Web Server Organization

Harvest


Squid

Commonly used proxy server. The next version of Harvest. Why Squid? “All the good ones are taken” (Harris’ Lament). http://www.squid-cache.org/

Like Harvest it:

- never forks (single non-blocking process). State of all connections is kept within Squid—complex!
- non-blocking I/O
- keeps meta data and hot objects in VM
- caches DNS lookups
- can be arranged in a hierarchy (uses Internet Cache Protocol (ICP))

Available for most (all?) Unix platforms.

Current stable version is 2.5.

Private (single client) vs. public objects. Only public objects are saved on disk as part of cache.

Can use ICMP (ping) to determine nearest parent cache.
Cache Coherency

If-Modified-Since Get. If IMS Get received then handle in one of three ways:

- if object not in cache then handle as a MISS
- if object is in the cache and has a more recent timestamp, it is treated as a regular HIT
- otherwise the object is assumed to be valid and Squid returns a NOT MODIFIED response

Object Purge Policy

Uses a LRU replacement algorithm for cached (on disk) objects. Its aggressiveness in purging depends how much store swap space is available—the less space, the more aggressive in purging objects.

All objects within a hash bucket that exceed a LRU age threshold are purged. The entire cache is scanned every 24 hours.

Memory Use

Rough allocation of memory (assuming machine only runs the Squid server):

- 1/3 machine memory for storing objects (use high and low water marks for purging objects)
- 1/3 machine memory for per-object metadata
- 1/3 for other data structures, malloc(), etc.

Multithreading

What about using real multithreading? Could then be used effectively on a multi-processor.

Theoretically looks straightforward. Code is too complex now to seriously consider re-writing. Trade one set of problems for another.
Cache Digests

Caches share compact digests with other caches. Based on Bloom Filters. Paper: fan:sigcomm98.

Version 2 Features

- HTTP/1.1 persistent connections
- internal FTP support
- asynchronous disk operations (optional using pthreads library)
- URN (Uniform Resource Names) support
Web Server Benchmarks

Look at WebSTONE and SPECweb.

Generating Server Load

Look at Banga and Druschel paper from USITS97. Found that clients do not scale because they back off waiting for TCP connection. Build more scalable clients. Slides: http://www.cs.wpi.edu/~cs535/s03/banga:usits97/

Operating Systems Support for Busy Internet Servers

Paper in HotOS’95 conference by Jeff Mogul. Based on experience with Digital work in running large Internet information servers (IISs). Particularly 1994 California election service.

Characteristics of IISs distinguishing them from distributed systems applications:

- huge user base—several million people right now and growing
- short TCP connections
- long and variable network delays—different types of network service
- frequent network partitions
- no single administrative domain
- different penalties for failure—no one to complain to (see previous)
- no scheduled downtime

Makes comparison with transaction-processing systems in handling lots of short-duration requests, but notes they don’t require frequent, fast and synchronized updates of stable storage.
Other Observations

Lack of benchmarks (particularly high load).

How does the cost of a fork() compare with cost of select() handling large number of file descriptors—latter approach used in Harvest/Squid.

Notes potential problems of IIS systems in overload situation leading to livelocked.

Lack of scaling of some OS facilities—such as linear search for PCB entries versus more efficient approaches.

Operating System “Wish List”

- Direct control of timeouts and resources—to adjust many limits and time outs in the system. A different type of application
- Resource introspection—give IIS system capabilities to better understand system state
- disaster management. Need capability for diagnosis and control during exceptional periods.
Apache Web Server


Based on NCSA httpd 1.3 (early 1995) server. It is a Unix-based HTTP server. Most popular WWW server on the Internet.

Result of a group of core contributors with patches to the original server. Core group continues to control development.

A PAChy server—hence Apache.

Lots of features.