Security considerations with Squid proxy server

GIAC Security Essentials Certification Practical Assignment
Version 1.4b, Option 1

By: Eric Galarneau
April 2, 2003

Abstract

Securing and controlling workstation access to the web has never been an easy task for security professionals. Firewalls and access control list on routers alone may not bring an acceptable level of security for your organization. Even if their primary role is to reduce network traffic and improve performance, HTTP proxy servers (also called cache servers) are likely to be installed as an additional security layer and as a web surfing monitoring system. Having secure proxy servers is critical because many users depend on it for their work. Several proxy server products are available nowadays. Since commercial products are very expensive, alternative products such as open source software are often considered. The most popular and widely used proxy server in this category is indisputably Squid.

This paper will cover various security aspects and recommendations to improve Squid's overall security during its installation time. Software configuration parameters and an overview of logging and content filtering software will also be approached.

Installation

It is critical that Squid be installed on a dedicated server located in a secure restricted room. The main reason for installing Squid on a dedicated server is to prevent people using other services to access the outside world without going through the proxy service. If your proxy server is easily accessible to anyone, it then becomes vulnerable and a very attractive target. This vulnerability allows someone with malicious intent to damage, violate, steal and falsify the information flowing through your proxy server. Thus, other security measures in place would be totally wasted. The importance of a secure restricted room in this case is even more critical because firewalls are likely to trust proxy servers for some ports and protocols.

Squid is designed to run on a UNIX type system. When installing most UNIX operating system, many unwanted services are likely to be installed by default. Get rid of any services that would not be used since vulnerabilities in other services may give potential access to your system. Attacks are often performed through unused running services. For example, many UNIX like operating systems are installed with sendmail enabled by default while older versions of sendmail are known to contain weaknesses and vulnerabilities. This practice should already be covered in your server installation policy. If your organization does not have such policy, CERT provides very useful guidelines on how to secure your UNIX system. These guidelines are a good starting points for the creation of a server installation policy:
http://www.cert.org/tech_tips/unix_configuration_guidelines.html

Once you are ready to proceed with the installation, create a sandbox user specifically dedicated to Squid. Assign an invalid shell to the sandbox user or consider a free program called noshell. In addition to being an invalid shell, noshell also sends a warning message to the syslog facility whenever a login is attempted with this sandbox user. Noshell can be downloaded at the following location:
http://www.fish.com/titan/src1
The purpose of creating a sandbox user is to limit the privileges from which a buffer overflow can be exploited. Version 2.4 is known to contain a few buffer overflows. For example it is possible to execute code with the Squid user privileges by sending malformed URL's. For additional details and patches on this vulnerability, refer to the Squid bug list available at the following location:
http://www.squid-cache.org/Versions/v2/2.4/bugs

Squid requires directories to store cache data locally. Ensure that only the Squid user (sandbox) has read/write/execute access to these directories. It is suggested to set the sticky bit in the directories' permissions. Having the sticky bit set prevent cache data from being deleted or altered by another user or other processes on the system.

A switched network offers protection against network sniffing and reduces the danger of having sensitive information stolen such as usernames and passwords used for Squid. Even if switched network is highly recommended, the possibility for an experienced hacker to use more advanced techniques to perform sniffing over a switched network exists. More relevant information on this topic can be found at the following location:
http://www.sans.org/resources/idfaq/switched_network.php

Strong security can only be achieved with defense in depth. Applying multiple layers of defense limits your system accessibility. The organization's web access security should not solely rely on Squid. In addition to Squid, the installation of firewalls and the use of router access lists are essential in order to restrict connections to your proxy server. As mentioned earlier, proxy servers are often targeted by hackers. Thus, access to Squid should be limited to the strict minimum.

Squid does not provide PGP signature at the present time. However, MD5 checksum is available at:
ftp://ftp.squid-cache.org/pub/squid-2/md5s.txt
The integrity of your source file should always be verified. This only takes a few minutes and can save you from a disaster. Some people learned that the very hard way after running sendmail from sources that contained a Trojan horse. Refer to the following link for SANS newsletter on Trojan horse found in sendmail:
http://www.sans.org/newsletters/newsbites/vol4_42.php

Configuration

Squid involves a large number of parameters and options. The default configuration file already includes minimal documentation to get started. This configuration file is stored on the system at the following location:
{INSTALLPATH}/etc/squid.conf.default
This paper will only cover the most important parameters that can affect the security of the proxy server. For additional information on Squid parameters, visit the Visolve web site at:
http://squid.visolve.com/squid24s1/contents.htm

http_port

The option *http_port* defines which port number squid listens to for incoming requests. The default port is 3128 but it is recommended to change this port number to any other available port in the high range. Since port scanners are likely to assume 3128 and 8080 for proxy service, running Squid on any other free port in the high range will make its detection more difficult. If your server has more than one interface, you can restrict the IP address on which it listens to. For example, *http_port 10.50.5.1:7548* will make Squid listen on port 7548 on 10.50.5.1 interface. Therefore, any request coming from other interfaces will be ignored. If more than one interface is used, Squid supports multiple http_port lines.

maximum_object_size value
minimum_object_size value

Data stored locally by a proxy server is often referred to as an object. The parameters *max_object_size* and *minimum_object_size* allows forcing limits on the size of a cached object. The primary goal of this option is to optimize the cache/HIT ratio. This option is also interesting in a security standpoint since it helps to protect against a denial of service. Setting the *maximum_object_size* too high would make it possible for an attacker to overload your proxy server by requesting numerous large objects and therefore slow down other proxy users. This type of denial of service attack can be performed even more easily if Squid is configured to continue downloading objects after the request has been cancelled. Unfortunately, this is Squid's default configuration but it can be changed using *quick_abort_min*, *quick_abort_max*, *quick_abort_pct* options.

ftp_user Squid@
ftp_passwive on
ftp_sanitycheck on

It is important to understand that Squid is not an FTP proxy but it supports FTP through HTTP. The default password sent by Squid to anonymous FTP sites is assigned by the *ftp_user* option. In some cases, FTP sites look for valid email addresses used as a password. Create a valid email alias that will serve exclusively to this purpose and make sure emails sent to this address can be read regularly. This email address will be used to report any abuse coming from your proxy server.

The option *ftp_passive* is turned on by default and should be left as is unless your firewall does not support passive mode. The passive mode is considered to be more secure because it uses 2 fixed ports; one for connection and one for data transfer while the active mode uses ephemeral ports. Firewalls generally need fixups in order to handle an active FTP session. Recent versions of squid now offer the *ftp_sanity_check* option. This option uses an extensive mechanism to ensure the connection is established with the requested server and it must be left on.

dns_nameservers value

Squid uses the name servers listed in */etc/resolv.conf* file. Different name servers can be listed by using *dns_nameservers* directive. Name servers used by squid must come from trustable sources and configured safely. A compromised DNS server is often used by attackers to divert proxy servers and certain Squid versions can be crashed by sending malformed DNS answers. Additional details on this Squid vulnerability can be found at the following location: http://www.xatrix.org/print1312.html

authenticate_ttl value
authenticate_ip_ttl value

When authentication is used, the parameter *authenticate_ttl* defines how long Squid is to remember client authentication information. This option forces the client to re-authenticate himself after the specified period (TTL) has expired. Recent versions of internet explorer allow you to remember username and password making this option useless. Microsoft policy tools can be used to prevent people from selecting this feature in internet explorer. The above should be covered in your organization's internet usage policy.

The parameter *authenticate_ip_ttl* defines how long a client's authentication should be bound to a particular IP address. The purpose of this parameter is to discourage people from sharing their passwords among themselves. The downside of using this parameter is it can inadvertently block legitimate access to some clients where IP addresses are subject to change regularly such as dialup users. This parameter should therefore be used with precaution.

request_header_max_size value

The *request_header_max_size* option is used to limit the size of acceptable HTTP headers. The default value of 10 kilobytes is more than reasonable since the average header size is approximately 512 bytes while very large headers may hit kilobytes. Most denial of service attacks against proxy servers are attempted by sending them headers larger than what they can handle. This option should therefore be left to its default value and never be disabled.

client_lifetime value
pconn_timeout value

The *client_lifetime* sets the maximum time a client is allowed to be bound to a Squid process. This protects your server from having too many sockets opened. Depending on your organization setup, a lifetime of 6 hours would probably be appropriate while a lifetime of 24 hours would be considered quite long. It is common to see clients connected for hours during the course of a working day but most of the time, these are simply hang-up processes. An attacker can take advantage of a lifetime that is too long by opening a large number of sockets. Therefore, a longer client lifetime period makes Squid more vulnerable to this type of denial of service attack. The *pconn_timeout* parameter is similar to the *client_lifetime* parameter with the exception that it only applies to idle connections.

Acl aclname element
http_access permit|*deny aclname*

From a security perspective, one of the most important section of the configuration file is the access control list (ACL's). This controls whatever a particular HTTP request is permitted or denied. Two parameters are required to achieve this control; *acl* defines the element to be controlled and *http_access* permits or denies the element.

According to the Squid FAQ, the possible elements are:

- **src**: source (client) IP addresses
- **dst**: destination (server) IP addresses
- **myip**: the local IP address of a client's connection
- **srcdomain**: source (client) domain name
- **dstdomain**: destination (server) domain name
- **srcdom_regex**: source (client) regular expression pattern matching
- **dstdom_regex**: destination (server) regular expression pattern matching
- **time**: time of day, and day of week
- **url_regex**: URL regular expression pattern matching
- **urlpath_regex**: URL-path regular expression pattern matching, leaves out the protocol and hostname
- **port**: destination (server) port number
- **myport**: local port number that client connected to
- **proto**: transfer protocol (http, ftp, etc)
- **method**: HTTP request method (get, post, etc)

- **browser**: regular expression pattern matching on the request's user-agent header
- **ident**: string matching on the user's name
- **ident_regex**: regular expression pattern matching on the user's name
- **src_as**: source (client) Autonomous System number
- **dst_as**: destination (server) Autonomous System number
- **proxy_auth**: user authentication via external processes
- **proxy_auth_regex**: user authentication via external processes
- **snmp_community**: SNMP community string matching
- **maxconn**: a limit on the maximum number of connections from a single client IP address
- **req_mime_type**: regular expression pattern matching on the request content-type header
- **arp**: Ethernet (MAC) address matching

Squid ACL's are first match meaning that the first *http_access* statement permitting/denying the request wins over any other declaration thereafter. A good practice would consist of including an *http_access deny all* statement at the end of your ACL's list. This will make whatever is not explicitly permitted to be denied by default.

The following is an example of limiting proxy access from source addresses 10.50.0/255.255.0.0:

Acl all src 0.0.0.0/0.0.0.0 # Every IP addresses.
Acl mainplan src 10.50.0.0/255.255.0.0 # IP addresses from 10.50.0.0 network.

http_access allow mainplan # Allow IP addresses from 10.50.0.0 network.
http_access deny all # Deny every IP addresses.

The *trusted_ports* is one of the most important acl element and is used to restrict connections to certain port numbers. The best approach for this declaration is to strictly add the port numbers as needed. Port numbers should never be added simply because you think you might need them some day. Note that HTTP and SMTP headers are very similar in design and because of this, port 25 should never be included to your list of trusted ports. Including port 25 would make it possible for someone to relay email messages through Squid. The example below demonstrates how multiple port numbers can be specified and how they can be placed on the same declaration line:

Acl all src 0.0.0.0/0.0.0.0 # Every IP addresses.
Acl mainplan src 10.50.0.0/255.255.0.0 # IP addresses from 10.50.0.0 network.
Acl trusted_ports port 21 80 443 # List of trusted ports

http_access deny !trusted_ports # Deny if not a trusted port
http_access allow mainplan # Allow IP addresses from 10.50.0.0 network.

http_access deny all # Deny every IP addresses.

Squid supports authentication and its usage is strongly encouraged. In addition to providing more security, this feature will make the monitoring task much easier by showing the username along with the GET request log entry. Various authentication modules such as NCSA, LDAP and NTLM are supported by Squid and referred to as helpers.

Squid supports two authentication methods. These methods are known as Basic and Digest and are differentiated by encryption. Basic sends password in cleartext while Digest sends it encrypted. If authentication is only required for accounting purposes, Basic authentication method is an acceptable choice. In other cases, the Digest authentication method is undoubtedly the method to be used. For complete details on these methods, read the RFC 2617 available at: ftp://ftp.isi.edu/in-notes/rfc2617.txt

The *acl proxy_auth REQUIRED* declaration is used to define that authentication will be used. This declaration combined with *http_access* will prompt the client browser to specify a username and password before allowing the request. After having entered the username and password, the browser uses credentials to remember this information for future requests.

The following required parameters should be inserted before going any further with authentication:

# Required parameters
auth_param basic program {PATH}/libexec/ncsa_auth {PATH}/etc/passwd
auth_param basic children 5
auth_param basic credentialsttl 2 hours
auth_param basic realm External network access

The main purpose of the *auth_param basic program* parameter is to specify which helper is to be used by Squid. The *auth_param basic children* states how many children helper process to start and *auth_param basic credentialsttl* protects against replay attacks by specifying how long usernames and passwords are valid for. Finally, *auth_param basic realm* is the message displayed in the browser authentication dialog box. Below is an example of enabled authentication with Basic NCSA helper:

Acl all src 0.0.0.0/0.0.0.0 # Every IP addresses.
Acl mainplan src 10.50.0.0/255.255.0.0 # IP addresses from 10.50.0.0 network.
Acl trusted_ports port 21 80 443 # Port numbers to be trusted.
Acl passwd proxy_auth REQUIRED # True if authentication is successful.

http_access deny !trusted_ports # Deny if not on a trusted port number.

http_access allow mainplan passwd # Allow authenticated users from 10.50.0.0 network.
http_access deny all # Deny every IP addresses.

reply_body_max_size value

The *reply_body_max_size* option imposes a maximum size of a reply body. This option is very useful to prevent users from downloading large files such as mpeg video files. Use this option with care because if the value is smaller than Squid error pages, it will cause an infinite loop and possibly crash your proxy server. Squid does not have a default maximum body size. A reasonable value of approximately 2 megabytes should therefore be considered.

cache_effective_user nobody
cache_effective_group nobody

The *cache_effective_user* and *cache_effective_group* options specify under which user and group Squid will be running. The defaults are set to user and group nobody. These values are to be changed to the dedicated (sandbox) user as recommended earlier.

forwarded_for value

A web server does not see connection with the client but rather sees the connection with the proxy. To accommodate possible web server ACL's based on client address, Squid has its own HTTP header called HTTP_FORWARD_FOR. Turning off this header might cause clients some problems accessing certain web sites. However, turning it off gives you the advantage of hiding the client's IP address on the internet which is a very good security practice.

delay_pools
delay_access

The *delay_pool* option allows you to limit transfer rates. To help manage network bandwidth utilization, Squid must be compiled to support delay pools. Acl declarations must be defined in your configuration to enable delay pools. In a practical point of view, this option gives you the possibility to limit the network usage from a particular network leaving more network availability to other services. In a security point of view, a global delay pool makes it difficult for an attacker to harm other network services by trying to flood your proxy server.

coredump_dir

The *coredump_dir* is used to indicate where Squid is to save its coredump files. The default is to save coredump into Squid's installation directory. If someone should find the coredump directory in its default location (/ or /usr), that person would be able to fill the partition in which the coredump files are saved and cause a denial of service. Since these coredump files can get very huge, make sure you change this option to a directory outside a partition critical for system operation.

ignore_unknown_nameservers

This option verifies if the nameserver answering the lookup has the same IP address than the one the lookup was sent to. This makes it much more difficult for an attacker to falsify DNS queries. This option is on by default and should be left as such.

Cachemgr.cgi

Squid comes with a cgi script called *cachemgr.cgi*. This script allows an administrator to obtain useful information such as DNS statistics, Cache hits, active connections etc…
This script could be useful to an attacker due to the information it reveals. If this type of information is not required by your organization, disregard installing *cachemgr.cgi* script. On the other hand, if such information is required, a Squid Acl can be used to limit its access as follows:

```
acl cachemgr proto cache_object
acl proxyserv src 10.50.1.1/255.255.255.255
http_access deny cachemgr !proxyserv
```

As a secondary security option, consider another ACL on your web server to access the script. Be aware that *cachemgr.cgi* sends over password using an HTML form. This method is dangerous since username and password are directly displayed on the browser address space. This weakness will definitely need to be improved in upcoming versions of Squid. Until then, stay alert for eavesdropping!

Logging

To improve performance, Squid offers you the possibility to turn off logging. Knowing that logging is the only way you can tell what is happening with your server, turning it off is a very bad idea. If performance is a concern for you, consider taking advantage of Squid's peering capabilities. The performance benefit of turning off logging is simply not worth it!

Squid uses *access.log* to log requests. Dealing with huge log files may become a problem. This file can be rotated by sending a *{INSTALLPATH}/sbin/squid –k rotate*. In doing so, your current *access.log* file will be moved to *access.log.1* and Squid starts over using a new *access.log*. To manage huge log files effectively, write a script that will run this command periodically, zip the file and append a timestamp to it. Archived log files should be saved periodically on safe long term storage for future reference.

Squid has its own logging format but also supports the HTTP standard log format. One interesting advantage of the Squid log format is its capability of reporting session duration. This is useful to detect audio/video streaming passing through your proxy server.

HTTP Log format example:

10.50.5.2 - myname [21/Mar/2003:00:02:08 -0500] "GET http://www.somewhere.com/object? HTTP/1.0" 200 3906 TCP_MISS:DIRECT

Squid Log format example:

1048204026.331   488 10.50.5.7 TCP_MISS/200 505 GET http://www.somewhere.com/object? myname DIRECT/207.68.177.124 image/gif

Log Analyzers

Log analyzers are used to generate comprehensible reports from the *access.log* file. A large number of these analyzers are listed on Squid's web site: http://www.squid-cache.org/Scripts
In my opinion, SARG is the most interesting free log analyzer currently available. SARG takes your given *access.log* file and generates a well presented HTML report. This report gives you an overview of where your uses have been, at what time and how long they have spent browsing the internet. In addition, this report will help to detect abusive or abnormal usage of your proxy server. SARG is available at:
http://web.onda.com.br/orso

Virus and content filters

Any virus administrator will tell you that keeping anti-virus software up to date on every workstation is almost not feasible. As an additional anti-virus protection, you may want to install HTTP virus filters. TrendMicro is probably the most popular product which can work with Squid. In addition to virus detection capabilities, the TrendMicro product offers content filtering features. However, it only supports a few operating systems. If an HTTP filter is considered, make sure you plan additional resources to deal with the extra load caused by such filter.

Content filtering can be performed on the URL with a Squid acl but you must be aware that large Squid acl's reduce the proxy's performance. If very large lists are expected, other softwares can achieve this more efficiently such as SquidGuard which uses Berkeley databases to improve performance. SquidGuard also offers additional interesting features. For example, SquidGuard

can be configured to filter certain keywords from a determined network during a specified timeframe. SquidGuard is available at:
http://www.squidguard.org

Content filters are not 100% effective. One popular technique to avoid content filters consists of inserting HTML tags between words to prevent them from being matched. The following is a good example:

Will match.
<B>Iamabadword</B>

Will not match:
<B>Iam</B><B>abadword</B>

Both of the above examples will be displayed as Iamabadword (bold) in an HTML document.

Conclusion

Security precautions must be taken at time of installation and configuration in order to bring Squid's security to an acceptable risk level for your organization. For optimal security, it is imperative to log and monitor Squid. Many tools have been suggested and analyzed in this paper to help you achieve these tasks successfully.

As a security professional, keeping up to date with Squid's updates and new discovered vulnerabilities is mandatory. Two good ways to be kept up to date is to consult Squid's security advisories at:
http://www.squid-cache.org/Advisories
and Squid's mailing lists available at:
http://www.squid-cache.org/mailing-lists.html

Remember that one is never too informed. People with bad intentions are always coming up with new techniques and ideas to breach security. It is your duty to be one step ahead and the internet is a powerful resource to use for you to maintain this advance.

References

1. Cert Coordination Center "UNIX Configuration Guidelines" Feb. 12, 1999.
URL http://www.cert.org/tech_tips/unix_configuration_guidelines.html (Mar 6, 2003)

2. Brad Powell, Matt Archibald, Dan Farmer. "Noshell Ver 4.0.11" Apr. 22, 2001.
URL http://www.fish.com/titan/src1 (Mar 26, 2003)

3. Squid 2.4 patches. "Buffer overrun on certain malformed URLs" Sep. 12, 2002.
URL http://www.squid-cache.org/Versions/v2/2.4/bugs (Mar 26, 2003)

4. Steven Sipes. "Why your switched network isn't secure" Sep. 10, 2002.
URL http://www.sans.org/resources/idfaq/switched_network.php (Mar 26, 2003)

5. Unknown. "md5s.txt" Mar. 19, 2003.
URL ftp://ftp.squid-cache.org/pub/squid-2/md5s.txt (Mar 26, 2003)

6. Cert "Advisory CA-2002-28 Trojan Horse Sendmail Distribution" Oct. 8, 2002.
URL http://www.sans.org/newsletters/newsbites/vol4_42.php (Mar 26, 2003)

7. Dalibor Glavan "Squid Compressed DNS Buffer Overflow Vulnerability" Mar. 28, 2002.
URL http://www.xatrix.org/print1312.html (Mar 26, 2003)

8. Network Working group "Request for Comments: 2617 - HTTP Authentication: Basic and Digest Access Authentication" June, 1999.
URL ftp://ftp.isi.edu/in-notes/rfc2617.txt (Mar 26, 2003)

9. Squid Scripts. "Squid Cache Logfile Analysis Scripts" May, 1998.
URL http://www.squid-cache.org/Scripts (Mar 26, 2003)

10. Pedro L Orso. "Squid and Web utilities – SARG Ver 1.4" Mar. 16, 2003.
URL http://web.onda.com.br/orso (Mar 26, 2003)

11. Pal Baltzersen, Lars Erik Haland. "An ultrafast and free filter, redirector and access controller for Squid – SquidGuard Ver. 1.2.0" Dec 18, 2001.
URL http://www.squidguard.org (Mar 26, 2003)

12. Squid Advisories. "SQUID-2002:1, SQUID-2002:2, SQUID-2002:3" July 3, 2002.
URL http://www.squid-cache.org/Advisories (Mar 26, 2003)

13. Squid Mailing lists. "Squid Mailing lists" Jan. 22, 2003
URL http://www.squid-cache.org/mailing-lists.html (Mar 26, 2003)