
# Squid on Fedora Core 8 Platform

## Brief History of Squid

In the beginning was the CERN HTTP server. In addition to functioning as an HTTP server, it was also the first caching proxy. The caching module was written by Ari Luotonen in 1994. That same year, the Internet Research Task Force Group on Resource Discovery (IRTF-RD) started the Harvest project. It was "an integrated set of tools to gather, extract, organize, search, cache, and replicate" Internet information. I joined the Harvest project near the end of 1994. While most people used Harvest as a local (or distributed) search engine, the Object Cache component was quite popular as well. The Harvest cache boasted three major improvements over the CERN cache: faster use of the filesystem, a single process design, and caching hierarchies via the Internet Cache Protocol.

Towards the end of 1995, many Harvest team members made the move to the exciting world of Internet-based startup companies. The original authors of the Harvest cache code, Peter Danzig and Anawat Chankhunthod, turned it into a commercial product. Their company was later acquired by Network Appliance. In early 1996, I joined the National Laboratory for Applied Network Research (NLANR) to work on the Information Resource Caching (IRCache) project, funded by the National Science Foundation. Under this project, we took the Harvest cache code, renamed it Squid, and released it under the GNU General Public License.

Since that time Squid has grown in size and features. It now supports a number of cool things such as URL redirection, traffic shaping, sophisticated access controls, numerous authentication modules, advanced disk storage options, HTTP interception, and surrogate mode (a.k.a. HTTP server acceleration).

Funding for the IRCache project ended in July 2000. Today, a number of volunteers continue to develop and support Squid. We occasionally receive financial or other types of support from companies that benefit from Squid.

Looking towards the future, we are rewriting Squid in C++ and, at the same time, fixing a number of design issues in the older code that are limiting to new features. We are adding support for protocols such as Edge Side Includes (ESI) and Internet Content Adaptation Protocol (ICAP). We also plan to make Squid support IPv6. A few developers are constantly making Squid run better on Microsoft Windows platforms. Finally, we will add

more and more HTTP/1.1 features and work towards full compliance with the latest protocol specification

## Squid Is Open Source

Squid is free software and a collaborative project. If you find Squid useful, please consider contributing back to the project in one or more of the following ways:

- Participate on the squid-users discussion list. Answer questions and help out new users.
- Try out new versions and report bugs or other problems.
- Contribute to the online documentation and Frequently Asked Questions (FAQ). If you notice an inconsistency, report it to the maintainers.
- Submit your local modifications back to the developers for inclusion into the code base.
- Provide financial support to one or more developers through small development contracts.
- Tell the developers about features you would like to have.
- Tell your friends and colleagues that Squid is cool.

Squid is released as free software under the GNU General Public License. This means, for example, that anyone who distributes Squid must make the source code available to you. See http://www.gnu.org/licenses/gpl-faq.html for more information about the GPL.

**Minimum Hardware Requirement**

```
Ram   : - 2GB
HDD   : - 80GB
```

**Partition Status**

```
/boot : - 100MB
/     : - 10000MB
SWAP  : - 4000MB (It should be double of RAM)
/usr  : - 5000MB
/var  : - 20000MB (Because sarg report will generate under /var/www/sarg)
/home : - 5000MB
/data : - fill maximum allowable size (to any other data)
```

**Package Selection**

1. Desktop Environment
   -Kde (Select all)

2. Application
   -Editor (VI, vim, and skip others)
   -Graphical Internet (by default selected)
   -Text based Internet (elinks, lynx)

3. Development
   -Development Libraries (Select all)

-Development Tools (Select all)
-Java Development (Select all)
-Kde S/w Development (Select all)

Skip remaining Packages from Development Tab…..

4. Servers
-FTP Server (Select all)
-Legacy N/w Server (Select all)
-MySql Database (Select all)
-Network Server (Select all)
-Remove PostgreSql Database
-Remove Printing Support
-Server Configuration Tools (Select all)
-Web Server (Select all)
-Remove Windows File Server

5. Base System
-Administration Tool (Select all)
-Base (Select all)
-Remove Dialup Networking support
-Remove Fonts
-Hardware Support (By default selected)
-Java (Select all)
-Legacy Fonts (select only xorg)
-System Tools (Select all)
-Remove Virtualization
-Remove X Window System

6. Languages (Skip Everthing)

**Essential Applications for server**

Squid : - squid-2.6STABLE16-2.fc8
Apache: - httd-2.2.6-3
Sarg : - sarg-2.2.3.1-1.el5

These all Essential applications are installed in previous Package Selection, so you need not to install it separately.

After your installation gets finished the very first step should be taken is to change the **ownership** of the **/cache Partition.**

To change the ownership of cache partition type following command.

```
chown -R squid.squid cache
```

Now your Actual Squid Configuration starts where you have to make changes in /etc/squid/squid.conf

```
vi /etc/squid/squid.conf
```

You can see a big file with lot of hashes in the starting of each line. This is actually the configuration file, if u removes the hash or uncomments the parameters then it will perform some action.

You actually have to uncomment only those parameters for which you have to perform some action. I will explain you giving a short example.Suppose u uncomment http_port 3128 and give any desired port say 8080, then the active port would be 8080. This is how it looks……..
http_port 8080

Now we will go to exact configuration implementation of the Squid Proxy Server But before we proceed we should not forget for what purpose Proxy is used. **Proxy** is a server that all computers on the local network have to go through before accessing information on the network. By using Proxy server, an organization can improve the network performance and filter what users connected to the network can access.

First we will define the cache location and the size.
(Squid is a caching proxy for the Web supporting HTTP, HTTPS, FTP, and more. It reduces bandwidth and improves response times by caching and reusing frequently-requested web pages.)

1. Search for
   **cache_dir ufs /var/spool/squid 100 16 256**
   first uncomment the line and change to
   **cache_dir ufs /cache 40000 16 256**
   (40000 :- The default size for cache is 100 (MB) But we have separate partition for cache (40000 MB) so we have to change the value of size to 40000)

This is where you set the directories you will be using. First, you will want to use about 60% or less of each cache directory for the web cache. If you use any more than that you will begin to see a slight degradation in performance. Remember that cache size is not as important as cache speed, since for maximum effectiveness your cache needs only store about a weeks worth of traffic. You'll also need to define the number of directories and subdirectories.
In our scenario, we have given different partition for cache. So that there will be good performance.

2. Search for
   **cache_mem 8 MB**
   Replace with cache_mem 64 MB which will look like
   **cache_mem 64 MB**

You should be aware that cache_mem does not limit the process size of squid. This sets how much memory squid is allowed to set aside for "hot objects" which are the most recently used objects. Having said all that, keep in mind that the buffer cache in Linux is also quite good, so the gains you'll see by raising this are very small, but still measurable.
We have changed the by default size to 64 MB.

3. Search for
   **access_log /var/log/squid/access.log squid**
   Let the above path as it is

Squid saves key information about HTTP transactions in *access.log*

Squid provides a number of logs that can be used when debugging problems, and when measuring the effectiveness and identifying users and the sites they visit. Because Squid can be used to "snoop" on users browsing habits, one should carefully consider privacy laws in your region and more importantly be considerate to your users. That being said, logs can be very valuable tools in insuring that your users get the best service possible from your cache.

**Access log file** defines the location of the cache `access.log`. The Squid `access.log` is the file in which Squid writes a small one line entry for every request served by the cache. This option correlates to the `cache_access_log` directive and usually defaults to `/usr/local/squid/log/access.log` or on some RPM based systems `/var/log/squid/access.log`. In our case, its RPM based means the path is /var/log/squid/access.log

4. Search for
   **http_port 3128**
   Replace with
   **http_port 8080**

You can instruct Squid to listen on multiple ports with additional http_port lines. This is often useful if you must support groups of clients that have been configured differently. For example, the browsers from one department may be sending requests to port 3128, while another department uses port 8080. Simply list both port numbers as follows:

```
http_port 3128
http_port 8080
```

But our configuration defines only one port ie 8080

5. Search for
   **#acl our_networks src 192.168.1.0/24 192.168.2.0/24**
   1st uncomment the line and specify your home network class and Subnet.
   **acl our_networks src 10.0.0.0/8**

The above parameter describes network and subnet mask of the internal network. We have described the network as 10 series, so it will allow only 10.0.0.0 network to go through proxy!

6. Search for
   **#http_access allow our_networks**
   Just uncomment the line with
   **http_access allow our_networks**

It will give http access ie web access to our network ie 10.0.0.0 which we have selected in our previous parameter (**acl our_networks src 10.0.0.0/8)**

# PAM AUTHENTICATION

Pluggable Authentication Modules (PAM) enables the implementation of authentication mechanisms separate from the invocation of those mechanisms in programs.

In many UNIX System programs and utilities, such as **login**, much of the code used to authenticate the identity of the person calling the program is specific to the authentication method, and may require detailed knowledge of other subsystems as well that are related to authentication management. The main goal of PAM is to allow use of the following groups of authentication and authentication-related management functions from an application without detailed knowledge.

Now we will enable pam authentication in our proxy server.

1. Search for
   **#auth_param basic program <uncomment and complete this line>**
   **#auth_param basic children 5**
   **#auth_param basic realm Squid proxy-caching web server**
   **#auth_param basic credentialsttl 2 hours**
   **#auth_param basic casesensitive off**

   1st uncomment the above lines and edit the following changes.
   **auth_param basic program /usr/lib/squid/pam_auth**
   **auth_param basic children 5**
   **auth_param basic realm Squid proxy-caching web server**
   **auth_param basic credentialsttl 2 hours**
   **auth_param basic casesensitive off**

This is the default settings done for ITM where you can see only the first parameter have changed and the path which have been set is the /usr/lib/squid/pam_auth. Pam_auth file contains binaries.

**ACL TYPE**

## Synopsis

Provides match against external ACL lookup via a helper class defined by the external_acl_type tag

## Example(s)

auth_param basic program < put your authenticator here >
auth_param basic children 20
auth_param basic realm Squid proxy-caching web server
auth_param basic credentialsttl 1800 seconds
external_acl_type checkip children=20 %LOGIN %SRC /usr/local/Squid/bin/checkip.pl
acl password external checkip
acl it src 172.16.20.1-172.16.20.199/255.255.255.255
http_access allow it password
Allows user if user belongs to a group that is allowed during a given time and using a given ip.

And for more details about parameters, visit,
http://www.visolve.com/squid/squid30/accesscontrols.php#dstdomain

2. Search for

```
#acl password proxy_auth REQUIRED
And replace with
acl password proxy_auth REQUIRED
acl pam proxy_auth REQUIRED
```

## proxy_auth

Description: This ACL type calls an external authenticator process to decide whether the request will be allowed
Some of the authenticator helper programs available for Squid are PAM, NCSA, UNIX passwd, SMB, NTLM, etc. Note that authentication cannot work on a transparent proxy or HTTP accelerator. The HTTP protocol does not provide for two authentication stages (one local and one on remote Web sites). So in order to use an authenticator, your proxy must operate as a traditional proxy, where a client will respond appropriately to a proxy authentication request as well as external Web server authentication requests.

Note: *proxy_auth* can't be used in a transparent proxy. It collides with any authentication done by origin servers. It may seem like it works at first, but it doesn't. When a Proxy-Authentication header is sent but it is not needed during ACL checking the username is NOT logged in access.log.

### Arguments

| | |
|---|---|
| *aclname* | Access list name |
| *username* | User name to be authenticated |

### Example(s)

acl ACLAUTH proxy_auth ramesh senthil muthu
http_access allow ACLAUTH
http_access deny all
The above configuration will allow only ramesh, senthil and muthu if they give valid username and password.

```
3. Search for
   http_access deny !Safe_ports
   and change to
   http_access allow pam
   Now you can save and close the /etc/squid/squid.conf file
```

## How do I use authentication in access controls?

Make sure that your authentication program is installed and working correctly. You can test it by hand.

Add some *proxy_auth* ACL entries to your squid configuration. For example:

```
acl foo proxy_auth REQUIRED
acl all src 0/0
http_access allow foo
http_access deny all
```

The REQURIED term means that any authenticated user will match the ACL named *foo*.

Squid allows you to provide fine-grained controls by specifying individual user names. For example:

```
acl foo proxy_auth REQUIRED
acl bar proxy_auth lisa sarah frank joe
acl daytime time 08:00-17:00
acl all src 0/0
http_access allow bar
http_access allow foo daytime
http_access deny all
```

In this example, users named lisa, sarah, joe, and frank are allowed to use the proxy at all times. Other users are allowed only during daytime hours.

4. Now we will enable pam modules for squid proxy authentication
   **vi /etc/pam.d/squid**
   and add the following lines
   **auth required /lib/security/$ISA/pam_env.so**
   **auth sufficient /lib/security/$ISA/pam_unix.so likeauth nullok**
   **auth sufficient /lib/security/$ISA/pam_ldap.so use_first_pass**
   **auth required /lib/security/$ISA/pam_deny.so**

# ACL

(site, word and download blocking)

Squid's built-in blocking mechanism or access control is the easiest method to use for implementing web site blocking policy. To deploy the web-site blocking mechanism in Squid, add the following entries to your Squid configuration file

```
vi /etc/squid/squid.conf
```

Site level blocking

1. search for
   **#acl aclname url_regex [-i] ^http:// ...**
   and change to
   **acl blocksites url_regex "/etc/squid/blocksites.acl"**
   **blocksites** is an acl name and the given file contains all block sites list and add following line
   **http_access deny blocksites**

The file `/etc/squid/blocksites.acl` contains web sites or words you want to block. You can name the file whatever you like. If a site has the URL or word listed in `blocksites.acl` file, it won't be accessible to your users. The entries below are found in `blocksites.acl` file used by my clients:

```
.oracle.com
.playboy.com.br
sex
...
```

Similarly as shown below we have defined blockwords.acl and blockdownload.acl files where it restricts the client to access words and extension defined in the file.

Word level blocking

2. search for
   **#acl aclname dstdom_regex [-i] xxx ...**
   and change ft
   **acl blockwords dstdom_regex -i "/etc/squid/blockwords.acl"**
   **blockwords** is an acl name and the given file contains all blockwords list (eg. Sex porn naughty) and add following line
   **http_access deny blockwords**

Download blocking

3. search for
   **#acl aclname urlpath_regex [-i] \.gif$ ...**
   Change to
   **acl blockdownload urlpath_regex -i "/etc/squid/blockdownload.acl"**
   **blockdownload** is an acl name and the give file contain all blocked extension (e.g. .mp3, .mpeg and .mov) add following line

```
      http_access deny blockdownload
```

Now you can save and close the **/etc/squid/squid.conf** file

```
Chkconfig --level 35 squid on
which keep your squid service on runlevel 3(text mod) and 5(GUI)
service squid start
```

# SARG

(Squid Analysis Report Generator)

Sarg - Squid Analysis Report Generator is a tool that allows you to view "where" your users are going to on the Internet. Sarg provides much information about Squid users activities: times, bytes, sites, etc...

Download sarg-2.2.3.1-1.el5.rf.i386.rpm through following link http://rpmfind.net/linux/RPM/dag/redhat/el5/i386/sarg-2.2.3.1-1.el5.rf.i386.html Put it in to **/opt** and run the following command.

```
rpm -ivh sarg-2.2.3.1-1.el5.rf.i386.rpm
```

Now edit the sarg configuration file

```
vi /etc/sarg/sarg.conf
```

```
search for
access_log /var/log/squid/access.log

And do not change this path because sarg fetches the log information of
all proxy users from this file.(access.log)
```

SARG will automatically create folder called sarg under /var/www/ which will be the Document root for httpd server. now we will configure apache server to view SARG reports.

```
vi /etc/httpd/conf/httpd.conf
```

```
search for
DocumentRoot "/var/www/html"
and change to
DocumentRoot "/var/www/sarg"

search for
<Directory "/var/www/html">
change to
<Directory "/var/www/sarg">
```

Now you close the apache configuration file

```
chkconfig --level 35 httpd on
```

```
which keep your squid service on runlevel 3(text mod) and 5(GUI)
service httpd restart
```

Start mozill firefox and put your proxy IP in the browser http://127.0.0.1 or http://your proxy ip or http://localhost normally sarg take 24 hours to generate report. Also you can manually generate the reports

E.g. if you want the report of all users for 9th Feb. 2009 then type following command

```
sarg -d 09/02/2009-09/02/2009
```

You can also refer to help for more details.

```
sarg --help
```

**By IT Team**
**ITM KHRARGHAR**