for better performance, safer surfing, and a way to block access to dangerous Web content, try putting a Squid proxy server in your attic. For the cost conscious among you: a Pentium II router consumes about 11 kWh/week. You should balance this expense against the increased security and reduced headaches of operating your own firewall with Squid proxy caching. ■

Enforce Parental Control and Block Spyware

You should configure your iptables firewall so that it blocks all outgoing HTTP traffic unless the proxy server is used. Since the proxy server is on the local network, it allows all incoming requests from local browser clients.

Any attempt to bypass the Squid filters is blocked by the FORWARD firewall rule blocking HTTP outgoing traffic.

Spyware programs mostly use the HTTP protocol (remember: port 80) for outgoing connections, but it seems that Spyware hardly ever uses the proxy (because Spyware authors are too lazy to inspect the rconfiguration). Spyware is thus blocked by the firewall rules.

THE AUTHOR

Geert Van Pamel has worked as a project manager at *Belgacom* in Belgium since 1997. He has been a member of DECUS since 1985 and a board member of *HP-Interex* since 2002. He learned UNIX on a PDP system in 1982, and he currently works with Linux in a mixed environment with other servers such as Tru64 UNIX, HP-UX, OpenVMS, NonStop Tandem, SUN, and NCR Teradata.