Reference Manual

ufdbGuard version 1.24

the internet filter for the Squid web proxy

Table of Contents

1 Introduction

ufdbGuard is a URL filter software suite that can be used with the Squid web proxy to block unwanted web sites on the internet. URLfilterDB is the URL database that ufdbGuard uses to determine which web sites are restricted. ufdbGuard also features the capability to enforce Google's SafeSearch, HTTPS tunnel detection and verification of HTTPS/SSL certificates.

You may register as a trial user at www.urlfilterdb.com to receive a 60-day trial license for the URL database. The trial license is for an operational URL filter without restrictions.

Internet access restriction tools are usually implemented to increase the productivity and shield users from unwanted web content. ufdbGuard is also a network protection tool blocking access to proxies and tunnels.

To be able to use an internet access restriction tool, it is necessary to have a well-defined internet usage policy. It is common to actively share this policy with all employees and define clearly what is expected from employees.

1.1 What is URL Filtering?

URL filtering is a mechanism to block access to certain types of websites. Usually the reason to use URL filtering is to achieve higher productivity and to lower bandwidth usage. By only blocking advertisements and file sharing sites (p2p) the internet bandwidth usage can drop considerably. A URL filter can also block adult sites and prevent access to websites where an individual can copy an internal document to this public website with one or two mouse clicks. Finally, a URL filter can protect the IT infrastructure by blocking access to websites that promote themselves as "safe" but are not.

Users browse the internet and often without knowing it, they usually use a web proxy like Squid that is in between a PC and the internet. Squid and ufdbGuard can verify if a user has access to a particular (part of) a website that is visited. Approved websites can be visited without restriction. In case that a (part of) a website is restricted; the web displays a message that access is prohibited.

1.2 Why is Tunnel Detection a Must-Have ?

Users who want to circumvent company internet and security policies, may use so-called *tunnels* to break through the firewall that protects your internal network from attacks from the outside. Tunnels can be extremely harmful since they circumvent firewall rules, the anti-virus protection is circumvented and tunnels can be *bidirectional*, i.e. the firewall is open to attacks from the outside through the tunnel. ufdbGuard detects and blocks these tunnels.

SSH tunnels are one of the most dangerous types of tunnels where an entire corporate LAN can be exposed to anybody on the internet. Tunnels transfer firewalls into open doors!

Very little knowledge is required to make a tunnel: everyone who knows how to type "how to punch holes in firewalls" at a search engine can do it. The reader is urged to get familiar with the risks of (SSH) tunnels to be able to implement a proper security fence. Security officers and senior management are advised to verify that a security solution is implemented to block tunnels.

1.3 Latest Major Enhancements

Version 1.18 supports a new URL database format which reduces the database load time. The time to load the database is reduced significantly on Solaris with multithreaded applications when the libumem library is used. The old database format is supported for backward compatibility.

Version 1.22 fixes a bug that crashes ufdbguardd when it verifies some SSL certificates and has a new script to convert flat file URL databases to the `ufdb` database format. Starting with version 1.22, the SafeSearch feature can also be used in a fine-grained mode.

Version 1.23 was ported to OpenBSD and supports time-based ACLs which are usually used to relax filtering rules after a certain time. It has has a significant performance improvement of 20% to 96% depending on which URL categories are used. A new option to block/allow Skype was introduced.

Version 1.24 introduced options to monitor the URL filter by email and command script execution and has new configuration parameters to define how to behave in case of a fatal error or during a database reload. Ufdbguardd also reports on the status of the license and gives a warning if it expires in 2 months or less.

A detailed list of all changes can be found at the top level of the source tree in the file CHANGELOG.

1.4 Copyright

The ufdbGuard software suite version 1.x is free. To protect the ownership and the freedom of use of ufdbGuard, there is a copyright and you have a license to use and modify the software freely, known as the GPL version 2 license. The license is here: www.gnu.org/licenses/old-licenses/gpl-2.0.html.

The software suite is divided in 2 packages known as *ufdbguardd* and the *ufdbguardd API*. Ufdbguardd is inspired by squidGuard and parts of it are written by third parties and have a GPL2 copyright by those authors. The ufdbguardd API is meant to be used by vendors who wish to integrate URL filtering in their products and does not contain any code of third parties.

The URL database is a commercial product and has a copyright by URLfilterDB. A license is required to use the URL database which is defined in The Terms of Contract document that can be downloaded at the website: www.urlfilterdb.com.

1.5 Support and Feedback

We welcome feedback from clients and those who test our software. Feel free to send your questions and feedback to the support desk: support@urlfilterdb.com.

2 Prerequisites

ufdbGuard runs on all flavors of UNIX and is usually installed on the same system where Squid is installed. The Squid proxy software can be downloaded at www.squid-cache.org.

ufdbGuard needs 2 GB disk space and 200 MB memory for the 32-bit version and 350 MB memory for the 64-bit version. The required CPU power is compared to Squid relatively low, so it can run on one CPU for small to medium sized user groups. For large user groups, it is highly recommended to use a multicore system and to use top-of-the-line servers for very large environments.

ufdbGuard uses probes to HTTPS sites to verify SSL certificates and to detect SSH tunnels and proxies. For these features ufdbGuard needs to be able to open direct communication channels with web servers. If any of these security enhancing features of ufdbGuard are required, firewall configurations may need to be modified.

For all systems, ufdbGuard must be compiled with a C compiler. Most UNIX distributions come with the free GNU C compiler, `gcc` (see also gcc.gnu.org) or a native C compiler. In addition, the `wget`, `make` and `install` commands are required which are all included in most UNIX distributions.

ufdbGuard uses a compressed database and therefore requires the compression library bz2. This library is installed on most UNIX systems. In case that it is not on your system, it can be installed from your UNIX distribution CDs or downloaded from www.bzip.org. It also uses openssl and needs the openssl header files which are included in your UNIX distribution (usually the package name is openssl-devel).

Starting with version 1.24 it is recommended to install the `gdb` debugging package to receive detailed information in the event that ufdbguardd terminates unexpectedly.

3 Architecture

Squid is a popular web proxy that is used as an internet cache and is usually part of the internet security solution where users are not given direct access to the internet. Squid is free and can be downloaded from www.squid-cache.org.

Squid uses child processes called *redirectors* or *URL rewriters* to verify URLs for blocking. The redirectors are lightweight processes (`ufdbgclient`) that communicate with a multithreaded daemon process (`ufdbguardd`). The daemon has only one copy of the URL database in memory and does the actual URL verification. The daemon sends a reply indicating OK or BLOCKED to the redirector which sends the answer to Squid.

The `ufdbguardd` daemon uses a UNIX socket to communicate with `ufdbgclient`. Alternatively, a TCP socket can be used with port 3977. `ufdbguardd` and `ufdbgclient` can be configured to use an alternative TCP port number.

To analyse SSL certificates of HTTPS sites and to detect proxies and SSL tunnels, ufdbGuard connects directly to web servers on the internet. Therefore ufdbGuard needs to be installed on a server that has direct internet access. Usually such server resides within a DMZ. A DMZ is not a requirement by ufdbGuard but a DMZ is highly recommended for all corporate environments for reasons of security.

In the next simplified diagram, all pieces are put together: the web browser on a PC connects to the Squid web proxy who uses ufdbGuard to block access to websites.

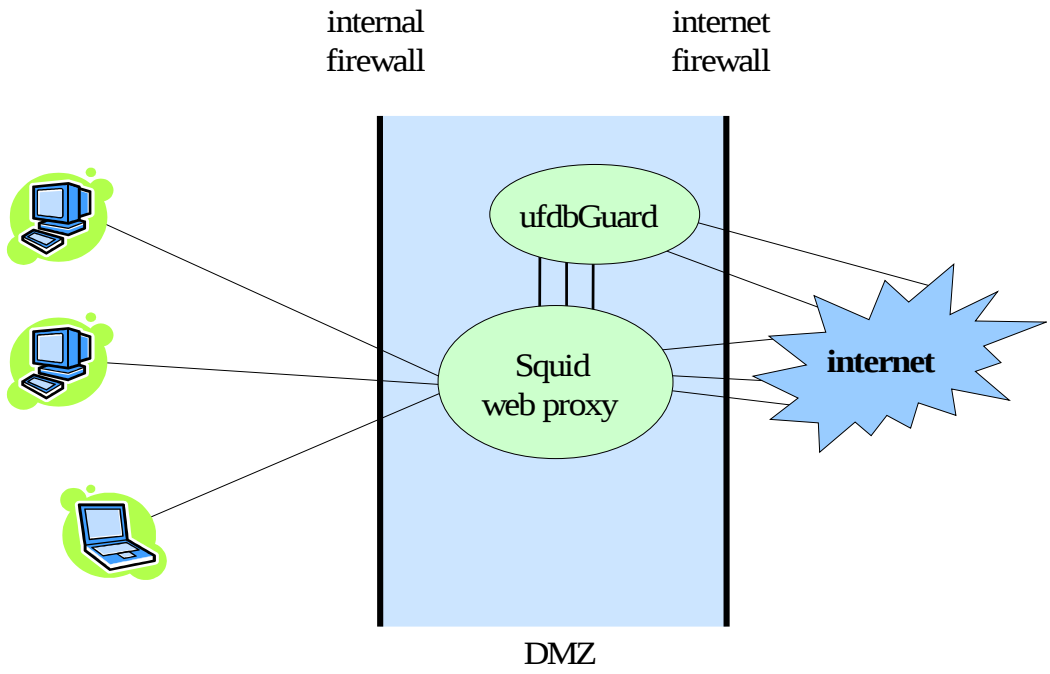


Figure 1: simplified architectural overview

Each request that Squid receives to retrieve a page from a website is first verified with ufdbGuard before Squid tries to fetch the content of the URL.

In case that the daemon is not running or when the daemon is loading a new version of the URL database, all URL verifications get an immediate OK and no web site is blocked. A database reload typically takes between 15 and 25 seconds.

ufdbGuard can be used with any URL database that comes in either a flat file format or the proprietary `.ufdb` format. The ufdbGuard software suite comes with a utility to generate an `.ufdb` database file from a flat file.

3.1 URL redirection

Squid sends each URL that it receives from a browser to ufdbGuard for inspection. If ufdbGuard finds a URL that needs to be blocked according to the local internet usage policy, it sends a substitution URL to Squid. This substitution URL is then used by Squid and content of the substitution URL is sent back to the browser.

A substitution URL is by default similar to this:

```
http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?
category=adult&username=John&url=www.sex.com
```

So for each blocked URL, an appropriate message is retrieved which was generated by `URLblocked.cgi` at `cgibin.urlfilterdb.com`.

For reasons of efficiency, it is recommended to use the lightweight http daemon, `ufdbhttpd`, which has a built-in `URLblocked.cgi`. `ufdbhttpd` is part of the ufdbGuard software suite, can easily serve 700 requests/sec, and is recommended for all environments except the very large ones. Users of ufdbGuard which generate a very large number of hits on `URLblocked.cgi` on the server `cgibin.urlfilterdb.com` will be asked to stop using it.

Very large environments should use a web server on their LAN and install `URLblocked.cgi` on a web server. The web server must be able to incorporate perl scripts.

3.2 Dynamic Proxy Tunnel Detection

The HTTPS protocol is encrypted and designed for secure online transactions like online banking and is considered a safe protocol since it cannot be decrypted. Unfortunately, the HTTPS protocol is also a security risk since anybody can type "proxy tunnel" at Google and find out how to transfer data to and from any system on the internet *bypassing firewalls and internet access security measurements* using the https protocol. This is possible since https is encrypted and most security measurements like anti-virus software, malware detection and intrusion detection systems cannot scan encrypted data.

ufdbGuard has a unique protection mechanism that detects abuse of the HTTPS protocol and blocks all unauthorized transfers of data with these so-called proxy tunnels.

Proxy tunnels can also be used with reverse port forwarding (using ssh) which means that from any system on the internet an unauthorized connection can be made into the protected network. It also does not matter how good the firewall is! As long as HTTPS is allowed and there is not a proper countermeasure against proxy tunnels, a security risk exists.

The ufdbguardd daemon verifies all websites that are accessed with the HTTPS protocol and detects all popular proxy tunnels.

3.3 Enhanced HTTPS Security

As stated earlier, the HTTPS protocol is a useful protocol where secrecy is desired. Financial transactions are a good example where HTTPS is desired. Unfortunately, because HTTPS is encrypted, the HTTPS protocol is also used by applications that may introduce security issues like proxy tunnels, remote access software and viruses. To enhance the security of HTTPS connections on the internet, two configuration options control the use of HTTPS. See section 5.4 for more information.

4 Software Installation

It is assumed that the Squid web proxy is already installed, configured and operational. Please refer to the documentation of Squid to make it operational (see also www.squid-cache.org).

4.1 Upgrading from a Previous Version

A previous version can simply be overwritten by a new version.
CAVEAT: you must stop squid and ufdbGuard to overwrite executables, so you must stop them before a '`make install`'.

When an upgrade is performed, it is recommended to perform all installation steps including the last step, retrieval of database update in section 4.8.

When an upgrade is performed, the default configuration file (`ufdbGuard.conf`) is not installed to prevent loss of previous settings. It is recommended to read the sections about configuration options and to add settings for the new options to the existing configuration file.

NOTE: starting with version 1.23, on systems that support configuration files in `/etc/sysconfig`, a new configuration file `/etc/sysconfig/ufdbguard` is used. Anybody who has modified the start script in /etc/init.d/ufdb may need to edit the new configuration file. During the installation a message reminds the administrator of the new file.

NOTE: starting with version 1.23, the installation defaults have changed. The configure script now has a default installation user `ufdb` (was `squid`) and a new default installation directory

`/usr/local/ufdbguard` (was `/usr/local/squid`). The defaults can be changed with the configuration options `--with-ufdb-user` and `--prefix`.

4.2 User Account

The ufdbGuard software suite should be installed with its own user account, `ufdb`. Formerly, it was recommended to use the username `squid`, but this is now considered inappropriate because two different applications should have two different application owners. There is no technical reason not to use squid as the owner of the ufdbGuard package, so if you do not agree you can overrule the default by using user squid when running configure, see section 4.5 for details.

NOTE: the following steps in this section should be done as the aforementioned user.

4.3 Installation Directory

ufdbGuard can be installed in any directory and by default is installed in `/usr/local/ufdbguard`. In this manual, the word `TOPDIR` refers to the top level installation directory for ufdbGuard.

4.4 Unpack Software

The tar file that contains the ufdbGuard software suite must be unpacked in a source directory of your choice, e.g. `/local/src`.

Unpack the ufdbGuard tar file in the source directory:

```
$ cd /local/src
$ tar xzf ufdbGuard-latest.tar.gz
```

A directory with the name `ufdbGuard-1.24` contains the software.

4.5 Configure the Software Build

Before ufdbGuard is compiled, it needs to be configured and told what the TOPDIR of your choice is:

```
$ cd ufdbGuard-1.24
$ ./configure –-prefix=TOPDIR
```

So, assuming that TOPDIR is `/usr/local/ufdbguard`, the `configure` command becomes:

```
$ ./configure –-prefix=/usr/local/ufdbguard
```

The command `./configure –-help` displays a description on how to configure alternative locations and username.

NEW: UfdbGuard is installed with the assumption that the user `ufdb` is owner of the installed files. If you prefer that ufdbGuard is installed with a different user account, you may use the `--with-ufdb-user` option. The administrator should create such a user account and the configure script reminds the administrator of this. Since previous versions used a default user '`squid`', you may want to use `--with-ufdb-user=squid`.

ufdbGuard will use UNIX sockets whenever they are available. To force the usage of TCP sockets which have more system overhead, use the configure command with the option `--without-unix-sockets`. This option must be used if ufdbGuard runs on a different system then where Squid runs.

The most common error is that the *development* packages for openssl and bzlib are not installed. The configure command checks for the existence of these packages and gives an appropriate message. For most operating systems, one can find the packages *openssl-devel* and *bzip2-devel* on the installation media. Note that each Linux distribution uses different package names and you may find that openssl-devel has an other similar name like *libssl-dev*. Likewise, the bzip2-devel package may be called *libbz-dev*. Refer to the Operating System manual on how to install additional packages.

4.6 Compile Software

Now compile the software suite:

```
$ make all
```

And ignore warnings about compiler options.

4.7 Install Software

In case that you perform an upgrade and squid is already running with ufdbgclient redirectors, squid must be stopped to be able to perform a proper installation. After the installation, squid is started after ufdbGuard.

Stop any running instances of squid:

```
$ su –
# /etc/init.d/squid stop
```

Stop any running instances of ufdbGuard. i.e.

```
$ su –
# /etc/init.d/ufdb stop
```

The programs can now be installed. Note that you must be `root`.

```
# cd .../ufdbGuard-1.24
# make install
```

Note: the installation will look for an old configuration file and will update it and give feedback. The automatic updates are adding new URL categories (extappl and p2p) and replacing references to www.urlfilterdb.com/cgi-bin/URLblocked.cgi with cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi to offload www.urlfilterdb.com. It is recommended to read the output of this installation step and to modify the ACLs where necessary.

It is recommended to start the ufdbguardd daemon at system boot and before Squid is started. The start script is in the `init.d` directory (this may vary and depends on the OS). Optionally modify UFDB_OPTIONS to use the `–T` option to use the test mode.

And then start ufdbGuard and Squid – in this order only.

```
# /etc/init.d/ufdb start
# /etc/init.d/squid start
```

4.8 Get Daily Updates

The script `ufdbUpdate` takes care of downloading a new version of the URL database and signaling ufdbGuard that a new version is downloaded. Daily updates are available for everyone with a permanent or trial license of URLfilterDB. The `ufdbUpdate` script should be run twice per day by the cron scheduler. There are 3 things to do:

- make sure that the `ufdb` user is allowed to run crontab jobs (`crontab –l` run as user `ufdb` should give no errors)
- enter the username and password in `ufdbUpdate` that you received when you purchased a license or received a trial license.
- choose an appropriate time at the end of the day and in the early morning to run `ufdbUpdate`

Edit the `ufdbUpdate` script to include the username and password that you received when the (trial) license was received:

```
$ vi /usr/local/ufdbguard/bin/ufdbUpdate
...
DOWNLOAD_USER=lic99999
DOWNLOAD_PASSWORD=aa22bb
```

Users that evaluate the URL database may use the demoXX username and password.

Test the ufdbUpdate script with the verbose option:

```
# su – squid
$ /usr/local/ufdbguard/bin/ufdbUpdate –v
```

The output should be similar to:

```
new database downloaded:
-rw-r--r-- 1 root root  5121312  Dec 6 18:04 /tmp/urlfilterdb-latest.tar.gz
done.
```

To install the cron job, edit the crontab table of the user `ufdb` and add the appropriate lines:

```
$ crontab –e
```

To run the URL database update each day at 10:00 PM and 6:15 AM, add these two lines:

```
00 22 * * * /usr/local/ufdbguard/bin/ufdbUpdate
15 6  * * * /usr/local/ufdbguard/bin/ufdbUpdate
```

At this time, you may want to verify that the Squid housekeeping is also executed. Verify that the crontab for user `squid` has an entry to run "`squid –k rotate`" (see the squid manuals for more details). It is a common misconception that this command only rotates the logfile of Squid but it also does necessary housekeeping. Note that the Squid housekeeping must be done when there is almost on load (e.g. at 03:00 AM).

4.8.1 Exit Codes of ufdbUpdate

To monitor URL database updates, ufdbUpdate has a defined set of exit codes.

code	*explanation*
0	all OK
1	version mismatch warning; most likely there is a new version of ufdbguardd
2	license expiration warning: less than 2 months to renew license
3	license expired: a license renewal is required immediately
11	configuration error
12	temporary file error
13	download OK but cannot signal ufdbguardd
21-40	exit code of ufdbUpdate is exit code of wget + 20. wget is the command that downloads the new URL database from the servers of URLfilterDB.
41-60	exit code of ufdbUpdate is exit code of gunzip + 40. gunzip uncompresses the downloaded URL database. There may be an issue with file system space.

61-80	exit code of ufdbUpdate is exit code of tar + 60. tar unpacks the downloaded URL database. There may be an issue with file system space.

In case of an error, it is advised to run `ufdbUpdate -v` from the command line to have more feedback about what is going wrong. License expiration warnings are also issued bu ufdbguardd.

4.9 Firewall and Proxy Settings

4.9.1 ufdbGuard

ufdbGuard send probes to HTTPS sites to detect tunnels, detect proxies, detect Skype, and verifies SSL certificates. It also sends statistics to the servers of URLfilterDB. Since tunnel and proxy detection are considered an important feature, it is strongly recommended to modify firewall rules to provide access.

ufdbGuard needs access to port 443 for all internet addresses and needs access to port 80 of updates.urlfilterdb.com.

4.9.2 ufdbUpdate

ufdbUpdate downloads the URL database and obviously needs access to the servers of URLfilterDB. Firewall rules may need to be modified to provide access to `updates.urlfilterdb.com`.

A proxy can be used to download the URL database: edit the ufdbUpdate script and assign the appropriate values to the variables http_proxy, PROXY_USER and PROXY_PASSWORD.

4.10 Database Updates from Other Sources

In case that a URL database is used that is not provided by URLfilterDB, the URL database is most likely a collection of flat files with domain names and URLs. In this case the user should implement itself a method for getting regular database updates. After downloading a new flat file URL database, the database must be converted to the proprietary database format of ufdbguardd using the `ufdbConvertDB` tool.

4.10.1 Converting a URL Database

Version 1.22 and later of the ufdbGuard software suite has a new utility to convert a whole URL database without the need to call ufdbGenTable for each URL category: ufdbConvertDB. ufdbConvertDB requires one parameter which is the top level directory name where the URL database resides.

Example:

```
# /usr/local/ufdb/bin/ufdbConvertDB /usr/local/ufdb/blacklists
```

5 Configuration

This chapter and chapter 7 and 8 describe the features and configuration syntax for ufdbGuard in detail. Chapter 9 describes the GUI for the features. It is recommended to read first chapters 5, 7 and 8 to get a better understanding of all features before one starts to use the GUI.

5.1 Know your Internet Usage Policy

Before one can start with the configuration of ufdbGuard, an internet usage policy must already be defined and state which web site categories are considered unwanted and therefore must be blocked by ufdbGuard. The HR department might be a good starting point to find out which categories of the database must be used to block access to parts of the internet.

Note: the default configuration of ufdbGuard blocks only adult, security, p2p, proxies, gambling and warez. By default, ufdbGuard blocks all unsafe HTTPS connections with the options enforce-https-with-hostname, enforce-https-official-certificate and https-prohibit-insecure-sslv2.

The internet usage policy may block certain categories which need exceptions. E.g. the internet usage policy may prohibit visiting shops but an authority may decide to to allow access to the local pizza delivery – which is usually included in the category shops – to provide a meal for those who do overtime. Or a policy may allow access to shops but block one or two particular shops. The URL categories *alwaysallow* and *alwaysdeny* are used to define these exceptions and contains URLs that should be allowed or denied according to the local policy. Section 7 explains on how to configure these categories.

ufdbGuard supports time-based ACLs which enable the implementation of internet usage policies that have different policies during different time of the day or week. With time-based ACLs it is, for example, possible to define that after working hours a different – usually more relaxed – policy is used. See section 8.4 for more information.

It is recommended to start with the URL filter in the test mode. In this mode, sites are not blocked but only logged in the log file. An analysis of the log file may help in defining the appropriate Internet Usage Policy and to find out which sites should never be blocked (see also section 7.1). At last a free tip: to prevent a storm at the help desk or system administrator, it is advised to inform users about (change in) the implementation of a URL filter *before* it is implemented and to allow users to request to define exceptions with the alwaysallow category.

The ufdbGuard software suite includes a tool to find out what type of sites are currently visited by analyzing the Squid log file. See section 11 for a detailed description of ufdbAnalyse. This tool can also be used in determining or adjusting the Internet Usage Policy.

5.2 Automatic Improvements of the URL Database

Although the URL database is updated daily, it may happen that some web sites are not included in the URL database and therefore users might be able to visit inappropriate websites. UfdbGuard has a configuration option to collect a sample of these uncategorized domains and to upload them to URLfilterDB for inclusion in the database. This option is ON by default. Privacy is guaranteed since no identification of client, source or user is included, just the domain name is registered for categorization. URLfilterDB has a privacy policy which prohibits sharing of any information with 3rd parties and all software tools of URLfilterDB strip all parts of (intermediate) data which may contain privacy-related items such as IP address, user name, password and query parameters. See also http://www.urlfilterdb.com/en/privacystatement.html .

To prevent analysis of uncategorized URLs, the `analyse-uncategorised-urls` configuration option must be set to `off`. It is, however, highly recommended to leave this option ON to receive more updates in the URL database which results in better filtering. URLfilterDB tries to categorize URLs as soon as possible and to include them in a new URL database.

5.3 Proxy Tunnel Detection

Proxy tunnels are a security risk and it is strongly recommended to use detection of proxy tunnels. Enable proxy tunnel detection in the configuration file `ufdbGuard.conf` with the following line:

```
check-proxy-tunnel queue-checks
```

This option queues checks for proxy tunnels to be detected a few seconds later. It means that a proxy *can* be used but only for 1-2 seconds. Alternatively, proxy tunnels can be detected in an aggressive mode, where all HTTPS traffic is tested for proxies *before* access is given. The aggressive mode introduces some delays for HTTPS traffic and is therefore only recommended in very high security environments. If proxy tunnels are allowed, the value for check-proxy-tunnel can be `off` or `log-only`. All valid options are:

```
check-proxy-tunnel queue-checks     # recommended
check-proxy-tunnel aggressive
check-proxy-tunnel log-only
check-proxy-tunnel off
```

NOTE: proxy tunnel detection depends on the usage of the category *proxies*. Only when the URL category proxies is blocked, the proxy detection is performed.

NOTE: when the "aggressive" detection method is used, the number of threads in the ufdbguardd daemon and the number of redirector processes must be increased to 64 (maximum is 128). Therefore, change in squid.conf the parameter `url_rewrite_children` to 64 and restart squid. To start the ufdbguardd daemon with 128 worker threads, the command line option `-w 128` must be used. To use this option, the variable UFDB_OPTION in `/etc/sysconfig/ufdbguard` (or the startup script) must be set to "-w 128" (maximum is 128).

5.4 Control HTTPS Usage

Most websites that use HTTPS for legitimate business reasons use an SSL certificate that is signed by a well-known certificate authority and have a fully qualified domain name in the URL for maximum security and a clear identification of the website, while most websites that use HTTPS for other reasons, have self-signed SSL certificates and IP addresses instead of domain names. HTTPS is usually secure enough to protect the connection to eavesdropping but has an old protocol option which is rarely used and insecure. The old and insecure SSLv2 protocol can be blocked by means of a configuration option. Access to HTTPS websites can be controlled with 3 options.

The following options are default in ufdbGuard.conf:

```
enforce-https-with-hostname on
enforce-https-official-certificate on
https-prohibit-insecure-sslv2 on
```

It is recommended to keep these options set to "on" to have a increased protection level against phishing sites, proxies and websites with untrusted SSL certificates. In case that a legitimate website uses an IP address in the URL or an SSL certificate that is not signed by a trusted authority, it is recommended to add this site to the locally trusted websites (see section 7.1).

It is mandatory to use these three options *inside* the definition of the category "security":

```
category security
{
   domainlist "security/domains"
   option     enforce-https-with-hostname on
   option     enforce-https-official-certificate on
   option     https-prohibit-insecure-sslv2 on
   redirect   ...
}
```

Finally, include the category security into the ACLs (see section 5.6) to block HTTPS abuse.

Note: the option to verify HTTPS certificates only works when the URL database of URLfilterDB is used since the URL database includes the necessary SSL certificates to verify HTTPS websites. The CA certificates are in the file `.../security/cacerts` and the keyword `cacerts` that can be used inside the security category definition can be used to use an alternative file with CA certificates. The configuration file looks like this:

```
category security
{
   domainlist “security/domains”
   cacerts    “/usr/local/ufdbguardd/cacerts”
   ...
```

5.5 Which HTTP Server to use for Redirection Messages

Whenever a website is blocked, Squid receives a redirection URL from ufdbguardd. This URL must point to a web server that can display an appropriate error message. Version 1.15 introduced a new lightweight http daemon with a built-in URLblocked.cgi: `ufdbhttpd`.

There are 3 options for serving redirection messages:

1. use `ufdbhttpd` on a non-privileged port on host *myhost.mylan*:
   `http://myhost.mylan:8080/cgi-bin/URLblocked.cgi`
2. use the perl script `URLblocked.cgi` on your own internal web server:
   `http://myhost.mylan/cgi-bin/URLblocked.cgi`
3. use `URLblocked.cgi` of URLfilterDB:
   `http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi`

Option 1 is preferred for all but very large environments. Option 2 is preferred for very large environments. You need to have a web server like Apache with mod_perl or FastCGI enabled to implement option 2.

The default configuration file comes with option 3 preconfigured. If option 1 or 2 is used, the `redirect` statements must be changed accordingly. E.g., change the lines with

```
redirect  http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?color=orange&...
```

into

```
redirect  http://myhost.mylan:8080/cgi-bin/URLblocked.cgi?color=orange&...
```

ufdbhttpd is lightweight and can be installed on the same server where ufdbguardd is installed.

For option 2: the perl script URLblocked.cgi is included in the distribution in the directory `samples`. The perl script uses some images which are also in the samples directory and which need to be installed on the webserver. The reader is referred to the documentation of the used web server on how to install this perl script.

5.6 Main Configuration Settings ufdbGuard

Use your favorite editor to edit the configuration file and define which categories must be blocked.

```
$ vi /usr/local/ufdbguard/etc/ufdbGuard.conf
```

There are 5 sections with a comment ‘EDIT THE NEXT LINE’. Find each section and change the configuration where needed.

The first section defines the locations of the log directory and the blacklist database directory. This section does not have to be changed if the initial choice of these locations during the configuration of ufdbGuard (see section 4.5) has not changed. The section looks like this:

```
# EDIT THE NEXT LINE FOR LOCAL CONFIGURATION
dbhome /usr/local/ufdbguard/blacklists
logdir /usr/local/ufdbguard/logs
```

The second section defines the IP address range of your local network. The section looks like this:

```
source allSystems {
   ip 10.0.0.0/8
}
```

The appropriate network subnet must be entered in this section. `10.0.0.0/8` and `192.168.0.0/16` are the most common values for this. Consult your network administrator for assistance.

The third section defines the usage and settings of `ufdbhttpd`, the lightweight http daemon:

```
http-server { port= 8080, interface= all, images="/usr/local/ufdbguard/images" }
```

If you don not use ufdbhttpd, then transform the line above into a comment (with a #).

The fourth and fifth sections are close to each other and define the list of categories to be blocked (one list for the systems with IP address defined in `allSystems` and one list for all other systems). Change the list of categories to be blocked. The default list of blocked categories contains the categories security, adult, p2p, proxies, gambling, violence and warez. The fourth section looks like this:

```
acl {
   allSystems {
      # EDIT THE NEXT LINE
      pass !security !adult !p2p !proxies !gambling !violence !warez any
   }
```

To block a category, it needs to be present with an exclamation mark (!) that is used as a blocking indicator. So to block the *adult* category, `!adult` must be present in the line that starts with `pass`. If you prefer to allow gambling, the definition "`!gambling`" must be removed.

At a site that only blocks security, adult, p2p and proxies, the section looks like this:

```
acl {
   allSystems {
      # EDIT THE NEXT LINE
      pass !security !adult !p2p !proxies any
   }
```

The fifth section is very similar to the fourth section and defines which categories to block for computer systems that are not part of `allSystems`.

The configuration for the use of Skype is more complex. Read section 8.6 on how to allow use of Skype.

5.7 Ensure Access to Internal and 3rd Party Websites

The domain name of your company may be included in the URL database of URLfilterDB. To ensure access to all own websites, the category *alwaysallow* should be configured (see section 7.1).

To prevent unhappy users, one should carefully examine which own sites and sites of 3rd parties are used for regular daily activities and make sure that these sites can be accessed without restriction. Therefore, it is recommended to run a few days in test mode (see section 5.10.1) and add sites of important 3rd parties to the category *alwaysallow* (see section 7.1).

5.8 Redirection Message Variables

A redirect statement may contain variables that are dynamically expanded to their values. E.g. `%u` is substituted by the URL string that is blocked. So when www.adult.com is blocked, the redirection URL

`http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?`*`url=%u`*

is dynamically expanded to

`http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?url=`***`www.adult.com`***

The variable *%u* is associated with the parameter *url*. The parameters are interpreted by the URLblocked.cgi script and used to produce an appropriate HTML message for the internet browser of the end user. The following table contains the list of all redirection message parameters.

parameter	*variable*	*explanation*
admin	%A	administrator
clientaddr	%a	source address (IP or FQDN)
clientuser	%i	username
clientname	%n	source domainname
clientgroup	%s	source identifier
category targetgroup	%t	URL category
url	%u	URL
url	%U	parsed URL
-	%%	the literal character '%'

redirection message parameters & variables

5.9 Redirection Message Styles

When a URL is prohibited to be visited, a message is displayed that access is forbidden. The style, size and background color can be set by the administrator.

The options for the background color are: orange (default), white, black, grey and red. The options for the font size of the message are: normal (default), small and large.

To change the style of a message, edit the configuration file and change the default settings of all `redirect` rules.

```
$ vi /usr/local/ufdbguard/etc/squid.conf
...
redirect http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?admin=
%A&mode=default&color=orange&size=normal&clientaddr=%a&clientname=%n&clientuser=
%i&clientgroup=%s&targetgroup=%t&url=%u
...
```

Substitute the default value for a new value according to the options specified earlier in this section.

The default error message includes an explanation why the URL is blocked, the URL, the email address of the administrator and a link for more information. Alternatively, the output can be configured to be very simple and just consist of the word “Forbidden” (in different languages) and the category of the URL in red. To use this style one should set the *mode* parameter to `simple-red`.

The *mode* parameter can have additional values specific for advertisements only. Valid values for *mode* are: transparent (default), noads, square, cross and simple-red. The transparent mode displays nothing, the noads mode displays “no ads”, the square mode shows a grey square, the cross mode shows a transparent cross, and the simple-red mode shows a minimalistic “*ads*”.

The *color* parameter defines the background color and determines automatically an appropriate foreground color. The *size* parameter defines the size of the text.

parameter	*value*
`mode`	default noads square cross simple-red
`color`	white black grey orange red
`size`	small normal large

redirection style parameters

5.10 Configure Squid

To let Squid use the URL filter (in Squid terminology the URL filter is a *redirector* or a *URL rewriter*), the following 2 parameters must be added to the squid configuration file.

```
$ vi /usr/local/ufdbguard/etc/squid.conf
```

Add the following 2 lines for Squid **2.6** and newer:

```
url_rewrite_program /usr/local/ufdbguard/bin/ufdbgclient -l /usr/local/ufdbguard/logs
url_rewrite_children 64
```

NOTE: by default, the `ufdbguardd` cannot support more than 64 `ufdbgclient` processes so do not use more than 64 url_rewrite_children. Consult the support desk of URLfilterDB in case that you find that the number of URL rewriters is insufficient.

5.10.1 Test Mode

ufdbGuard has a test mode where internet access is not blocked and where the log file contains lines which web sites would have been blocked in normal mode.

In case that you want to use the test mode, add the `-T` option for `ufdbguardd`. The place to set the options is in `/etc/sysconfig/ufdbguard` on systems that have `/etc/sysconfig` or `/etc/init.d/ufdb` (might be elsewhere for your OS). The file should then have a line like this:

```
UFDB_OPTIONS="-T"
```

In test mode, the log file of ufdbguardd contains lines like this:

```
TEST BLOCK adult www.sex.com
```

5.10.2 Configuration for 2 servers

In case that ufdbguardd runs on a different system then where Squid runs, the server name and port number need to be configured for `ufdbgclient` with the following options:

```
-S servername -p 3977
```

The `squid.conf` file should then have a line like this:

```
url_rewrite_program /local/squid/bin/ufdbgclient -S urlchecker01 -p 3977
```

Also at configuration time, it is required to define the use of 2 servers (see section 4.5).

5.11 Configuration of Browsers

The browsers like Internet Explorer, Firefox, Opera and others need to be configured to

- use the Squid proxy
- display original error messages.

5.11.1 Configure Browser to use Squid

How to configure the browser depends much on the chosen technical infrastructure. If Active Directory Server is used, it can be used to configure browsers to use the Squid proxy as the HTTP and HTTPS proxy. Alternative browser can be configured to automatically detect a proxy using the WPAD[1] discovery mechanism or can be configured manually.

5.11.2 Configure Internet Explorer to Display Site Error Messages

Internet Explorer of Microsoft has a feature to use an alternative method of displaying HTTP errors which can be confusing because the option ignores any error text produced by a HTTP server and substitutes it by a standard text. It is highly recommended to configure the browser to display the original error messages by *unchecking* the following option:

Internet Options – Advanced – Show friendly HTTP error messages.

5.12 Monitoring of ufdbGuard

ufdbGuard uses 2 log files and the system log to write messages to. The location of the log files depends on the choice that was made during the configuration phase (see section 4.5). The names of the log files are `ufdbguardd.log` and `ufdbgclient.log`. The location and name of the system log is OS-dependent and is usually `/var/log/messages` or `/var/adm/syslog.log`.

It is recommended that the log files are regularly inspected, either manually or automatically. Serious errors have a 5-star indicator (`*****`).

5.13 Compatibility between ufdbGuard and squidGuard

ufdbGuard 1.0 was based on squidGuard 1.2.0 and a lot has been changed since then. ufdbGuard has new features that squidGuard lacks: *SafeSearch*, *safer HTTPS options*, *logfile rotation*, *HTTP daemon*, *configurable error messages*, *analysis of uncategorised URLs*, and *CPU binding*. ufdbGuard does not support the squidGuard features *user quota*, *ldap*, *mysql* and *separate logfiles*. Despite the differences, a configuration file of squidGuard works with ufdbGuard with little or no modifications.

ufdbGuard uses the more logical keyword `category` where squidGuard uses `destination`. For reasons of compatibility, ufdbGuard recognizes the `destination` keyword.

ufdbGuard has a different database format than squidGuard and has the utility `ufdbConvertDB` to convert the flat files of the database of squidGuard to the database format of ufdbGuard.

6 Start the URL Filter

To start the URL filter daemon:

```
# /etc/init.d/ufdb start
```

Now you can restart squid to use the URL filter:

```
# /etc/init.d/squid reload
```

or

[1] See for example http://en.wikipedia.org/wiki/Web_Proxy_Autodiscovery_Protocol for more information on WPAD.

```
# /etc/init.d/squid stop
# /etc/init.d/squid start
```

7 Exception Rules

In cases where exceptions to the categories of URLfilterDB are desired, an administrator can define 2 extra categories that are managed by the administrator and never by URLfilterDB.

7.1 Allow Extra Sites

A common case is that you want to ensure access to your own websites and websites of 3rd parties that are used for normal activities. To grant users access to the company websites, the URL `yourcompany.com` needs to be added to the category *alwaysallow*.

For universities that want to allow access to all other universities in the United States, a simple `edu` in the alwaysallow list. In the UK, only `ac.uk` needs to be included!

Edit the file that contains the extra sites that should always be allowed:

```
$ cd /usr/local/ufdbguard
$ vi blacklists/alwaysallow/domains
```

Add the appropriate URLs and always remove a leading `www.`:

```
yourcompany.com
news.google.com
google.com/news
```

In case that you have a file with many URLs having a leading `www.`, you may use the -W option to remove the `www.` prefix automatically.

Additional domains can be added according to the local internet usage policy. For example, if news should be blocked but access to CNN allowed, then `cnn.com` should be added also. Alternatively, when news should be blocked but Google news allowed, `news.google.com` and `google.com/news` should be added.

ufdbGuard only uses proprietary database files, so generate an `.ufdb` database file from the ASCII file with `ufdbGenTable`:

```
$ cd /usr/local/ufdbguard
$ bin/ufdbGenTable -W -n -t alwaysallow -d blacklists/alwaysallow/domains
```

The above command generates the file `blacklists/alwaysallow/domains.ufdb` and should be invoked each time that the domains file is changed. `ufdbGenTable` has a sanity check option (`-s`) which performs a few checks on the validity of the URLs. It is recommended to use this option. The `-W` option removes the initial www. from all URLs.

Then activate the category by editing the `ufdbGuard.conf` file and uncomment the category definition for *alwaysallow*. The configuration file should have the following lines:

```
category alwaysallow
{
   domainlist alwaysallow/domains
   redirect ...
}
```

Also add the category alwaysallow to the ACL `allSystems`. The ACL should then start with

```
   pass alwaysallow !adult ...
```

Finally signal the ufdbguardd daemon about the new configuration:

```
# /etc/init.d/ufdb reconfig
```

7.2 Block Extra Sites

In case that you want to block access to a site that is not in any category, you can add this site to the category *alwaysdeny*. For example, `google.com` is not in any category but if you like to block access to this popular search engine, `google.com` can be included in the alwaysdeny category. Analogous to the alwaysallow category, the domain (without leading `www.`) must be added to the category domains file, and the ACL `allSystems` must be extended.

Edit the file that contains the extra sites that should always be blocked:

```
$ cd /usr/local/ufdbguard
$ vi blacklists/alwaysdeny/domains
```

Add the appropriate URLs (always remove a leading `www.`):

```
google.com
```

ufdbGuard only uses proprietary database files, so generate an `.ufdb` file from the ASCII file with `ufdbGenTable`:

```
$ cd /usr/local/ufdbguard
$ bin/ufdbGenTable -W -n -t alwaysdeny -d blacklists/alwaysdeny/domains
```

The `-W` option removes the initial www. from all URLs. The above command generates the file `blacklists/alwaysdeny/domains.ufdb` and should be invoked each time when the domains file is changed. Then activate the category by editing the `ufdbGuard.conf` file and uncomment the category definition for alwaysdeny. The configuration file should have the following lines:

```
category alwaysdeny
{
   domainlist alwaysdeny/domains
   redirect ...
}
```

Also, add the category alwaysdeny to the ACL `allSystems`. The ACL should then start with

```
   pass alwaysallow !alwaysdeny !adult ...
```

Finally signal the ufdbguardd daemon about the new configuration:

```
# /etc/init.d/ufdb reconfig
```

7.3 Block Extra Sites – a more Advanced Example

If *parts* of a website need to be blocked, the method described in the previous section does not provide a solution. Suppose we allow users to watch CNN news but do not want them to look at the CNN news videos.

It was observed that the CNN videos are made available through the following URLs:

```
http://www.cnn.com/video
http://edition.cnn.com/video
http://vid.cnn.com
```

Since access to www.cnn.com is allowed and access to www.cnn.com/video is not, a *domain block* is not an option, but a so-called *url block* is. Therefore, in this example the above URLs need to be divided into

2 categories: those which can be blocked with domain blocks and those which can be blocked with url blocks.

The block referring to a domain goes into the file …/alwaysdeny/domains:

```
vid.cnn.com
```

and the blocks referring to a full URL go into the file …/alwaysdeny/urls:

```
cnn.com/video [2]
edition.cnn.com/video
```

The `ufdbGenTable` command has a `-u` option to include the url blocks and the command to generate a new `.ufdb` file including all options becomes this:

```
$ cd /usr/local/ufdbguard/blacklists
$ ../bin/ufdbGenTable -W -n -t alwaysdeny -d alwaysdeny/domains -u alwaysdeny/urls
```

Do not forget to signal the ufdbguardd daemon about the new configuration:

```
# /etc/init.d/ufdb reconfig
```

8 Advanced Options

8.1 Blocking Adult Images produced by Search Engines

Search engines like Google, Yahoo and MSN have a capability to search for images and allow users to view adult images that can not be blocked in a simple way since the images come from Google, Yahoo and MSN themselves and in general one would not like to block all images from the search engines.

Google[3], Yahoo and other search engines offer a S*afeSearch*™ feature which blocks most adult images. UfdbGuard has the configuration parameter `safe-search` that enforces the safesearch policies of the search engines. The default value for the parameter is ON.

The safesearch feature enforces safe searches for the following search engines: A9, Alltheweb, Ask, Bing, Blinkx, Buscamundo, Dogpile, Excite, Foxnews, Google, Infospace, Live, Lycos, Metacrawler, Metaspy, MSN, Terra, Webcrawler, webpile, Yahoo and other less popular search engines.

8.1.1 Global SafeSearch Option

To enforce SafeSearch for all users, use

```
safe-search on
```

8.1.2 Per-ACL SafeSearch Option

To have fine-grained control over the SafeSearch feature, the global safe-search feature must be turned off and a category must be defined that includes the safe-search option. The newly created category can be used in the ACLs as demonstrated in the following example.

```
safe-search off
...
category safesearch {
   option safe-search on
}
...
```

[2] Note that there is no leading `www.` since the URL filter works internally with stripped URLs that do not contain a `www.` part. Use the -W option for files with URLs that have a www. prefix.

[3] Google SafeSearch filtering is a trademark of Google.

```
acl {
   ...
   allSystems {
      pass !alwaysdeny !proxies ... !safesearch any
   }
```

8.2 Different Policies for Different Users

Suppose that ufdbGuard is used in a bank. The internet usage policy could be defined as: block sex, chat, dating, entertainment, finance, news, webmail. This policy can be appropriate for most users but not for staff working in a dealing room where access to news and finance-related sites is required. This section explains how to achieve this.

Always be careful with the order of rules and make sure that the more privileged user (groups) is defined first in the ACL.

8.2.1 Policy based on IP Address

If the dealing room has a separate IP subnet, the *dealing room* policy can be defined in the following way in the ufdbGuard configuration file.

```
source dealingroom {
   ip 10.4.0.0/16      # e.g. the dealing room uses this subnet
}

acl {
   # more privileged users first
   dealingroom {
      pass !security !adult !warez any
   }
   allSystems {
      pass !security !adult !p2p !proxies !dating !entertain !warez any
   }
   default {
      pass none
      # the following redirect is for the pseudo category 'none'
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
}
```

8.2.2 Policy based on Username

Internet access policies can also be based on an individual user or a list of usernames. In this example, a list of usernames is used to use the dealingroom policy. The `source` definition defines which users are members of the group *dealingroom*.

```
source dealingroom {
   user      "admin"
   userlist  "/usr/local/ufdb/etc/dealers"
}
```

The file `/usr/local/ufdb/etc/dealers` should contain the usernames of all dealers. The final policy definition is the same as a policy based on IP address:

```
acl {
   # more privileged users first
   dealingroom {
      pass !security !adult !proxies !dating any
   }
   allSystems {
      pass !security !adult !proxies !chat !dating !entertain !webmail any
   }
   default {
      pass none
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
}
```

CAVEAT: additional configuration is required to make Squid able to find out which users are using Squid. You may configure squid to use user authentication or you have to install `identd` on all PCs to support this feature. Please read the Squid documentation for more information and do not forget to use `acl` *`foo`* `ident REQUIRED`.

8.2.3 Policy based on UNIX Group name

Internet access policies can also be based on a UNIX group. In this example, a groupname is used to use the dealingroom policy. In this case, the `source` definition defines that the group `dealer` represents the dealingroom.

```
source dealingroom {
   unix group dealer
}
```

The group `dealer` must be a valid UNIX group name. The final policy definition is the same as a policy based on IP address:

```
acl {
   # more privileged users first
   dealingroom {
      pass !security !adult !proxies !dating any
   }
   allSystems {
      pass !security !adult !proxies !chat !dating !entertain !webmail any
   }
   default {
      pass none
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
}
```

CAVEAT: additional configuration is required to make Squid able to find out which users are using Squid. You may configure squid to use user authentication or you have to install `identd` on all PCs to support this feature. Please read the Squid documentation for more information and do not forget to use `acl` *`foo`* `ident REQUIRED`.

8.2.4 Policy based on Domain Name

Internet access policies can also be based on the domain name of a PC. In this example, the PCs in the dealingroom have a unique domain name that can be used to define a policy.

Let's assume that the dealingroom PCs have a name with the same subdomain like pc31.dealingroom.bank.com, and then the `source` definition for the `dealingroom` group is as follows:

```
source dealingroom {
   domain "dealingroom.bank.com"
}
```

NOTE: To use this feature, the reverse name lookup must be enabled in `squid.conf`:

```
log_fqdn on
```

8.3 Multiple ACLs

In case that there are exceptions for (groups) of users, multiple ACLs will be used (see also the examples for a dealing room in the previous sections).

The order of the definition of sources is important since the *first* source that matches a URL/user/IP is used to determine a block or a pass and no further sources are taken into account. The following example demonstrates this.

Suppose there is an administration PC that should have access to all websites and that the PC has a fixed IP address: 10.2.3.4. The configuration file should have the source definition for the administration PC *in front of* the source allSystems. Otherwise, the administration PC is considered part of allSystems and the ACL rule for adminpc is never used to decide a block or pass. So the order of the definitions is always important.

```
source adminpc {
   ip 10.2.3.4
}
source allSystems {
   ip 10.0.0.0/8
}
acl {
   adminpc {
      pass any
   }
   allSystems {
      pass !security !adult !proxies !chat !dating !entertain !webmail any
   }
   default {
      pass none
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
}
```

Note that the `redirect` statement is required because the pseudo-category `none` is used.

8.4 Time-Based ACLs

ufdbGuard supports time-based ACLs which enable the implementation of internet usage policies that have different policies during different time of the day or week. As the first step, a definition of one or more time intervals is given with the `time` statement. A time interval definition can contain two types of

time intervals: the `weekly` directive is used to define reoccurring time intervals and the `date` directive is used to define special dates. The syntax is explained with the following example.

```
time "working-hours" {
   weekly mon,tue,wed,thu,fri  08:00 - 19:00
   weekly sat,sun              08:00 - 12:30
   date   *-*-01   # every first of the month
   date   2010-12-31
}
```

Weekly reoccurring hours have one or more days separated by commas or a wild character, followed by a time interval: *HH:MM - HH:MM*. The names of the days may be replaced by the wild card "*" to denote all days. Dates have the format *YYYY-MM-DD* and may contain a wild character for the year, month and/or day.

Once the time intervals are defined, they can be applied to the ACLs with the directives `within`, `outside` and `else`. The following example shows how these directives can be used.

```
acl {
   allSystems within "working-hours" {
      pass !security !adult !proxies !p2p !dating !entertain !webmail any
   } else {
      pass !security !adult !proxies !p2p any
   }
   ...
```

In the above example users are not allowed access to sites in the categories dating, entertainment and webmail during office hours. The `else` part defines an alternative ACL for all other hours that are not inside the *working-hours* time interval. The `else` part is optional.

As an alternative to the `within` directive, the `outside` directive can be used which has opposite semantics.

8.5 Whitelisting

Whitelisting is used in case that users are only allowed to visit a predefined set of websites. In this case, the ACL for allSystems contains the categories *alwaysallow* and *none*. The ACL in the configuration file looks like this:

```
acl {
   allSystems {
      pass alwaysallow none
      # the following redirect is for the pseudo category 'none'
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
   default {
      pass none
      redirect http://cgibin.urlfiterdb.com/cgi-bin/URLblocked.pl?...
   }
}
```

Note that the extra `redirect` statement is required because the pseudo-category `none` is used.

8.6 Allowing Skype

Note: Skype is a VOIP application that also can be used to transfer files and give remote access using screen sharing. These features *may* violate the local Internet Usage Policy.

Skype is a popular VOIP application that requires a more complex configuration because it uses the HTTPS port 443 for its proprietary protocol, which would be blocked if the options for safer HTTPS are used. Skype also uses IP addresses instead of hostnames, so Skype conflicts with the options `enforce-https-with-hostname` and `enforce-https-official-certificate`. ufdbGuard can detect the use of Skype and allow Skype and at the same time block other HTTPS traffic that does not comply with the options that enforce proper HTTPS.

For those who want to use Skype and have a safer use of HTTPS, a new option has been introduced in version 1.23: `allow-skype-over-https`. An extra category needs to be configured:

```
category chat-skype {
   domainlist      chat/skype/domains
   expressionlist  chat/skype/expressions
   option          allow-skype-over-https on
   redirect        ...
}
```

Note that he category `chat-skype` must be placed *before* the category `security` in the ACLs to prevent that the security category blocks access to Skype URLs, e.g.:

```
acl {
   allSystems {
      pass alwaysallow !alwaysblock chat-skype !security !proxies ...
      ...
```

Skype applications need to be configured to use Squid as its proxy: menu Options, tab Advanced, tab Connection, HTTPS proxy.

8.7 Default blocking behavior

8.7.1 ufdbguardd

The URL filtering system has two conditions where it cannot perform the regular filtering function:

a) whenever there is a fatal error

b) when the URL database is reloaded.

By default, whenever one of the 2 conditions occur, ufdbguardd sends a message to Squid telling Squid that the URL is not filtered, i.e. allowed to see. The only possible alternative to allowing all URLs, is to block all URLs. For those environments where it is necessary to block all URLs under the above conditions, the following 2 configuration settings can be set in the configuration file:

```
url-lookup-result-during-database-reload  deny
url-lookup-result-when-fatal-error        deny
```

When any of the above two parameters is set to “deny”, browsers are redirected and display an appropriate message. To overrule the default messages one can configure URLs that point to web pages to display own messages for fatal errors and when it is loading a database with the following two parameters.

```
redirect-fatal-error       "http://www.example.com/fatalerror.html"
redirect-loading-database  "http://www.example.com/loadingdb.html"
```

8.7.2 ufdbgclient

ufdbgclient is the glue between Squid and ufdbguardd. By default, when ufdbgclient cannot communicate with ufdbguardd, it allows all URLs, i.e. all URL filter requests of Squid are answered with a message indicating that the URL is allowed. The only possible alternative to allowing all URLs is to block all

URLs. To block all URLs when ufdbguardd is not running or there is a fatal communication error, ufdbgclient can be started with the "`-e deny`" option.

In case of a fatal error, ufdbgclient returns by default the URL `http://cgibin.urlfilterdb.com/cgi-bin/URLblocked.cgi?category=fatal-error` which displays an appropriate message. To overrrule the default redirection URL for fatal errors, the `-E` option can be used which takes a valid URL as its argument, e.g. `-E http://example.com/error.html`

The `-e` and `-E` options must be specified in the configuration file of Squid, `squid.conf`. E.g.:

```
url_rewrite_program /local/squid/bin/ufdbgclient -e deny
```

8.8 Extended Logging

ufdbGuard has 2 options for extended logging. The keyword `logblock` followed by `on` or `off` tells ufdbGuard whether to register blocked URLs in its logfile. The keyword `logall` followed by `on` or `off` tells ufdbGuard whether to register *all* URL verifications in its logfile.

Warning: `logall on` requires many resources and has a performance impact on ufdbGuard. Use this option with care and for short periods only.

8.9 Logfile Rotation

ufdbGuard rotates its logfile automatically whenever it grows beyond 200 MB. When the logfile is rotated, the file `ufdbguardd.log` to `ufdbguardd.log.1` and recreates `ufdbguardd.log`. A maximum of 8 log files are kept. Note that with the default size of 200 MB, the file system where ufdbGuard is installed needs 1.6 GB free space for the log files.

On receipt of the USR1 signal, ufdbguardd also rotates the logfile. The most convenient way to do this is to use `/etc/init.d/ufdb rotatelog`, which takes care of sending the USR1 signal to the right process.

The maximum logfile size is configurable with a parameter in the configuration file. The following line sets the maximum size to 50 MB.

```
max-logfile-size 50000000
```

8.10 Using Quotes

ufdbGuard has many reserved words that cannot be used as labels without using double quotes. In case that you have the need to use a source or category name that is identical to a reserved word, you may use it surrounded by double quotes. E.g.

```
category "aggressive" {

source "unix" {

pass ... !"aggressive" ...
```

Also file names, URL redirection strings, ACLs and parameters dbhome and logdir accept quoted parameters. E.g.

```
logdir "/usr/local/ufdbguard"
time "working-hours" {
```

```
category “aggressive” {
   domainlist     “aggressive/domains”
   expressionlist “aggressive/expressions”
   redirect       “...”

source “unix” {

pass ... !”aggressive” ...
```

8.11 Monitoring

Starting with version 1.24, ufdbGuard has two monitoring options: monitoring by email and monitoring by execution of an external command.

ufdbGuard maintains a status of itself and whenever the status changes, it may send an email message to a configurable email address and/or execute a configurable external command. The external command can be any program or script. Usually it is a script that can send an appropriate command for a monitoring tool. By default, ufdbguardd does not send emails and does not execute external commands.

The status values are *started*, *reloading*, *reloaded*, *error* and *terminated*.

8.11.1 Monitoring by Email

The configuration parameters to enable monitoring by email are:

`mail-server “`*`hostname`*`”`
`admin-email “`*`email-address`*`”`

The mail server must accept SMTP connections on port 25. ufdbguardd uses the email address specified with `admin-email` as both the sender and the recipient address.

The subject and content of the email message are:

`ufdbGuard on `*`hostname`*` has new status: `*`status`*
`database status``: `*`dbstat`*
`license status: `*`licinfo`*

The *status* can have one of the following values: virgin, started, terminated, reloading, reloaded or error. The *dbstat* is a text about the URL database and looks like this:

up to date
one or more tables are more than 4 days old. Check cron job for ufdbUpdate.
one or more tables are EXPIRED. Check licenses and cron job for ufdbUpdate.

The *licinfo* is a self-explanatory text about the license. It may contain a text with the words OK, warning or expired. Warnings are given for licenses that will expire in 2 months or less.

8.11.2 Monitoring by Command Execution

The configuration parameter to enable monitoring by command execution is:

`external-status-command “`*`path-to-executable`*`”`

The command must be specified including the full path, e.g. `/usr/local/bin/ufdbmon`. When executed, the program will receive the parameters `-s `*`status`*` -d `*`dbstat`*` -l `*`licinfo`*.

See the previous section for an explanation about *status*, *dbstat* and *licinfo*.

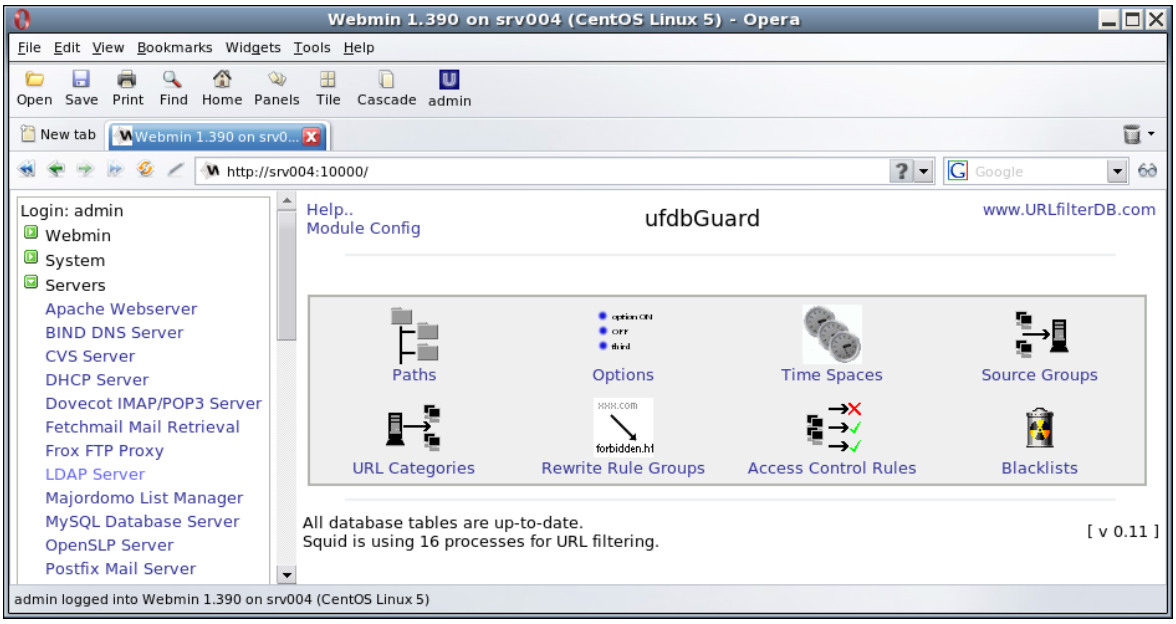
9 Web-based configuration

ufdbGuard can be configured easily by editing the configuration file ufdbGuard.conf. Alternatively, the web-based configuration can also be used. The web-based configuration is a module for webmin (see

www.webmin.com). Expand the *Servers* in the list of webmin modules on the left pane. The ufdbGuard module is at the bottom of the list of the Server modules.

On first usage of the ufdbGuard module for webmin, a check is performed that Squid is using ufdbGuard for URL filtering and offers the option to automatically adjust the configuration of Squid to do so.

After the initial check, the main ufdbGuard configuration window appears (see the next figure). Always configure the main parameters first and Click on *Module Config* in the left upper corner. After the module configuration the other parameters can be defined. The first parameters to define are the paths.



main webmin/ufdbGuard configuration window

The main window offers the configuration of the following items.

Paths define directories for the URL database and the log files

Options define various options

Time Spaces	optionally define time slot rules for internet access
Source Groups	define user groups
URL Categories	a list of predefined URL categories which can be extended with locally defined URL categories (e.g. alwaysallow)
Rewrite Rule Groups	optionally define URL rewrite rules
Access Control Rules	define the ACLs: which user groups are blocked by which URL categories
Blacklists	Edit user-defined URL blacklists (e.g. alwaysallow)
View Log	A simple viewer that shows the last 2500 lines of the log files.

10 Performance Tuning

10.1 Upgrade C libraries

UNIX distributions that have a glibc 2.3.4[4] or older are recommended to be upgraded. This is due to a bug in the regular expression matching subroutines in the standard C library which is fixed in glibc 2.3.5. On the older systems, ufdbguardd will work flawlessly but at the expense of a performance penalty since only one single thread is allowed to perform lookups based on regular expressions at any point in time. Since ufdbGuard cannot distinguish version 2.3.4 from 2.3.5 but can distinguish version 2.3.x from 2.4.x, an upgrade to glibc version 2.4 or higher is recommended.

10.2 Squid performance

The following is recommended to improve the performance of Squid:

- use Linux 2.6 instead of earlier versions since the multithreading code in the libraries and kernel is much more efficient.
- use the `noatime` mount option for the file system with the cache. If reiserfs is used, also use the `notail` mount option. All options reduce the number of I/Os to the disk.
- use squid 3.1.11 or newer since older versions have too many known issues. In case that you compile Squid, configure it to use epoll (`--enable-epoll`) or kqueue (`--enable-kqueue`).
- use a proper number of open files during the configuration phase of squid because at the time that `configure` is run, it is determined what the maximum number of open files can be during the runtime of squid. You may need to use the `ulimit` command and configure the kernel. There is no official guideline for a proper value for the number of open files, but you may find that 100+2.5*NUSERS with a minimum of 1024, is appropriate. Squid versions 2.7 and 3.1 have a new configuration parameter `max_filedescriptors` which can be added to `squid.conf`.
- use a moderately sized disk cache; a very large cache might have a slightly larger cache hit ratio, but the housekeeping of the cache requires more memory and CPU resources. Note that a larger number of cached objects also requires more physical memory for the index.
- use more than one disk for the disk cache, use only one cache directory per disk and do not stripe unless you are using an advanced disk array with internal striping.
- For optimal I/O performance, use the `aufs` cache directory type, but use `diskd` on FreeBSD.
- Squid does a lot of translations of hostnames to IP addresses and the translations are done by first trying to translate to an IPv6 address and if that does not work to translate to an IPv4 address. If the infrastructure does not have IPv6 routers, disabling IPv6 on the host where Squid runs will give a performance gain because the IPv6 lookups and nameserver delays are prevented.
- Use a moderately sized memory cache. Allow enough memory for the kernel's file system cache and other processes. Remember that the configuration parameter `cache_mem` specifies only the amount of memory to be used for objects and the total process size is usually 3 times the amount of `cache_mem`. So, for a machine with 2 GB memory which only runs Squid, a good starting value for `cache_mem` is 400 MB (Squid will occupy about 1.2 GB).
- Experiment with a memory-cache only (disable the disk cache) on large memory systems to get rid of the delays of I/O to disk. The benefit of not having disk I/O is usually bigger than the

[4] consult your system documentation on how to retrieve version information about the glibc library

increased bandwidth usage caused by a reduced cache hit ratio. Note that this only works well with a suitable sized internet connection.

- on systems based on Intel CPUs, disable hyperthreading and use only real CPU cores.
- do not forget to run "`squid -k rotate`" from `cron` as user `squid` because it also does important housekeeping of the cache. Perform the housekeeping on a moment when Squid is not used much, preferably in the middle of the night.
- visit http://wiki.squid-cache.org for more tips.

10.3 Linux 2.6 performance

There are various sources for Linux system tuning and this section is not a replacement. This section contains recommendation for tuning Linux systems that runs Squid and ufdbguardd only.

10.3.1 Optimize System Memory Usage

Linux has various settings that control allocation of memory and how the system behaves when real memory gets scarce. Below are given the setting that gave satisfactory system behavior on our test systems.

Add to `/etc/sysctl.conf` the following lines:

```
# swappiness can have a value of between 0 and 100
# swappiness=0 tells the kernel to avoid swapping processes out of physical
# memory for as long as possible
# swappiness=100 tells the kernel to aggressively swap processes out of
# physical memory and move them to swap cache
# default: 60
# For an application server we use a less aggressive setting of 15.
vm.swappiness=15

# VFS cache pressure:
# default: 100
# With values lower than 100, the data cache is reduced more and the
# inode cache preserved
vm.vfs_cache_pressure=50

# Overcommit or not memory allocations:
# 2      Don't overcommit. The total address space commit
#        the system is not permitted to exceed swap + a
#        percentage (default is 50) of physical RAM.
#        on the percentage you use, in most situations
#        means a process will not be killed while accessing
#        but will receive errors on memory allocation as
#        appropriate.
#        Also ensure that the size of swap is at least 75%
#        of physical RAM with a minimum of 2 GB.
vm.overcommit_memory=2
```

Run the following command to make the new configuration active:

```
# sysctl -p /etc/sysctl.conf
```

10.3.2 Bind ufdbguardd to a Fixed Set of Processors

For large sites that have both Squid and ufdbGuard running on a system with Linux 2.6 and with 2 or more real CPU cores, additional performance can be gained by optimizing the CPU cache efficiency. By separating the squid and ufdbguardd processes over different CPUs with their own memory caches, the individual CPU caches are used in an optimal way.

Squid is a very CPU intensive application and we reserve CPU 0 for Squid. By configuring ufdbGuard to use the other CPUs, the kernel will move squid automagically to CPU 0. The remaining CPUs can be used by ufdbguardd and the process is bound to these remaining CPUs by including the `cpus` keyword in the ufdbGuard.conf file.

system	*for optimal performance, use*
2 simple CPUs	`cpus 1`
2 Xeon CPUs without hyperthreading	`cpus 1`
2 Xeon CPUs with hyperthreading	`cpus 2,3`
4 simple CPUs	`cpus 2,3`
1 dual core	`cpus 1`
2 dual core	`cpus 2,3`
1 quad core	`cpus 2,3`
dual quad core	`cpus 4,5,6,7`

UfdbGuard and Squid use large amounts of memory and therefore the CPU caches are flushed many times. Always try to separate Squid from ufdbGuard and do not let ufdbGuard use more than 50% of the CPU cores. When in doubt, ask the support desk of URLfilterDB for advice.

10.4 Solaris Performance

Use a multi-core system and use a processor set of 2 or more CPUs for Squid and ufdbGuard, and disable interrupts for the CPUs of this processor set. On systems with many cores, use 2 cores for Squid and 2 cores for ufdbGuard.

11 Analysis of User Behavior

To learn more about which types of websites are visited by the users and how much bandwidth they use, the tool ufdbAnalyse is introduced in version 1.15.

ufdbAnalyse reads a Squid log file, strips all non-relevant and personal information, and uploads up to 100 MB of data to the servers of URLfilterDB for analysis. The stripped result file has no details about persons, passwords, session parameters or times, and the processing is done in compliance with our privacy policy (see section 14).

The support desk will send an email with the outcome of the analysis. The output contains for each URL category the #URLs, percentage of URLs, Kbytes and percentage of Kbytes. An example of the output is below.

```
The log files contain 51479 lines of which 2250 contain HTTP error codes.
49229 lines were processed for URLfilterDB categories.

Category                    #URLs     %         KB     %
Adult                        1935   4.0      78204  15.7
Audio & video                 128   0.3      58207  11.7
Advertisements               2449   5.0       2296   0.5
Chat                           18   0.1         56   0.1
Dating & Personals            190   0.4       4425   0.9
Drugs                           0   0.0          0   0.0
Entertainment                6379  13.0      59174  11.9
External Applications           0   0.0          0   0.0
Finance & Investment         1623   3.3      12780   2.6
Forums                        152   0.3        817   0.2
Gambling                      119   0.3        482   0.1
Games                         413   0.9       2565   0.6
Illegal                         0   0.0          0   0.0
Jobs                          357   0.8       3042   0.6
News                         2055   4.2      22224   4.5
Weblogs & private sites       882   1.8       5902   1.2
Peer-to-peer                    0   0.0          0   0.0
Web Proxies                   148   0.3        812   0.2
Security violations             0   0.0          0   0.0
Shops                        3733   7.6      32542   6.5
Sports                        813   1.7       6938   1.4
Toolbars                       23   0.1        290   0.1
Travel                       2586   5.3      21847   4.4
Hacking & warez                 9   0.1         89   0.1
Violence & hate                 0   0.0          0   0.0
Web-based email               105   0.3        809   0.2
Own domain                   1556   3.2       3249   0.7
Allowed sites               23556  47.9     183274  36.7
```

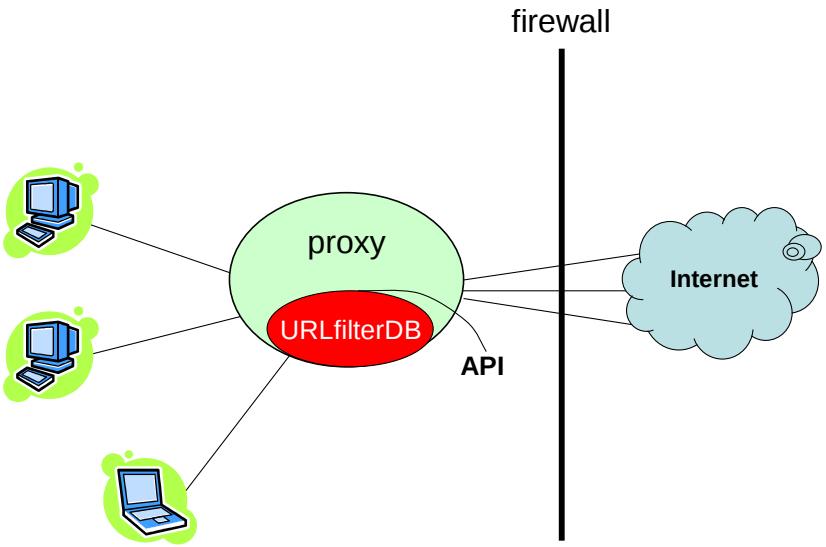
The ufdbAnalyse command uses options to specify the logfile, the own domain, your email address and your full name:

```
$ cd /usr/local/ufdbguard/bin
$ ufdbAnalyse –l access.log –d example.com –e “me@example.com” –n “Kim Li”
```

12 Integration with 3rd Party Products

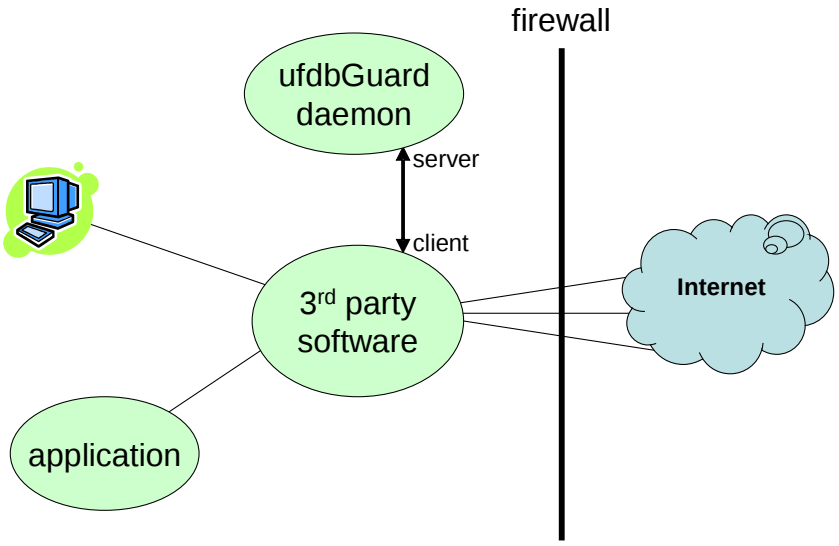
ufdbGuard also has an API that allows it to be easily integrated with 3rd party products. With the use of the API, you have access to the core functions of ufdbGuard and can perform URL verifications from any program written in C with a very good performance: 90,000 URL verifications per second on an Intel Core 2 Duo CPU (introduced by Intel in 2006). The benchmark was performed with 100,000 URLs using a single 64-bit core on an Intel Core2 E6600 clocked at 2.4 GHz using 9 URL categories: ads, adult, gambling, proxies, p2p, sports, news, webmail, games and warez.

The following diagram gives a simplified overview of how a 3rd party product could operate. In this example, a proxy is linked with the URLfilterDB library for extremely fast URL verifications due to the lack of any inter-process communication overhead.



the URLfilterDB library is part of the web proxy

Alternatively, the next diagram shows any 3rd party can copy the simple code of ufdbgclient and incorporate this in 3rd party software and act as a client to the ufdbguardd daemon.



ufdbGuard daemon processes URL verification requests

You may contact the support desk for additional information and licensing for 3rd parties using the URL database.

13 Frequently Asked Questions

ufdbGuard does not block

The most common reasons are either a configuration error (check `ufdbguardd.log`) or that the daemon is not active (verify with `ps –ef | grep ufdbguardd`).

Another common reason is that your trial license has expired.

Verify the log file for warning messages. The most serious errors have a 5-star indicator (`*****`).

ufdbGuard filters HTTP but not HTTPS

ufdbGuard filters everything that Squid asks to filter. The most common mistake is to not use Squid for HTTPS. Perform a test: browse to https://www.google.com and the `access.log` file must have a line with "`CONNECT www.google.com:443`".

A less common mistake is an error with the Squid configuration parameter `url_rewrite_access`.

I want to allow access to site *myfavoritesite.com*

See section 7.1. Do not forget to run `ufdbGenTable` after each change.

Can I define or add new categories ?

Yes. The ufdbGuard software suite includes a utility to convert a plain ASCII file into a URLfilterDB database file.

Can I use ufdbGuard with a free database?

The ufdbGuard software is free and can be used with free URL databases if the database is in .ufdb format or in an ASCII format. The tools `ufdbGenTable` and `ufdbConvertDB` convert ASCII formatted (flat file) databases to the .ufdb format.

Which URLs are in the database?

URLfilterDB does not disclose the content of their database. You are encouraged to test the URL database yourself to find out its effectiveness for your user base. Please refer to section 13.1 for a more detailed description of what you may find in each URL category.

What does it cost?

The URL filter software is free. Prices for a subscription to the URL database are on the website and you can request a quote online. The prices are in EURO and we use a fair exchange rate for countries that prefer to be billed in US dollars. Non-profit and educational institutions receive a 40% discount.

Any other question

You may contact the support staff at support@urlfilterdb.com to answer any other question. Always provide as much details as possible and supply a relevant fragment of the log files.

13.1 URL Categories

The URL database of URLfilterDB uses the following URL categories.

Ads

Advertisements, traffic trackers and web page counters.

P2P

Free sites that can be used directly or indirectly to upload, download and share files. Most P2P sites have copies of movies, adult content, warez and entertainment, and much content violates copyright.

Proxies

Sites that can be used to download content of other sites, URL rewriting sites and VPNs. Proxies are commonly used in an attempt to circumvent a URL filter and should always be blocked.

Adult

Sites suitable for adults only (not only sexual content).

Warez

Illegal software, illegal software codes, hacker's sites.

Toolbars

A toolbar is an extension to a web browser that may violate your privacy or make private files public.

Illegal

Illegal acts

Violence

Violent behavior, aggressive sales of arms.

Gambling

Gambling.

Drugs

Hard drugs, soft drugs, penis enlargement drugs, medicines.

Webmail

Private email accessible with a web browser.

Dating

Love, dating, romantic poetry, and friend sites.

Chat

IRC and chat.

Forum

Sites where people exchange non-business information in a forum.

Private

Blogs and sites of private persons, private web disk, and private file stores.

Audio-Video

Audio and video streams (a TV station is usually in the entertainment category).

Sports

Sports.

Finance

Banks, insurance companies, stock markets, stock brokers

Jobs

Job applications

Games

Games and sites about games

Entertainment

Entertainment, lifestyle, hobby, arts, museums, food, fashion, electronic cards, magazines, horoscopes, desktop wallpapers, clip art, photos, portals, events, fan sites, meditation, baby-related, child sites, file sharing, religion, non-business private interest.

Shops

Shops, price comparisons, and auctions aimed at consumers (b2b is excluded).

Travel

Travel agencies, airliners, tourism sites, hotels, holiday resorts.

News

News and opinions.

External Applications

Free web-based document editors, spreadsheet applications, desktops, groupware, etc. where "internal" documents can be stored on external servers.

Security

This is an administrative category to be able to enforce strict HTTPS usage options (see section 5.4).

Checked

URLs that are verified not to be part of any category and hence always allowed by the URL filter. This "hidden" category is used by the URL filter to track uncategorized URLs. Users should not configure this category.

URLs may be part of 2 or more categories, e.g. `www.usatoday.com` is news while `www.usatoday.com/sport` is both news and sports. If a configuration blocks sports and allows news, it may occur that parts of websites are accessible while other parts are not.

The nature of the content is more important than the strict definition, so an advertisement with a nude person is classified as adult rather than advertisement, and a forum about games is classified as games rather than a forum.

The general impression is also taken into account when a site is categorized. For example, most buyers at `ebay.com` are consumers rather than business users and therefore `ebay.com` is considered a shop for consumers and part of the shops category.

14 Privacy Policy

The privacy policy of URLfilterDB is defined in our Terms of Service document which can be downloaded from the website: www.urlfilterdb.com.

15 More Information

More information can be found on the internet at the following addresses.

URLfilterDB	www.urlfilterdb.com
Squid	www.squid-cache.org