Privoxy and Webcleaner content filters

DOORKEEPERS

Angel Janer, http://home.coqui.net/janer/

Content filters protect a web user's privacy and keep the flood of unsolicited advertising at bay. We'll show you a pair of popular Open Source content filters.

BY THOMAS LEICHTENSTERN

Content filters are useful for preventing unrestricted traffic between browsers and web servers. A good content filter only allows the traffic the user really wants, keeping the browser clear of junk advertising, web bugs, cookies, and unwanted Javascript. Some filters can also manage outgoing traffic. A properly configured content filter can even protect you against browser security holes which, unfortunately, are still extremely common.

This article investigates the Privoxy [1] and Webcleaner [2] content filters. Both tools provide content filtering, but whereas Privoxy focuses on web content, Webcleaner has a variety of additional features, such as a virus filter and an image compressor.

Redirecting Access

Content filters work like proxy servers; that is, as intermediate systems between the browser and the web server. To use a content filter, you need to redirect browser web access attempts to the address of the content filter by changing your connection settings to point to the filter's IP address and port. If the filter is running on your local machine, the address will be 127.0.0.1 ("localhost").

Privoxy

The Junkbuster-based Privacy Enhancing Proxy – or Privoxy for short – is a simple content filter that does not use caching. In contrast to a simple URL filter, however, the program does check the complete website content based on predefined rules.

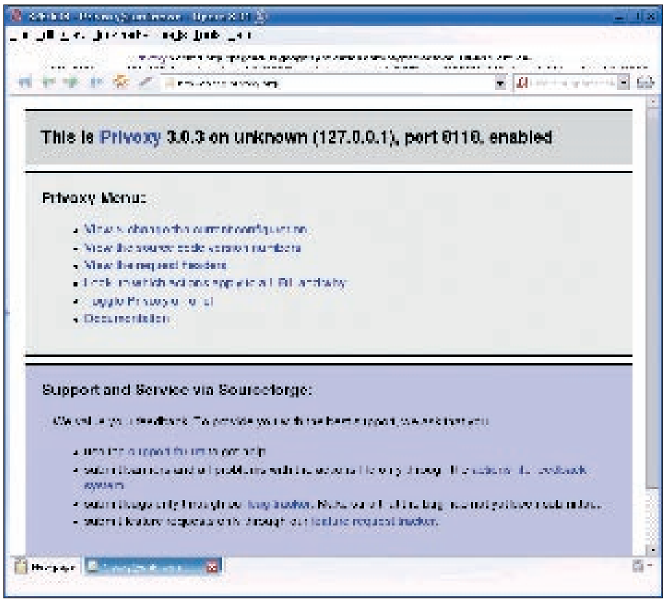
Installation

Privoxy is available for any major distribution, so you should have no trouble finding RPM or DEB packages. Debian users can enable the *Universe* repository and install Privoxy by entering *apt-get install privoxy*. For Suse 9.3, run Yast to set up the program from your installation media. Note that Yast installs the program in a chroot jail. The path to the configuration and log files is */var/lib/privoxy/*.

Configuration

By default Privoxy binds to the localhost (127.0.0.1). If you want the content filter to be available to other machines, you can change the default value in the */etc/privoxy/config* configuration file from *listen-address 127.0.0.1:8118* to the LAN interface address, 192.168.0.1, for example. If you leave out the IP address, the service will bind to all network interfaces, which is not recommended, especially not for computers with direct Internet access.

A web interface allows you to set up the filter rules; you can reach the interface at *privoxy.org/config*. But before you do so, set up Privoxy as a proxy for your browser. For Debian-based systems, this function needs to be enabled first. To do so, look for the *enable-remote-toggle* and *enable-edit-actions* entries in the */etc/privoxy/config* configuration file, and set both values to *1*. After completing the



This is Privoxy 3.0.3 on unknown (127.0.0.1), port 8118, enabled

Privoxy Menu:

- Documentation

Support and Service via Sourceforge:

Figure 1: The main Privoxy window offers a menu of links for various functions.

changes, relaunch the program by entering */etc/init.d/privoxy restart*. As Privoxy does not support user authentication, any user with access to the content filter can change the filter settings.

Filters

Privoxy distinguishes between filter and action files. The filter includes rules such as a rule for removing banners of over a certain size. The action files map rules to addresses. The latter can be anything from simple URLs to wildcards that represent fragments of addresses belonging to advertising pages. *ad*.example.com* would cover all the subdomains in the *example.com* domain that included *ad* followed by an arbitrary string.

Hasta la Vista, Baby

The default filter file (*/etc/privoxy/default.filter*) comes with a large collection of rules. You should not attempt to change anything in the filter file unless you feel comfortable with regular expressions. As Privoxy does not have a front-end for editing the filter files, you can use your favorite editor for making changes. The following example shows what filter rules look like:

```
FILTER: LinuxMagazine
Sample filter rule
s/rain(?!.com)/sun/ig
```

FILTER: defines a new class and is followed by the filter name (*LinuxMagazine*) and a description (*Sample filter rule*). The Privoxy web GUI displays this information for the filter. The second line contains the rule itself. In this case, it replaces *rain* in strings with *sun*. A class can comprise any number of rules, which you can enable by single-clicking in the web front-end. You'll find a number of downloadable filter lists on the Internet [4].

And ... Action!

The most carefully crafted rule is useless if you don't have a target. And this is what action files are for. *user.action* and *default.action* are both action files, and both are edited via Privoxy's web GUI, which you can access via *config.privoxy/show-status*. Although both files have an identical structure, they are used for different purposes. While *default.action* specifies global behavior, *user.action* handles specific applications.

The *default.action* file includes default rules that apply when neither of the other files applies. Privoxy has three predefined policies for new users; in fact, you can point and click to set the filter behavior from *Cautious* to *Adventuresome*.

The following sections contain policies for handling addresses and address fragments, describing an approach to handling ads based on URL patterns such as *ad*.* or **banner*.*, for example. The global configuration settings are mainly restricted to selecting one of the default policies.

User-defined rules are created in the *user.action* section. If Privoxy is blocking content from your favorite page, you can use the *http://config.privoxy.org/show-url-info* link to find out which filter is responsible. You can click the *Insert new Section at top* button in the *user.action* section, and then click *Add* to add the URL for the page. *Edit* shows you a list of the rules in *default.filter* that apply in this specific case. The default setting for these rules is *No Change*. Any settings made here take priority over the default policies.

Work in Progress

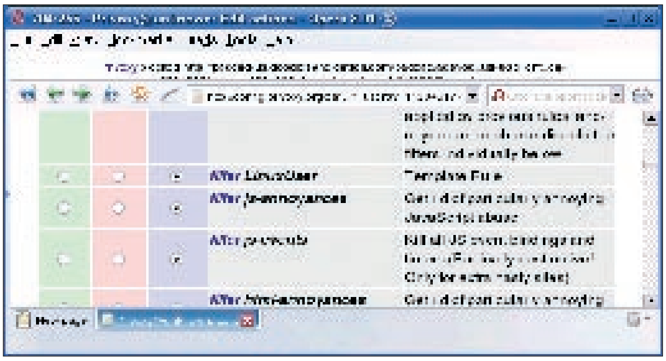
Privoxy is quite unobtrusive in daily use and hardly impacts page loading times, even for larger pages. Of course, you should make sure that the hardware where you will run Privoxy is not too ancient. A 500 MHz CPU and 256 MB RAM is the lower limit.

The program will display most websites correctly, even if you apply the *Cautious* filter setting. If not, you can launch the URL checker, which tells you the rules that apply for the current page. Unfortunately, the checker lacks a clear structure, leaving the user with the problem of finding the rule responsible for spoiling their view.

If a site is blocked completely, Privoxy still has an escape route via an aptly titled *go there anyway* link. The link disables the content filter temporarily for the current page. The program provides so-called bookmarklets for quickly enabling and disabling filters. You can open the bookmarklets by clicking on the *Privoxy - Toggle Privoxy* link on the *http://config.privoxy.org* welcome page. This takes you to a small popup where you can enable and disable Privoxy.

Privoxy: Conclusions

Much praise goes to the developers for the exemplary documentation that



Template Rule

Figure 2: Privoxy displays rules in the web front-end.

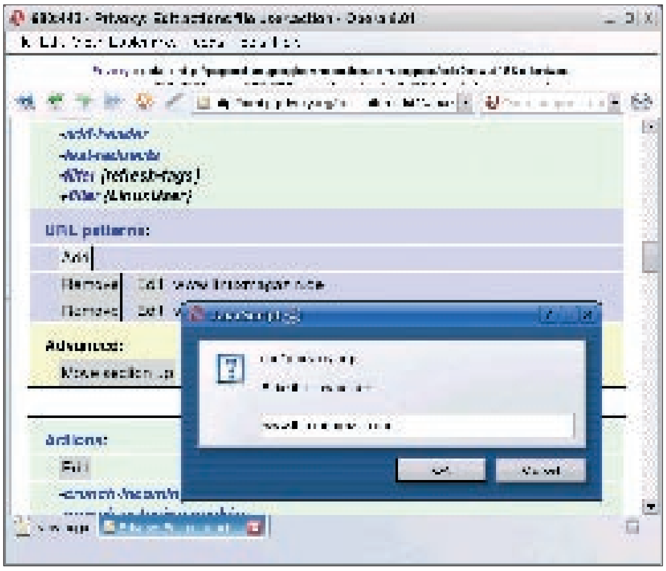


Figure 3: Once you have grasped the principle, you can easily create your own actions.

describes all of the program's features in detail. The combination of filters and action files might seem confusing at first glance, but if you look closer, it turns out to be a very good idea.

All in all, Privoxy has a very mature feel. It ran without a single glitch in our lab, and it fulfilled the assigned task of keeping our browser free of advertising and protecting the user's privacy without a change to the defaults.

Webcleaner

Our second test candidate is Webcleaner, which has an almost unbelievable feature list. Besides content filtering, the developers point to capabilities such as image compression and scaling, virus filtering, and detection and correcting of known HTML errors.

Installation

As Webcleaner was mainly developed for Debian, it is easiest to install on Debian-based systems such as Ubuntu. This said, the program can be installed on other distributions, although this may involve a lot more effort, especially in the case of Suse Linux.

Webcleaner requires *runit* and version 2.4 of the Python interpreter, including the developer packages. To build Webcleaner from the source code, you additionally need a C compiler such as *gcc*.

You need to install the following programs and libraries in advance if you want to use Webcleaner's full featurescope:

- PIL (Python Image Libraries) – for image compression and scaling
- Open-SSL and Python-openssl – to apply the content filter to SSL-encrypted web pages
- Clamav (*clamd*) – to use the virus filter

The *psyco* Python extension can be installed as an optional extra. According to the developers, *psyco* enhances performance when compiling Python scripts by a factor of between 2 and 100, although it does take up a lot of memory.

Users with Debian can run apt-get or the Synaptic front-end to install the required packages. If you have Suse 9.3, you will need to set up *runit*, *psyco*, and PIL manually, although the remaining packages should be on your Suse installation media.

Installation

Type *tar xfvz webcleaner-2.29.tar.gz* to unpack the tarball, and then change to the newly created directory. Then launch the build by entering *./configure && make*. The script might not complete due to a missing library (*/lib/cpp*), especially on Debian. If so, install the *openC++* package and repeat the command.

When you are finished, compile the Python files by entering *python setup.py build*. Entering *python setup.yp install* then sets up the Webcleaner build on your computer.

If you intend to use Webcleaner's proxy filter feature for encrypted websites, you will additionally need to install the required certificates. You can install the certificates by entering *webcleaner-certificates install*. Finally, type *make installservice* to set up the Webcleaner daemon, which is monitored by *runit* and launches immediately after you type the command. Users with Suse systems will first need to create a */var/*

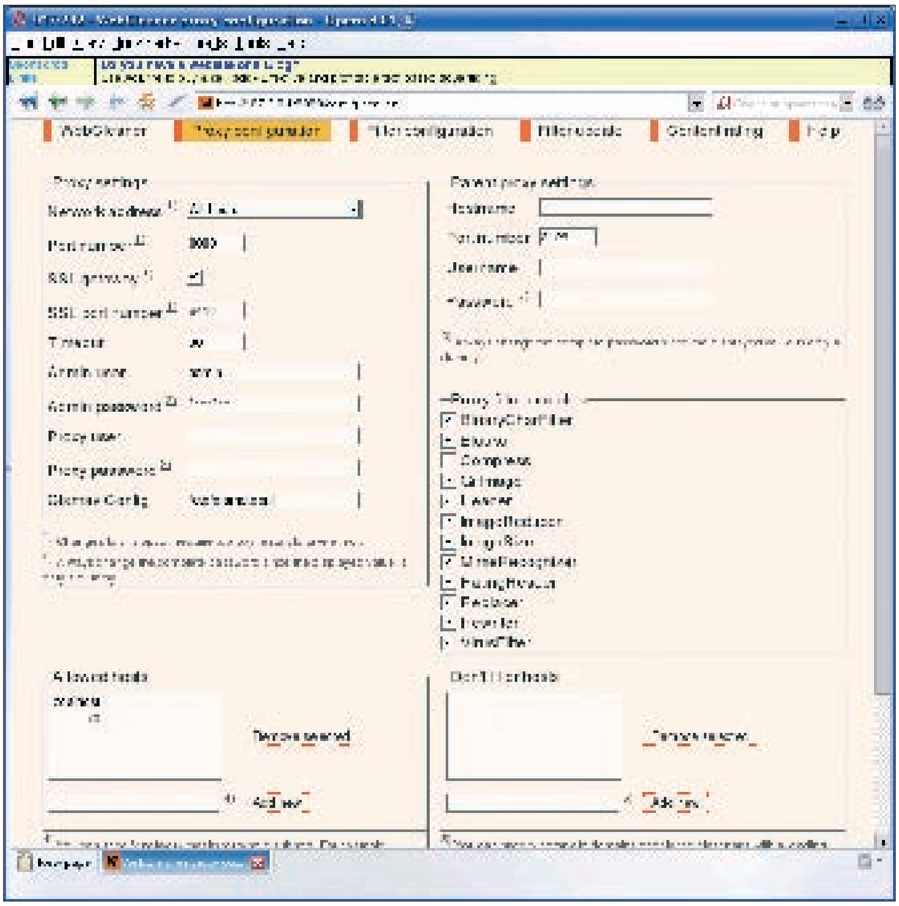


Figure 4: The Webcleaner command HQ.

service/ directory before they can run the script.

Configuration

Webcleaner can be accessed as a direct or parent proxy via Squid. The path for the connection then runs from the browser via Squid to Webcleaner, and on to the Internet from there. To use this approach, configure port *3128* as your browser's proxy port. If you would like to extend access to other machines, you can add the following entries to your configuration:

```
....
acl localnet src 192.168.0.0/255.255.0.0
....
http_access allow localnet
....
```

The direct route to Webcleaner uses port 8080, which you will need to configure in your browser connection settings. Webcleaner is configured via a web front-end accessible at *http://127.0.0.1:8080*. Before launching Webcleaner for the first time, you need to add the password that Webcleaner generates to your configuration file. To do so, copy the MD5 password to your */usr/share/webcleaner/config/webcleaner.config* configuration script, inserting the password after *adminpass* = , and relaunch Webcleaner by typing *kill -HUP Prozess-ID*. Then log on as *admin* and retype the plain text password displayed below the MD5 password on the webpage.

Proxy Configuration gives you access to basic program settings. Check the *Proxy filter modules* box for an overview of the available filter modules, the most important of which are:

- *Blocker* – the URL filter
- *compress* – compresses transmitted files
- *header* – changes, modifies and deletes the HTTP header
- *Image Reducer* – compresses images using a low-res, low volume JPEG format.
- *Rewriter* – parses and rewrites HTML and Javascript code

At the bottom of the window, below *Allowed Hosts*, you can specify the machines allowed to log on to the proxy. Enter any machines permitted to receive unfiltered content in *Don't filter Hosts*.

The *Filter configurations* section includes rules for files. The entries in the left column represent directories, and the rules for these directories are displayed in the center column when you click on a directory. Clicking on an existing rule opens a configuration menu in the right-hand column. To create a new rule, click on *New rule*; this opens a configuration menu from the drop-down on the right, depending on the entry you selected. You can then enter the desired action.

You can use the *Content rating* section to specify how to rate web pages. In our lab, Webcleaner only sporadically accepted any entries made here.

Pie in the Sky

In our lab, Webcleaner showed some major weaknesses. After we enabled the virus filter, some downloads didn't work or were unreliable. When we opened a carefully crafted website that contained a current browser exploit, our lab machine crashed. The SSL gateway refused to work at all in our test. When we tried to access an encrypted URL, the browser just displayed an empty page. The web interface seems to accept some entries arbitrarily while refusing others.

The Webcleaner documentation is no help with this kind of issue, as it does not give any real insight into how the program works. Combined with the highly unintuitive filter configuration, using the program is complicated. Our attempts to talk to the project's initiator about these issues remained unanswered when this issue went to press.

One obvious advantage that Webcleaner has in comparison to Privoxy is that it supports user authentication, both for access to the configuration front-end, and for use of the proxy itself.

Webcleaner: Conclusions

Although Webcleaner shows some promise, the sheer bulk of bugs makes it hard to see. The biggest benefit that Webcleaner offers in comparison to other programs – the virus filter – fails miserably under practical conditions, and other features, such as the image reducer and file compression, will not have much impact on the local machine. ■

Filter Mechanisms

Filtering tools fall roughly into the following catagories:

1. URL Filters

URL filters simply compare the URL entered in the browser with blacklists or whitelists stored locally. The addresses can comprise complete domain and host names or just URL fragments. URL filters are mainly used for user access control. Squidguard is a major player in this category: *http://www.squidguard.org/*.

Benefits:

- Low resource usage
- Fast processing speed
- Easy to configure

Drawbacks:

- Lacks privacy protection
- Does not check web page content

2. Content filters

This approach checks the content of any pages based on various criteria. The scope depends on your choice of program. Pop-up filters, cookie filters, and banner blockers are common. Some really intelligent programs, such as DansGuardian, *http://dansguardian.org*, validate websites based on weighting. If a term on the page exceeds a preset threshold value, previously defined rules are applied.

Benefits:

- Granular control of displayed content
- High level of security

Drawbacks:

- Depending on the configuration, loading time can be slow; high resource usage
- False Positives (legitimate content may be filtered)
- Complicated configuration

3. Special Cases

Some content filters, such as Webcleaner, have additional mechanisms, including web page and image compression, as well as virus checks.

INFO

[1] Privoxy: *http://www.privoxy.org*

[2] Webcleaner: *http://webcleaner.sourceforge.net*

[3] Junkbuster: *http://internet.junkbuster.com/*

[4] Privoxy Rules: *http://www.neilvandyke.org/privoxy-rules/*