

Web Filtering and Monitoring on Linux

This is a tutorial on how to set up a proxy on Linux for the purpose of filtering and monitoring web access – it can even be done totally transparently so that no browser needs to be reconfigured AND no browser can get around the filter.

In this document, different formattings mean different things:

`/this/is/a/filename` (hint: most files need to be edited as root user)

`this is a line or a part of a line in a file`

`this is a command` (hint: most commands need to be executed as root user)

and finally, this is a complete script

that you should use as it is printed in this document

The software you need is:

- **Cron, AnaCron & LogRotate**: install it from your distribution, e.g. on Debian by executing `apt-get install cron anacron logrotate`
- An **email server** allowing you to send email using the “mail” command. You can test your existing installation by executing `echo test | mail -s subject yourmail@domain.com`
If there is no error reported and the mail appears in your inbox, there is nothing to be done. Else go to Appendix A now and install an email server on your machine! You will really avoid a lot of trouble if you check this first!
- **Squid**: from your distribution (`apt-get install squid`) or www.squid-cache.org
- **DansGuardian**: from your distribution (`apt-get install dansguardian`) or www.dansguardian.org
- **iptables**: from your distribution (`apt-get install iptables`) or www.netfilter.org (you might have to enable this feature in your kernel configuration, too – if you don't know what this means, don't worry, it's most likely to work out-of-the-box)

Once Squid and DansGuardian are installed, you still need to configure them.

Do this editing and restarting as root!

Edit the file `/etc/squid/squid.conf` with respect to the following recommendations:

1. enable the Apache log format emulation (for monitoring purposes):
`emulate_httpd_log on`
2. set the hostname:
`visible_hostname localhost`

Don't forget to restart the squid daemon (on Debian with `/etc/init.d/squid restart`).

Edit `/etc/dansguardian/dansguardian.conf` and ensure that in your file the line `UNCONFIGURED` is commented out like this:
`# UNCONFIGURED`

Now edit `/etc/dansguardian/dansguardianf1.conf` and set the blocking limit to a good value (I recommend to leave it ≤ 130), e.g.
`naughtynesslimit = 130`

You can see all the blocking lists in the first part of this file. Now start DansGuardian (on Debian with the command `/etc/init.d/dansguardian start`).

Because of the above mentioned line UNCONFIGURED it has not been started when it was installed.

To test your installation this far, configure your browser to use the proxy server **localhost** with proxy port **8080** (not 3128) and open a page that would be blocked. I suppose you take www.dgtest.dyndns.org (this is simply the file /etc/dansguardian/lists/phraselists/pornography/weighted which I reduced to 8K and uploaded to that server). It should surely be blocked because all of the weighted phrases are inside this file! My dansguardian said the weighted phrase count was 29365 (this is surely greater than the defined allowed value).

Hint: If your installation does not block the page mentioned above, be sure to clean your browser's cache before looking for a complicated error!

Now you might think: I don't like to configure every browser I use to go through that proxy, not even thinking of all the browsers I run in my different VMware machines! And what about simply disabling the proxy? Wouldn't that break everything and grant full access?

You're right. At this point, you can change the browser setting back to "direct access" again and have unfiltered and unmonitored web access. But read on...

A simple and fast possibility of automatically redirecting every request a browser makes for normal web pages is by inserting an iptables rule. This will effectively turn Squid from a *normal* proxy into a *transparent* proxy (browsers are unaware that they use this proxy and don't have to be configured to use it). To make Squid support this behaviour, open the file `/etc/squid/squid.conf` for editing as root and make the line `http_port 3128` look like this:

```
http_port 3128 transparent
```

Restart Squid (on Debian with the command `/etc/init.d/squid restart`).

Then, still as root, execute the following command (it is only one command, type it all on one line):

```
iptables -t nat -A OUTPUT -p tcp -m owner ! --uid-owner proxy --dport 80 -j REDIRECT --to-port 8080
```

Now test the transparent proxy setup by removing the proxy settings from your browser (activate the option "direct connection to the internet") and surfing.

You should have filtered web access, test it using www.dgtest.dyndns.org.

If you cannot reach any website, Squid might not run under the user "proxy" and so your request gets into an infinite forwarding loop. To test this, execute `lsof -i4 -i6|grep -e '^squid\W'`

and pay attention to the third column in the output, the username. If you don't see "proxy" there but i.e. "squid", you have to clear the iptables rule you set earlier by executing

```
iptables -t nat -F OUTPUT
```

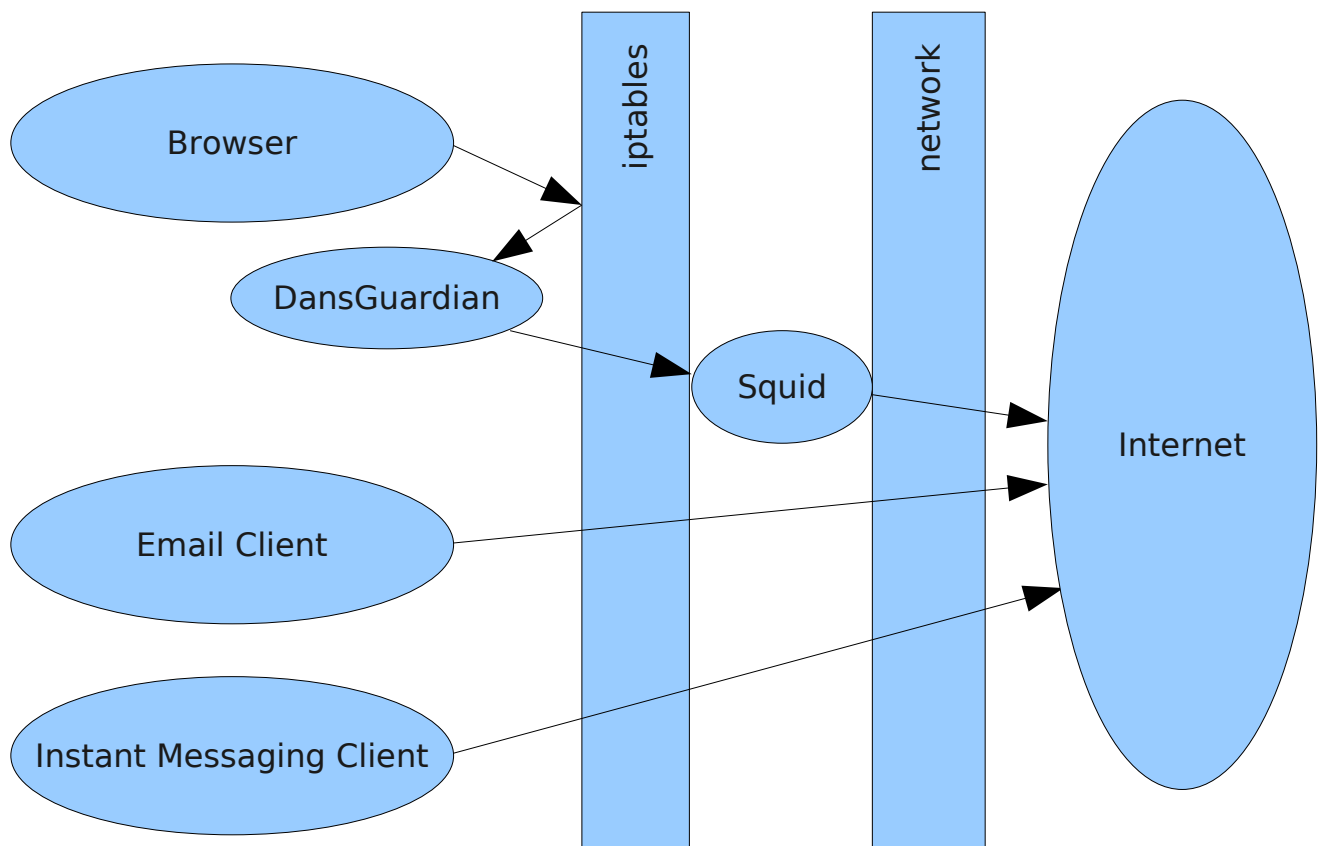
and finally you have to re-execute the above stated iptables command, but change the word "proxy" (after "--uid-owner") to the username you discovered.

Now there is only one hole left: a crafty user could configure his browser to use Squid directly. By setting **localhost** as proxy and **3128** as proxy port, he uses the Squid process, but circumvents the filtering done by DansGuardian. To fix this, execute the following command as root (again, it is only one command, type it all on one line):

```
iptables -t nat -A OUTPUT -p tcp -m owner ! --uid-owner dansguardian --dport 3128 -j REDIRECT --to-port 8080
```

With this rule, you allow only the user “dansguardian” to speak with Squid. Every other user is automatically redirected to DansGuardian. Test if you can still circumvent the filtering by inserting the proxy settings described above. If now all works to your complete satisfaction, you still have to create a file containing only the above iptables commands, each on one single line (or your customized version of it). Preferably use the file `/etc/init.d/transparentproxy` for this. Make it executable with `chmod a+x /etc/init.d/transparentproxy` and create startup links by executing (on Debian) `update-rc.d transparentproxy defaults`. With this, the iptables command gets executed automatically on boot time.

The layout of the system is now like this:



You can see that requests from the browser are intercepted by iptables and redirected through DansGuardian and Squid before they are really forwarded into the internet while every other program continues to work fine.

Now we can move on to the monitoring part:
Create the following script as e.g. `/bin/visited_urls`:

```
#!/bin/bash
#
# print the URLs that Squid was asked for
#
echo $1
```

```

echo ""
cat /var/log/squid/access.log | \
grep -v .tar.bz2 | grep -v .tar.gz | grep -v .pdf | \
grep -v .jpg | grep -v .gif | grep -v .png | \
grep -v .ra | grep -v .ram | grep -v .rm | grep -v .mp3 | grep -v .mid | \
grep -v .css | grep -v .js | grep -v .ico | \
sed -e 's#.*http://#http://#' -e 's#.*CONNECT.#https://#' -e 's#[[:blank:]].*###' | \
sed -e 's#\?###' -e 's#[0-9][0-9]*###' -e 's#[\|;].*###' | \
sed -e 's#/#$#3' -e 's#$.*###' | \
sort -u 2>&1

```

And create this one as e.g. `/bin/mail_visited_urls`:

```

#!/bin/bash
#
# mail the visited URLs to an accountability partner
#
/bin/visited_urls "Myname loaded something from the following web sites:" | \
mail -s "Web Sites - Myname" mypartner@accountability.net
echo "mail_visited_urls @ $(date)" >> /var/log/mail_visited_urls.log

```

In the second script you should of course fill in more appropriate values than *Myname* and *mypartner@accountability.net*. If you want to send it to more than one person, you may repeat the lines

```

/bin/visited_urls "Myname loaded something from the following web sites:" | \
mail -s "Web Sites - Myname" mypartner@accountability.net

```

with other values. It is important that you have them both for each recipient directly following one after the other.

Make the scripts executable by
`chmod a+x /bin/*visited_urls`

If you want to test the monitoring so far, edit the script `mail_visited_urls` to contain your own email address and execute `"/bin/mail_visited_urls"` as root. Now check your email!

And here comes the last part: Edit `/etc/logrotate.d/squid` to contain something like the following. The important parts are in red:

```

/var/log/squid/*.log {
    weekly
    compress
    delaycompress
    rotate 8
    missingok
    nocreate
    sharedscripts
    prerotate
        /bin/mail_visited_urls
    endscript
    postrotate
        test ! -e /var/run/squid.pid || /usr/sbin/squid -k rotate
    endscript
}

```

If you don't want to have the logs examined `weekly`, you may also set this line

to **daily** or **monthly**.

Now you are done, the AnaCron will execute the LogRotate script for the Squid log files every Sunday when you switch on your computer (if you chose “weekly”), and using the script mail_visited_urls it will create nicely formatted and alphabetically sorted mails and send them away. If you don't use your computer on Sunday, then the mail will be sent the next time you start it.

You may send any questions to filter@jesusfan.de!

Appendix A: Install and Configure an Email Server

This is pretty important, so read carefully. First you should install Postfix (this one is easy to configure) and some modules for it. On Debian machines simply execute “`apt-get install postfix postfix-tls libsasl2 libsasl2-modules`”.

While installing Postfix, you probably have to answer some questions. As generic configuration choose “Internet with Smarthost”. If asked for the relay or smart host, type in your provider's SMTP server.

Now add to `/etc/postfix/main.cf` the following lines:

```
# use SMTP with SASL authentication
smtp_sasl_auth_enable = yes
# passwords are in /etc/postfix/smtp_auth
smtp_sasl_password_maps = hash:/etc/postfix/smtp_auth
# no anonymous mails
smtp_sasl_security_options = noanonymous
```

If there is a line like

```
relayhost = smtp.myprovider.com
```

already, then it was created when you selected the “Internet with Smarthost” option above. If not, add this line with your provider's SMTP server in it!

Now create the file `/etc/postfix/smtp_auth` and edit it to look like:

```
smtp.myprovider.com    myusername:mysecretpassword
```

Execute `postmap /etc/postfix/smtp_auth` to convert the file into hash format. Now it should all be well-configured. Reload the Postfix configuration (with `/etc/init.d/postfix reload` on Debian).

Now send a mail to an address that is *not hosted by your own provider*. This is necessary because the password authentication would not be needed if you sent a mail to one of the server's own users, but this is what we want to test! The command you should use to send mail from the command line is `echo test | mail -s subject yourmail@domain.com`.

Now check if your mail was delivered. In the file `/var/log/mail.log` should appear something like the following:

```
Feb 2 13:14:51 myhost postfix/pickup[5966]: 8883118058: uid=0 from=<root>
Feb 2 13:14:51 myhost postfix/cleanup[5972]: 8883118058: message-
id=<20050202121451.8883118058@myhost>
Feb 2 13:14:51 myhost postfix/qmgr[5967]: 8883118058: from=<root@myhost>, size=278,
nrcpt=1 (queue active)
Feb 2 13:14:52 myhost postfix/smtp[5974]: 8883118058: to=<otheremail@other.com>,
relay=smtp.myprovider.com[12.34.56.78], delay=1, status=sent (250 Ok: queued as
0F577342749)
Feb 2 13:14:52 myhost postfix/qmgr[5967]: 8883118058: removed
```

If so, your Postfix is functional. If not, you should send me an email or check the documentation of Postfix at www.postfix.org!