# Ad Zapping With Squid

Cameron Simpson <cs@zip.com.au>



## Licence

Until Wednesday 26may1999, this code was free for use by all. However, the Australian Government brought in some truly stupid and invasive legislation, so this code is now free except that it *MAY NOT* be used to enforce or support that legislation or other legislation of similar intent. I'm happy for people to use it to filter their own browsing, but not for people to force their morals onto others.

## Background

For some time at my workplace we've been running an ad-zapping service on our web proxy. This page documents how it works, how to use it yourself, how to join the mailing list for updates of the pattern file, and the weirdnesses of our local setup (which you need not duplicate yourself).

Ad zapping is not a new idea. Basicly you interpose between the reader and the web some kind of filter which replaces those annoying ad banners with something unobtrusive. (There are a few motivations for this; see this digression for mine.)

I first came across it at my ISP (Zip World - www.zip.com.au) a few years ago. Their technique was to use a complicated proxy.pac file. They supplied two: one which zapped ads and one which didn't. The zapping one was, I discovered, a piece of JavaScript which told your browser to go to one proxy for URLs matching known ad patterns and to the main proxy for everything else. The former proxy simply returned a placeholder GIF for everything asked of it. Initially I copied this for use at our site.

This method is a bit cumbersome. Firstly, you have to run a special web server to serve the placeholder GIF. Secondly, JavaScript interpreters are slow and (in Netscape at least) a tad buggy - eventually the browser gets flakey and may fall over. Thirdly, not all browsers support JavaScript and those that do needn't support proxy.pac files. Finally, the file was a pain to maintain and the size was making me fear for the sanity of the JavaScript interpreters.

Enter squid, arguably the best web proxy around. One great feature is the redirector. This is a program which reads request information on its input and writes (possibly) redirected information on its output. If activiated, squid will consult it for every request, permitting easy interception of ads. All you have to do to activate it is place the line:

```
redirect_program /home/marshall/bin/squid_redirect
```

in your squid.conf file. Obviously, that pathname should be replaced by wherever you install the redirection program.

Attempt number 1 was a shell script. Short and effective, it was a simple while loop with a case statement. However, it seemed to have some scaling problems. Now it is a perl script called squid_redirect. In particular, because the expressions are compiled when the script starts the redirector runs quite efficiently. Looks complicated and hard to maintain, doesn't it? Fear not! It's actually automatically generated from this config file with this update script (updsquid_redirect).

# Installation

1. Install squid.
   Frankly, this is worth doing even for a single user home system (squid's very easy to install, btw). Also, the ad-zapper is very useful when you're connected to the outside world with a modem link.
2. Fetch the software.
   You can either fetch just the script (for the default uncustomised install) or fetch this tarball: adzap-20000928.tar.gz which contains the redirector, a set of the replacement images and a wrapper script for customising the environment for the zapper (if you want to tune its behaviour).
3. Install the redirector in some suitable spot (such as `/usr/local/bin/squid_redirect`).

   *Note 1*: The script must be executable. Run the command:

   ```
   chmod a+rx the-script
   ```

   when it's in place.

   *Note 2*: the first line of the script says:

   ```
   #!/usr/bin/perl
   ```

   You may want to change this to:

   ```
   #!/usr/local/bin/perl
   ```

   or suchlike if your perl isn't in `/usr/bin`. (Or put a symlink in `/usr/bin` - this may save you hassle with other perl scripts, many of which also expect a `/usr/bin/perl`.)

   *Note 3*: If you used a Windows box to fetch the script (eg via Internet Explorer) and then transfered it to the machine running your squid proxy then it's possible for the script to end up on your proxy in DOS text mode, which means it ends every line with a CR and a NL character (instead of just NL).
   If you suspect this, see this troubleshooting section.
4. Insert the line:

   ```
   redirect_program /path/to/squid_redirect
   ```

into the squid.conf file.

5. Send a SIGHUP to your squid:

   ```
   kill -1 pid-of-squid
   ```

   You should also do this after you've updated the script; squid starts new instances of the redirector. Brent J. Nordquist <bjn@visi.com> notes that you can also say:

   ```
   squid -k reconfigure
   ```

   to do the same thing.
6. Want to use a different placeholder image?
   Want to zap more than just ads?
7. **Help me keep the patterns up to date!**
   Just keep half an eye on the zapping.
   If you find a page with an annoying amount of unzapped ads, let me know.
   I will want to know the page itself as well as the ad image so I can sanity check and perhaps optimise or generalise the pattern. (No, I don't care where you browse; fear no censure!) I *am* more interested in zapping large or animated ads; static, small, cache-friendly ads are lower on the priority queue (and perhaps we should consider leaving them alone, to encourage their use).
   If you find pages with content being zapped which should not be, also let me know.
   Just as with the above. Ad zapping is inherently a moving target. Some patterns will match things which are not ads. If people are going to use this facility, I must keep the patterns well tuned.

## Using Different Placeholder Images

The default placeholder GIF is: `http://adzap.cs.zip.com.au/ad.gif` This will actually work fine (once cached it's irrelevant that it's not on your site). However, if you wish a customised placeholder you can do a few things to control what is used. Most involve the setting of environment variables to indicate your desires.

If you start customising things I suggest you install the `wrapzap` script next to the redirector and use it to effect the customisations. Simply tell `wrapzap` the full install path of `squid_redirect` and tell the `squid.conf` file the full path of the `wrapzap` script instead of the zapper. Then modify `wrapzap` to suit.
I used to suggest that people put these settings in the squid startup script but that has some downsides outlined in the comments at the start of `wrapzap`.

The `$ZAP_MODE` variable can be set to the word "`CLEAR`" to cause the zapper to use "clear" versions of the replacement images and text. This will mean the ads just "vanish" from your pages. The only real downside to this is that is the zapper, through some mischance, replaces some useful markup on the page then it's not very apparent.

The `$ZAP_BASE` variable can be set to point to a web directory containing your own versions of the replacement images. Place files named `ad.gif`, `closepopup.html`, `no-op.html` and `no-op.js` there.

The default for `$ZAP_BASE` is `http://adzap.cs.zip.com.au/`. If you set the `$ZAP_MODE` variable to "`CLEAR`" then you will naturally want files named `ad-clear.gif`, `closepopup-clear.html`, `no-op-clear.html` and `no-op-clear.js`.

You can replace classes of ad with specific replacements. The following classes are known: **AD** for inlined images, **ADHTML** for separate HTML pages inserted as an ad (usually via FRAME or ILAYER tags), **ADJS** for

javascript programs used to generate ads, **ADPOPUP** for those mega-annoying ads which pop up on their own as new web pages and **COUNTER** for inlined visitor count images. Each of these words matches the keyword on the start of the lines in the configuration file. To control each you would set the variable `$STUBURL_class` to the URL of the specific replacement for that class.

For example, setting

```
STUBURL_AD=http://adzap.cs.zip.com.au/ad-clear.gif
```

which would cause the inlined images to be the "clear" version while leaving the other classes as normal. That `ad-clear.gif` is a transparent single pixel GIF donated by David Finster <dfinster@airmail.net>. Another image you might like is http://adzap.cs.zip.com.au/ad-grey.gif, from Andrew Dalgleish <andrewd@axonet.com.au>, which is a low contrast replacement image whcih lets you see what's zapped without it standing out so much.

## Zapping Things Other Than Ads

The default behaviour of the zapper is to zap ads only (the AD, ADPOPUP and ADJS classes). However, I desire that it can be used to zap other animated annoyances like flashing "NEW!" icons and glowing line images used in place of the venerable <HR> horizontal rule markup.

Accordingly, the pattern list contains patterns for more than just ads. By default, these extra patterns are ignored. To cause the zapper to start using a particular class of pattern, set the environment variable `STUBURL_class` to a suitable URL.

For example, I set the STUBURL_* variables in the squid startup script like this:

```
STUBURL_AD=http://adzap.cs.zip.com.au/ad.gif \
STUBURL_NEW=http://adzap.cs.zip.com.au/new.gif \
STUBURL_ADPOPUP=http://adzap.cs.zip.com.au/closepopup.html \
/opt/squid/bin/squid -f /opt/squid/etc/squid.conf -sY &
```

Actually, I suggest you use the `wrapzap` script if you start customising things; its purpose to to put the customisations in a place distinct from both the redirection script (which gets updated occasionally, which could trash your customisations) and the system startup scripts (which have the same issue). Also, it's been reported that sending a SIGHUP (or saying "`squid -k reconfigure`") to restart the squid loses environment settings; wrapzap works around that too.

## Chaining Redirector Programs

Chris Lightfoot <chris@ex-parrot.com> wrote asking if I could make the zapper friendly to setups where people chain multiple redirection programs together. The specification for the redirectors says unredirected URLs should be indicated with a blank line, which is no good for piping the output of one into the next. Accordingly, if you set the environment variable $ZAP_CHAINING to "1" the redirector will write unredirected URLs as-is to its output instead of blank lines. Then Adam Hope <a.hope@csl.gov.uk> wrote to say that they were chaining to another redirector which wanted the full 4 word input a redirector might expect. Therefore, if you set the environment variable $ZAP_CHAINING to "FULL" the redirector will write the original 4 word input data to its output with the first word the redirected URL or not as required.

# Updates

You have a few choices about keeping up to date with the patterns (and the matching `squid_redirect`).

- Join my mailing list for updates; I will send you the new script and pattern list whenever the pattern list grows. Just email me to join.
  I send out three messages when an update occurs:
  - a new list of the patterns; this is the source file I maintain.
  - a list of differences to the patterns; this summarises the changes, in case you care.
  - a new script; this last is all you really need - it is the perl script autogenerated from the pattern list. Simply copy it over the existing script and restart your squid server. The command "`squid -k reconfigure`" will do this.
- Don't want to join the list?
  Then you can simply fetch the script from this page every so often. You can ask the URL Minder to keep an eye on it for you, too.
- Alternatively, you can fetch new versions automatically. At Thomas B. Fox's suggestion I devised the following method. Have a cron job invoked like so:

  ```
  0 0 * * * /usr/local/etc/update-zapper
  ```

  (Hack "`/usr/local/etc/update-zapper`" to match wherever you install the automatic update script.) That should go in the crontab of some user with write permission to the `squid_redirect` script as installed on your squid host and permission to send signals to the zapping squid daemon. Probably this user is called "squid", but this would work as root too. Then install the update-zapper script. The script needs the wget program . Damien Clermonte <damien@allconet.com> has sent me a copy of his update script which you might prefer to use.
- Don't want me to do your maintenance?
  Copy the config file, the redirector and the `updsquid_redirect` script and maintain your own set of patterns. You will also need mksquid_redirect_ptns, replace, and bsed if you choose this route.
  Obviously I'd rather collaborate in this; then we can keep a single central list.

# Using the zapper in `proxy.pac` files

[ Also see: Can I get my ISP to do this for me?, below. ]

If you have to support more than a few users, you may want to use a `proxy.pac` file. This is a file containing a JavaScript function used by a browser to decide which proxy to use (if any) on a per-URL basis. This is often known as "automatic proxy configuration", as all you tell the browser configuration is the URL of the `proxy.pac` file. Once you've set this up for each of your users, you can then control things by editing the central file. Both Netscape and Internet Explorer support `proxy.pac` files.

## Can I get my ISP to do this for me?

If you petition them, maybe. The setup at their end is pretty easy. However, they may refuse. For example, ZipWorld no longer support the zapping service themselves; instead I now supply this service for Zip people who wish it. (Essentially, their legal people have raised the spectre of zapping somehow being construed as a kind of copyright violation. Personally I think that's daft; it's no different to browsing in text mode with lynx or with image loading off in a graphical browser).

Thus it may become a case of "do it yourself". However, at least in my case, ZipWorld were happy enough to up my disc limit a bit, let me run the zapper all the time (even when not logged in), and automate a monthly post to a

local newsgroup to tell people about the zapper. Very cool!

### Sites with the Zapper already installed

ZipWorld
: People at Zip can simply use the prepackaged .pac file URL:

```
http://adzap.cs.zip.com.au/proxy-zip.pac
```

: Users of other ISPs can contact me for details on how to I set this up.

CISRA's intranet
: At my workplace I supply two for our users: one (the default) which runs through the ad zapping proxy and another which goes via our main proxy (which doesn't zap ads, and of which the zapping proxy is a child).

Here are a few example `.pac` files which I've set up for various sites. Each would require some customisation for your own site.

- proxy-cisra.pac, which I use at work.
- proxy-home.pac, which I use at home.
- proxy-zip.pac, which can be used by people at my ISP (ZipWorld).

## Why run an ad zapper?

There are a few reasons one might do this:

- General dislike of ads.
  I don't subscribe to this one, myself. I see the arguments for using advertising to support communal resources (especially ones which do not of themselves generate income).
- Bandwidth.
  Many ads seem to be devised by the technically illiterate. You find simple GIFs with hundreds of colours, tiny buttons requiring several kilobytes of download, etc. The simple act of quantising the colours to a few (between 4 to 32 would easily suffice for most ad graphics) would have *greatly* reduced the size of these things.
  You also find that many, maybe most, ads are CGI output. Instead of intelligently redirecting the browser to an image from a static library of banners (which would permit proxies to cache the images, saving repeated download) the CGIs emit the graphics themselves. Firstly, pretty much every CGI invocation has a different URL, as the source site is usually identified to track display rates. As a consequence identical ad graphics have different names and are not recognised as hits by the cache; each must be fetched from the CGI even though a cached version may exist. Secondly, CGI output is routinely marked no-cache; even when the URL is the same the content is often refetched.
  This cache unfriendly behaviour makes most ad banners the enemy of all who pay attention to the usage of their link.
- Animation.
  Even though I am on the end of a narrow link at home (i.e. a modem) my primary motivation in zapping ads is not that they are huge unwieldy wastes of bandwdth but that most of them are animated. My normally informative browser window has these bloody flashing, scrolling, moving distractions. They annoy the hell out of me and many others I know. A static ad can be a fine thing. Small and informative, it points the way to something possibly interesting. An animated GIF is an intrusive and rude wart on the surface of the web.

And it further offends by wasting bandwidth.

- Neal McBurnett <nealmcb@bell-labs.com> adds this remark:

> Ad sites commonly set permanent cookies on your browser. Via use of the HTTP_REFERER header they can then often track your activities on all sites they advertise on, and some companies advertise across a very wide range of popular sites. It wouldn't be unusual for them to get personal information on you just from the URLs in the HTTP_REFERER field, which may include form values including your name, things you like to search for, etc.
>
> See also - Cookie RFC (privacy section, but note that the popular browsers don't follow the guidelines!):
>
> `http://www.cis.ohio-state.edu/htbin/rfc/rfc2109.html`

He says this was his primary motivation for zapping doubleclick ads in 1996. I'd remark that while an ad zapper protects you from this (cookies attached to the inlined image), naturally if you follow the ad link anyway (since the savvy marketer will add a useful descriptive caption under the banner, permitting you to know what the ad is for) then you're on your own.

# Other Similar Software

Mine is hardly the only alternative you have in this line. Other tools include:

- Squid: Related Software
  A listing of interesting software related to squid, including a few other redirectors.
- Junkbuster
  Josh Marshall <MarshallJ@switch.aust.com> briefly compares them:

> **Similarities**:
> Both filter out those annoying advertisement pages that waste time and bandwidth, meaning money (we're paying for that!) Both use a list of sites and regular expressions to eliminate these advertisements. Both redirect the image to a default, smaller image.
>
> That's where the similarities end.
>
> **Differences**:
> Ad Zapper integrates much more nicely into squid. It is started from within squid (as many processes as you like) and is basically a URL redirector based on regular expressions that are contained inside the script.
> Junkbuster runs as a separate daemon, and you have to use it as a hierachial cache, with junkbuster as either the parent or child. I found having it as the parent (they document how to set it up as a child in the docs) to be the superior configuration. All fetches from an external web page must be redirected through junkbuster - which is quite slow compared to squid. Also the double handling makes for a slower transaction.
>
> Ad Zapper zaps ads - that's it. Junkbuster also can filter out cookies and web pages (like those annoying small ones that advertise the free web pages the site is from) I have found junkbuster to be a little too constrictive. It can also to web anonymity and return wafers instead of cookies for you with "leave me alone" privacy messages in them for the web administrators.
>
> My recommendation is this: If you want tight security then go for junkbuster. You're sacrificing some speed and some pages which simply wont load anymore since the pattern matching tries too hard. If you want performance without ads, go for Ad Zapper (you can even specify your own image which you can't do with junkbuster)

I've noticed that the recent squid release (2.2STABLE4 as I type this) has anonymising facilities, so you can perhaps use those in conjunction with squid_redirect to get what you want.

- Craig Sanders' <cas@taz.net.au> squid-redir tool.
  Quite similar in intent and implementation to my own.
- Cut The Crap
- AtGuard
- WebWasher
- adzapper (No, mot my ad zapper; this one is by Adam Feuer, and coded in Python.)
- SleezeBall, another squid based redirector.

# My complicated double-layer squid setup

At work we run a double layer squid setup. The purpose of this is to permit un-zapped access to the web for those few who want it (marketing types, as it happens:-).

We have a double squid cache (on the same machine). The usual proxy for users is:

```
proxy:8081
```

which has a smallish cache (100meg or something) and the URL redirector in its config:

```
redirect_program /opt/UCSDsquid/bin/squid_redirect
```

This lives off the main, non-redirecting cache at:

```
proxy:8080
```

which has a big cache. The proxy.pac file users use points them at:

```
PROXY proxy:8081; PROXY proxy:8080
```

and the proxy-raw.pac (which shows ads) says:

```
PROXY proxy:8080
```

# Troubleshooting

This might be as unhelpful as Microsoft's online help, but hopefully not.

Basic checks:

1. Make sure your squid proxy is working normally *without* the ad-zapper line in the config file.
2. Make sure the `squid_redirect` script has public read and execute permission.
3. Make sure the `squid_redirect` script is not in DOS text mode (if you fetched it from a Windows machine); see *Note 3* under step 2 of the install steps for a fix for this.
4. Examine your `cache.log` file for error messages from squid or the ad zapper.

Still stumped?
Here is a basic, untested, quick and dirty howto for setting this up from scratch if you haven't got squid running and have never used squid. *Please* attempt a normal squid install using their instructions (which come with the source) first! You should only need this is things fail obscurely and you're at a loss. It's just a sequence of things to do.
Here goes:

Planning:

Find out your ISP's proxy server and port. It's traditional that the server is called `proxy.`*`your.isp.domain`* and that it listens on port 8080. If that's documented by your ISP's web pages, well and good. If you have to guess, try connecting to it:

```
telnet proxy.your.isp.domain 8080
```

If you don't get a connection, try port 3128 instead of 8080.
If you get a complaint that the hostname is unknown, you'll have to consult your ISP.
If you get a connection, check that it's actually a web proxy. Type:

```
GET http://www.zip.com.au/~cs/ HTTP/1.0
```

and press return twice You should get an HTTP response (code 200 hopefully), some header lines, then some HTML. If you don't then that's not your ISP's proxy service, and you must contact them to find out the correct details.

Basic Sanity Checks:

Ensure your browser works with no proxies at all set up.
Ensure your browser works with its proxy setup to talk to your ISP's proxy service.

Squid:

Fetch the latest squid (2.2STABLE4 as I type this), build and install.
Edit the `squid.conf` file by walking through it from beginning to end in an editor, adjusting it to suit your host.
In particular:

- You should make it listen on a suitable port. Usually this is 8080; squid's default port is 3128. This is controlled by the `http_port` directive.
- You should make your squid use your ISP's proxy as its upstream service. This is controlled by the `cache_peer` directive. The relevant line from my squid at work says:

  ```
  cache_peer 203.12.172.230 parent 8080 3130 no-query default
  ```

  You would replace the "`203.12.172.230`" with the name of your ISP's proxy (eg "proxy.*your.isp.domain*") and the 8080 with the matching port number (probably the same).

Run "squid -z" to initialise your cache.
Run the squid startup script to set squid running.

Working?:

On your squid host, run the command:

```
netstat -an | grep -i listen
```

to check that squid (presumably) is listening on port 8080 on your machine.
As with your ISP's proxy, you should now test your proxy. Run the command:

```
telnet localhost 8080
```

to check, and issue the same `GET` command you used about to fetch a web page.

Test new squid:

Set your browser config to use the local machine (well, your squid host, which needn't be the same machine as where yoiur browser runs), port 8080 as its proxy.

Ad zapping:

Add the ad-zapper line to the squid.conf, restart the squid server and test again.

Not working? Maybe the script came fvia a DOS or Windows box and is in DOS text mode?

This usually shows up as failure (by squid) to run the script, so first check your script is usable by running it

by hand:

```
the-script </dev/null
```

That should do nothing, with no complaints. If this is greeted with messages like:

```
the-script: exec failed: No such file or directory
```

then you may have spurious CR characters in there. You can verify this with the command:

```
sed 1q the-script | od -c
```

which will print:

```
0000000 # ! / u s r / b i n / p e r l \n
0000020
```

for a good script and:

```
0000000 # ! / u s r / b i n / p e r l \r
0000020 \n
```

for a bad script (note that extra \r, which is a CR). These can be deleted with the `tr` command, viz:

```
tr -d '\015' <the-script >the-script.fixed
mv the-script.fixed the-script
```

which makes a new copy without the CRs and then replaces the orignal with the new one.