The Squid caching proxy

Chris Wichura

caw@cawtech.com

What is Squid?

- A caching proxy for
  - HTTP, HTTPS (tunnel only)
  - FTP
  - Gopher
  - WAIS (requires additional software)
  - WHOIS (Squid version 2 only)
- Supports transparent proxying
- Supports proxy hierarchies (ICP protocol)
- Squid is not an origin server!

Other proxies

- Free-ware
  - Apache 1.2+ proxy support (abysmally bad!)
- Commercial
  - Netscape Proxy
  - Microsoft Proxy Server
  - NetAppliance’s NetCache (shares some code history with Squid in the distant past)
  - CacheFlow (http://www.cacheflow.com/)
  - Cisco Cache Engine

What is a proxy?

- Firewall device; internal users communicate with the proxy, which in turn talks to the big bad Internet
  - Gate private address space (RFC 1918) into publicly routable address space
- Allows one to implement policy
  - Restrict who can access the Internet
  - Restrict what sites users can access
  - Provides detailed logs of user activity

What is a caching proxy?

- Stores a local copy of objects fetched
  - Subsequent accesses by other users in the organization are served from the local cache, rather than the origin server
  - Reduces network bandwidth
  - Users experience faster web access

How proxies work (configuration)

- User configures web browser to use proxy instead of connecting directly to origin servers
  - Manual configuration for older PC based browsers, and many UNIX browsers (e.g., Lynx)
  - Proxy auto-configuration file for Netscape 2.x+ or Internet Explorer 4.x+
    - Far more flexible caching policy
    - Simplifies user configuration, help desk support, etc.

How proxies work (user request)

- User requests a page: http://uniforum.chi.il.us/
- Browser forwards request to proxy
- Proxy optionally verifies user’s identity and checks policy for right to access uniforum.chi.il.us
- Assuming right is granted, fetches page and returns it to user

Squid’s page fetch algorithm

- Check cache for existing copy of object (lookup based on MD5 hash of URL)
- If it exists in cache
  - Check object’s expire time; if expired, fall back to origin server
  - Check object’s refresh rule; if expired, perform an If-Modified-Since against origin server
  - If object still considered fresh, return cached object to requester

Squid’s page fetch algorithm

- If object is not in cache, expired, or otherwise invalidated
  - Fetch object from origin server
  - If 500 error from origin server, and expired object available, returns expired object
  - Test object for cacheability; if cacheable, store local copy

Cacheable objects

- HTTP
  - Must have a Last-Modified: tag
  - If origin server required HTTP authentication for request, must have Cache-Control: public tag
  - Ideally also has an Expires or Cache-Control: max-age tag
  - Content provider decides what header tags to include
    - Web servers can auto-generate some tags, such as Last-Modified and Content-Length, under certain conditions
- FTP
  - Squid sets Expires time to fetch timestamp + 2 days

Non-cacheable objects

- HTTPS, WAIS
- HTTP
  - No Last-Modified: tag
  - Authenticated objects
  - Cache-Control: private, no-cache, and no-store tags
  - URLs with cgi-bin or ? in them
  - POST method (form submission)

Implications for content providers

- Caching is a good thing for you!
- Make cgi and other dynamic content generators return Last-Modified and Expires/Cache-Control tags whenever possible
  - If at all possible, also include a Content-Length tag to enable use of persistent connections
- Consider using Cache-Control: public, must-revalidate for authenticated web sites

Implications for content providers (continued)

- If you need a page hit counter, make one small object on the page non-cacheable.
- FTP sites, due to lack of Last-Modified timestamps, are inherently non-cacheable. Put (large) downloads on your web site instead of on, or in addition to, an FTP site.

Implications for content providers (continued)

- Microsoft’s IIS with ASP generates non-cacheable pages by default
- Other scripting suites (e.g., Cold Fusion) also require special work to make cacheable
- Squid doesn’t implement support for Vary: tag yet; considers object non-cacheable
- Squid currently treats Cache-Control: must-revalidate as Cache-Control: private

Transparent proxying

- Router forwards all traffic to port 80 to proxy machine using a route policy
- Pros
  - Requires no explicit proxy configuration in the user’s browser

Transparent proxying

- Cons
  - Route policies put excessive CPU load on routers on many (Cisco) platforms
  - Kernel hacks to support it on the proxy machine are still unstable
  - Often leads to mysterious page retrieval failures
  - Only proxies HTTP traffic on port 80; not FTP or HTTP on other ports
  - No redundancy in case of failure of the proxy

Transparent proxying

- Recommendation: Don’t use it!
  - Create a proxy auto-configuration file and instruct users to point at it
  - If you want to force users to use your proxy, either
    - Block all traffic to port 80
    - Use a route policy to redirect port 80 traffic to an origin web server and return a page explaining how to configure the various web browsers to access the proxy

Squid hardware requirements

- UNIX operating system (NT is not currently supported, nor has anyone announced work on a port)
- 128M RAM minimum recommended (scales by user count and size of disk cache)
- Disk
  - 512M to 1G for small user counts
  - 16G to 24G for large user counts
  - Squid 2.x is optimized for JBOD, not RAID

File system recommendations

- Use Veritas’ vxfs if you have it
- Disable last accessed time updates (for example, noatime mount option on Linux)
- Consider increasing sync frequency
- If using UFS
  - Optimize for space instead of time

Installing Squid (overview)

- Get distribution from http://squid.nlanr.net/
- Increase maximum file descriptors available per process *before* configuring Squid
- Run configure script with desired compile-time options
- Run make; make install
- Edit squid.conf file
- Run Squid -z to initialize cache directory structure
- Start Squid daemon
- Test
- Migrate users over to proxy

Squid distributions (versions)

- 1.x and 1.NOVM.x
  - No longer supported
  - Entire cache lost if even one disk in cache fails
  - Doesn’t understand Cache-Control: tag
  - Other problems
  - Bottom line: don’t use them

Squid distributions (versions)

- 2.0, 2.1, 2.2
  - Redesigned disk storage algorithm much improved
  - Understands Cache-Control: tag
  - Better LRU/refresh rule engine
  - Supports proxy authentication
  - See documentation for full list of enhancements
- Recommendation: 2.1 is fairly stable, but move to 2.2 when 2.2STABLE released

Squid compile-time configuration

- --prefix=/var/squid
- --enable-asyncio
  - Only stable on Solaris and bleeding edge Linux
  - Can actually be slower on lightly loaded proxies
- --enable-dlmalloc
- --enable-icmp
- --enable-ipf-transparent for transparent proxy support on some systems (*BSD)

Squid compile-time configuration

- --enable-snmp if desired
- --enable-delay-pools if desired
- --enable-cachemgr-hostname=<hostname> if using an alias for proxy or building on a different machine from the target proxy machine
- --enable-cache-digest and/or --enable-carp if using cache hierarchies

squid.conf runtime settings

- Default squid.conf file is heavily commented! Read it!
- Must set
  - cache_dir (one per disk)
  - cache_peer (one per peer) if participating in a hierarchy
  - cache_mem (8-16M preferred, even for large caches)
  - acl rules (default rules mostly work, but must reflect your address space)

squid.conf runtime settings

- Recommendations
  - ipcache_size, fqdncache_size to 4096
  - log_fqdn off (use Apache's logresolve offline)
  - Increase dns_children, redirect_children, authenticate_children based on usage statistics (see cachemgr.cgi front-end)
  - Tweak refresh_pattern rules (Danger Will Robinson! -- I suggest starting with examples found in the squid mailing list archives)

squid.conf runtime settings

- Recommendations (continued)
  - quick_abort_min 128 KB, quick_abort_max 4096 KB, quick_abort_pct 75
    - Tailor based on your bandwidth to the Internet
    - By default, squid will complete retrieval of any object requested, regardless of size; can burn considerable amounts of bandwidth!
- Too many other options in squid.conf to cover here; you really should read all the embedded comments!

squid.conf ACL example

- acl manager proto cache_object
- acl localhost src 127.0.0.1/32
- acl managerhost src 204.248.51.34/32
- acl managerhost src 204.248.51.39/32
- acl managerhost src 204.248.51.40/32
- acl cawtech src 204.248.51.0/24
- acl cawtech-internal src 172.16.0.0/16
- acl all src 0.0.0.0/0.0.0.0

squid.conf ACL example

- acl SSL_ports port 443 563
- acl gopher_ports port 70
- acl wais_ports port 210
- acl whois_ports port 43
- acl www_ports port 80 81
- acl ftp_ports port 21
- acl Safe_ports port 1025-65535

- acl CONNECT method CONNECT
- acl FTP proto FTP
- acl HTTP proto HTTP
- acl WAIS proto WAIS
- acl GOPHER proto GOPHER
- acl WHOIS proto WHOIS

squid.conf ACL example

- http_access deny manager !localhost !managerhost
- http_access deny CONNECT !SSL_ports
- http_access deny HTTP !www_ports !Safe_ports
- http_access deny FTP !ftp_ports !Safe_ports
- http_access deny GOPHER !gopher_ports !Safe_ports
- http_access deny WAIS !wais_ports !Safe_ports
- http_access deny WHOIS !whois_ports !Safe_ports

- http_access allow localhost
- http_access allow cawtech
- http_access allow cawtech-internal
- http_access deny all

Creating a proxy auto-configuration file

- Associate .pac extension with MIME type application/x-ns-proxy-autoconfig
- Create Javascript file and place on origin web server (suggest http://wwwinternal.domain.com/proxy.pac style URL)
- See Netscape documentation at http://home.netscape.com/eng/mozilla/2.0/relnotes/demo/proxy-live.html

Sample proxy auto-configuration

```
•  function FindProxyForURL(url, host)
•  {
•          if (isPlainHostName(host) ||
•                  dnsDomainIs(host, ".cawtech.com"))
•                  return "DIRECT";

•          if ((url.substring(0, 5) == "http:") ||
•              (url.substring(0, 6) == "https:") ||
•              (url.substring(0, 4) == "ftp:") ||
•              (url.substring(0, 7) == "gopher:"))
•                  return "PROXY proxy.cawtech.com:3128; DIRECT";

•          return "DIRECT";
•  }
```

Managing Squid

- I recommend the Calamaris.pl logfile analysis script, available at http://calamaris.cord.de/
- Use modified MRTG with Squid's SNMP support (see SNMP section in Squid FAQ for details)

Advanced topics briefly covered

- HTTP accelerator mode
  - Squid fronts a web server (or farm)
  - Particularly useful if server generates cacheable dynamic content, but generation is expensive
- Delay pools
- Cache hierarchies
  - Allows clustering and redundancy
  - World-wide hierarchies: NLANR, etc.

Squid resources

- Official web site: http://squid.nlanr.net/
  - Distributions
  - Mailing list archives and subscription info
  - FAQ
  - Link to Henrik’s web page for current patches and experimental features
  - Link to the Cache Now! web site (of particular interest to origin site implementers)
  - Lots of great information!