

Troubleshooting the SQUID proxy server

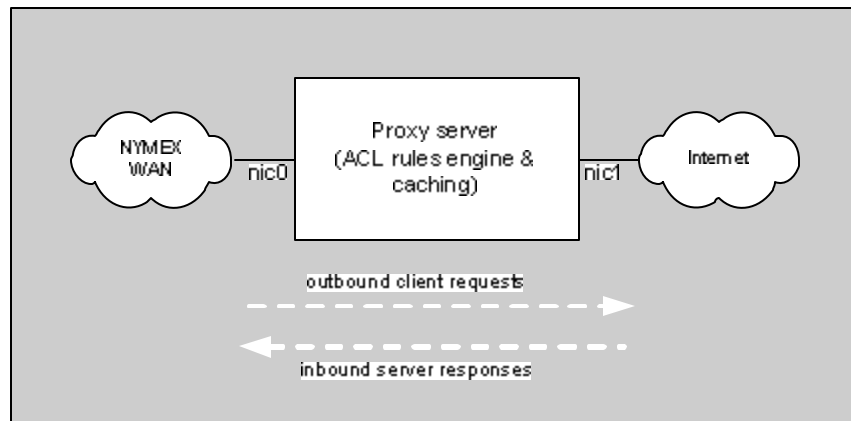
## Troubleshooting the SQUID proxy server

## **Introduction**

As the title describes, this document covers how to troubleshoot the proxy server/service. For the most part, the proxy server will seamlessly perform the duty of facilitating (proxying) internet access to users and/or applications. Because it will run largely unattended (i.e. forgotten), and because its significance will grow over time (as more users and application come to depend on it), it will be important to have a step-by-step procedure for troubleshooting the proxy server when we suspect that there is a problem with it. The intent of this document is to provide such a step by step procedure.

## Implementation

The architecture of the proxy service itself is simple enough: it consists of a multi NIC server running the Solaris or Linux operating system, and the public domain Squid Cache proxy server (<http://www.squid-cache.org>). As shown here, client applications make a request to a particular URL (http, ftp, https, etc), specifying the IP address and Port of the Proxy server that is to service that request. The request is either granted or denied access based on the proxy server working rule set (i.e. the in-memory ACL rule set). Any client that has the ability to specify a proxy server IP address & Port to service its URL requests, can make use of the proxy server to access the internet, provided that the request being made is authorized by the rules engine.



## Troubleshooting the SQUID proxy server

The diagram above underscores the need for multiple active proxy service instances to ensure high availability. As such multiple active proxy services will be deployed as follows:

1 North End (even number instances)	iPark (odd numbered instances)
proxy02.prod.nymex.com:3128	proxy01.prod.nymex.com:3218
proxy04.prod.nymex.com:3128	proxy03.prod.nymex.com:3218

The names above reflect the fully qualified DNS names for these servers, however client proxy configurations must specify the CNAMEs assigned to each of these. They are as follows:

Fully Qualified DNS Name	CNAME (initial association)
proxy01.prod.nymex.com	prxyA1.prod.nymex.com
proxy03.prod.nymex.com	prxyA2.prod.nymex.com
proxy02.prod.nymex.com	prxyB1.prod.nymex.com
proxy04.prod.nymex.com	prxyB2.prod.nymex.com

In addition to using CNAME's, applications (i.e. clients of the proxy) should be written so as to take advantage of multiple active proxy servers. If a proxy server is not responding to its requests, the application should seek to use the next one in its list.

## Troubleshooting the SQUID proxy server

### Troubleshooting

Netcool will be configured to query the various proxy servers & services. In addition to pinging the servers themselves, as well as checking that the proxy application daemon (squid) is running, Netcool will also be configured to generate actual URL requests and check their responses. You can be sure that when a proxy service goes down, Netcool will not be the only one sounding an alarm. When this occurs, below is what to do to troubleshoot and isolate the problem. Note: if at anytime during the troubleshooting process you determine that it is necessary to restart the squid proxy server, here is the proper way to do it (but do not automatically do this without cause):

```
root@proxy# /etc/init.d/squid* stop
Stopping squid that uses '/usr/local/squid/etc/squid.conf'... O.K.
root@proxy# ps -ef | grep squid | grep -v grep
root@proxy#

root@proxy# /etc/init.d/squid* start
Starting squid using '/usr/local/squid/etc/squid.conf'... O.K.
root@proxy#
```

Starting on the suspect proxy server itself and working outward

(1) Is the squid (proxy daemon running), and does the kernel TCP/IP stack:

```
user@proxy$ ps -ef | grep squid | grep -v grep
... /usr/local/squid/sbin/squid -f /usr/local/squid/etc/squid.conf
... (squid) -f /usr/local/squid/etc/squid.conf

user@proxy$ netstat -an | egrep -i "\.3128|squid" | grep LISTEN
172.17.37.25.3128    *.*    0  0  24576      0 LISTEN
-or-
172.17.37.25.squid *.*    0  0  24576      0 LISTEN
```

Note: The IP address will vary depending on the proxy server you are troubleshooting. Also, if "/etc/services" defines an association between the service name "squid" and its (default) port of 3128, the port number will be resolved into its service name (as shown in the second case above).

## Troubleshooting the SQUID proxy server

- (2) Next, on the proxy server itself, use telnet to see if you can access well known SSL and non-SSL web sites. Since Yahoo! is a highly available site, we use <http://mail.yahoo.com> (non-SSL) & <http://mail.yahoo.com> (SSL) here:

Case A (Non-SSL Yahoo Mail test):

```
user@proxy# telnet 172.17.37.25 3128
Trying 172.17.37.25...
Connected to proxy01.prod.nymex.com.
Escape character is '^]'.
GET http://mail.yahoo.com:80 HTTP/1.0<CR>
<CR>
HTTP/1.0 200 OK
Date: Thu, 07 Aug 2003 18:34:45 GMT
[...rest of the HTML output omitted ...]
```

Case B (SSL Yahoo Mail test):

```
user@proxy$ telnet 172.17.37.25 3128
Trying 172.17.37.25...
Connected to proxy01.prod.nymex.com.
Escape character is '^]'.
GET http://mail.yahoo.com:443 HTTP/1.0<CR>
<CR>
HTTP/1.0 400 Bad Request
Date: Thu, 07 Aug 2003 19:00:54 GMT
P3P: policyref="http://p3p.yahoo.com/w3c/p3p.xml"
[...rest of the HTML output omitted ...]
```

Note: these are carried out on the proxy server itself.

Lets go over this example in detail. First, when you issue the **telnet** command, be sure to specify the IP address that the proxy server is listening on. You cannot specify **localhost** because Squid has been configured to bind itself to a specific IP address (on the multi-nic host) for security purposes. You can determine the IP address using the same netstat command shown previously, or by issuing the command  
user@proxy\$ grep http\_port /usr/local/squid/etc.squid.conf.

The first example above is pretty straight forward. There is not much to say about it except that the successful response is **HTTP/1.0 200 OK**, followed by what is essentially header information and then the HTML code for the web page <http://mail.yahoo.com>.

The second example is also pretty straight forward, except that the response, **HTTP/1.0 400 Bad Request**, might lead us to believe that we have a problem. We don't; at least not from the perspective of the proxy server. The "Bad Request" response didn't come from the proxy server, it came from Yahoo. It is due to the fact that we used telnet, a plain text client that does not support SSL, to contact an SSL site (<http://mail.yahoo.com:443>). But whether or not Yahoo liked the format of our request is irrelevant. The fact that we got a response from Yahoo (good or bad), means that the proxy server allowed that request to go out to Yahoo over port 443.

## Troubleshooting the SQUID proxy server

*Note: There is a third case to test, but will be addressed later because telnet cannot help us with it here. It essentially involves generating an actual SSL request through the proxy server. Although port 443 is the de facto standard for SSL requests, to the squid proxy server 443 is just another port that it either allows or denies access to. Therefore, in this third case we are not so much concerned with whether the proxy server is allowing requests over port 443 (that much was proven in the second case above). Rather, we are concerned with whether the proxy server is properly handling (decoding and routing) SSL encrypted data that just happens to be commuting over port 443. More on this later.*

- (3) At this point we are comfortable that the proxy server is running and accepting requests from the local server. The next step is to perform exactly the same telnet tests from a remote client machine that can reach the proxy server (that is, any machine for which the command "**telnet Proxy-IPaddress 3128**" would not be prohibited by firewall restrictions). It would be wise to identify and document several candidate machines around the network that meet this criteria (they can be included here).

### Case A (Non-SSL Yahoo Mail test):

```
user@client$ telnet 172.17.37.25 3128
Trying 172.17.37.25...
Connected to proxy01.prod.nymex.com.
Escape character is '^]'.
GET http://mail.yahoo.com:80 HTTP/1.0<CR>
<CR>
HTTP/1.0 200 OK
Date: Thu, 07 Aug 2003 18:34:45 GMT
[...rest of the HTML output omitted ...]
```

### Case B (SSL Yahoo Mail test):

```
user@client$ telnet 172.17.37.25 3128
Trying 172.17.37.25...
Connected to proxy01.prod.nymex.com.
Escape character is '^]'.
GET http://mail.yahoo.com:443 HTTP/1.0<CR>
<CR>
HTTP/1.0 400 Bad Request
Date: Thu, 07 Aug 2003 19:00:54 GMT
P3P: policyref="http://p3p.yahoo.com/w3c/p3p.xml"
[...rest of the HTML output omitted ...]
```

Note: these tests are carried out on a client machine (not the proxy).

If the above tests do not indicate a problem (to be continued)....

- Check for name service issue.
- Configure Netscape or IE to use the proxy server and browse to <https://mail.yahoo.com>
- tail -f /usr/local/squid/var/logs/access.log
- squid commands

## Troubleshooting the SQUID proxy server

Sample /usr/local/squid/etc/squid.conf

```
#####
# Squid server squid.conf for NYMEX OpenSystems Application #
# use. (Milton Vega squid.conf v1.0). DO NOT EDIT! #
#####

#####
# Remember to define squidhost & thishost #
# in "/etc/hosts". #
#####
http_port squidhost:3128
cache_effective_user nobody
cache_effective_group nogroup
store_avg_object_size 13 KB

cache_access_log /usr/local/squid/var/logs/access.log
cache_log /usr/local/squid/var/logs/cache.log
pid_filename /usr/local/squid/var/logs/squid.pid
icon_directory /usr/local/squid/share/icons
error_directory /usr/local/squid/share/errors/English
mime_table /usr/local/squid/etc
unlinkd_program /usr/local/squid/libexec/unlinkd
cache_dir ufs /usr/local/squid/var/cache 100 16 256
coredump_dir /usr/local/squid/var/cache
cache_store_log none

refresh_pattern ^ftp: 1440 20% 10080
refresh_pattern ^gopher: 1440 0% 1440
refresh_pattern . 0 20% 4320

hierarchy_stoplist cgi-bin ?
acl QUERY urlpath_regex cgi-bin \?
no_cache deny QUERY

auth_param basic children 5
auth_param basic realm Squid proxy-caching web server
auth_param basic credentialsttl 2 hours

acl all src 0.0.0.0/0.0.0.0
acl manager proto cache_object
acl localhost src 127.0.0.1/255.255.255.255
acl to_localhost dst 127.0.0.0/8
#
acl SSL_ports port 443 563 8443
acl Safe_ports port 443 563 8443 # https
acl Safe_ports port 80 8080 # http
acl Safe_ports port 21 # ftp
acl Safe_ports port 22 # ssh
acl Safe_ports port 7001 # Weblogic
acl Safe_ports port 280 # http-mgmt
acl CONNECT method CONNECT
```

## Troubleshooting the SQUID proxy server

```
http_access allow manager localhost
http_access deny manager
http_access allow CONNECT SSL_ports
http_access deny CONNECT !SSL_ports
http_access allow Safe_ports
http_access deny !Safe_ports
http_access deny all
http_reply_access allow all
icp_access allow all

#####
# Added to allow purging objects from localhost #
#####
acl PURGE method PURGE
acl src_local src 127.0.0.1
http_access allow PURGE src_local
http_access deny PURGE
#####
mvega@proxy01.p
```