DynaCache: Weaving Caching into the Internet

Abdelsalam 'Solom' Heddaya

InfoLibria, Inc.

Abstract

World Wide Web traffic is swamping the Internet. In a recent study of backbone usage, HTTP packets comprised two-thirds of all activity[1]. Internet service providers are adding capacity as fast as they can, but they can't keep up with demand. Internet traffic doubles every six months[2]. The net effect is that per-user bandwidth in the Internet core is down to 50 Kbps[3]. Improvements at the edges—server load balancing and multi-megabit access technologies like ADSL and cable modems—only aggravate the problem. Caching is needed to speed up web access without requiring more bandwidth.

Web caching presents an appealing solution. By moving information closer to the users who want it and by eliminating repeated delivery of the same information, web caching can benefit everyone involved:

- End users enjoy faster web access.
- Network operators get bandwidth relief.
- Content providers see reduced load on web servers.

Yet with current technologies, the cost of web caching can outweigh the benefits. Burdensome configuration procedures, limited scaling, and increased load on routers and switches have restricted the deployment of caching products with many ISPs. And few network operators will tolerate the single points of failure created by most web caching architectures.

Existing products also neglect the business requirements of content providers. Most providers promise their users fresh information, and many base their service revenue on hit counts. But current caching products often deliver stale web pages and fail to forward hits and cookies back to originating servers—an unacceptable breach of the user-provider relationship. Consequently, many content providers choose to override web caching, canceling out the benefits to end users and network operators alike.

This paper looks at the evolution of web caching, from browser-based techniques through the first two generations of network-based products. It continues with a discussion of the DynaCache distributed

---

[1] K. Thompson, G.J. Miller, and R. Wilder, "Wide-area Internet traffic patterns and characteristics," *IEEE Network*, Nov/Dec 1997.

[2] MAE East Aggregate Input Traffic, http://www.mfsdatanet.com/MAE/east.stats.html , May 1998.

[3] *Boardwatch Magazine*, July 1997.

network caching solution from InfoLibria, Inc. Patent-pending DynaCache technology is a unique third-generation method of network-wide caching that improves Internet performance for everyone involved—end users, network operators, and content providers—while eliminating the risks and costs of earlier generations.

Proxy web servers

Network-based web caching has evolved through two generations and, with the introduction of the DynaCache, is entering a third. The first generation uses proxy servers to bring web pages closer to end users. The second generation adds transparency to simplify configuration and creates cache hierarchies and cache clusters to improve scaling.

Proxy cache servers are a crude, first-generation solution. End users must configure their own web browsers to take advantage of the proxy. And once a browser is pointed at the proxy, it can't reach the World Wide Web any other way—resulting in "forced HTTP routing." The proxy server quickly becomes a performance bottleneck and a single point of failure. The only workaround if the server crashes is for end users to reconfigure their web browsers—a support nightmare for the service provider.

Routing all HTTP traffic through the proxy can also have a negative effect on network performance. As shown in Figure 1, the browser may be topologically closer to the "real" web server than to the proxy server. Nonetheless, all requests must detour through the proxy. And even when the proxy server is on the shortest path, it still adds delay when overloaded.

Lacking a reliable mechanism for discovering when pages have changed, proxy servers are notorious for delivering stale web pages to end users. Moreover, proxy caches don't forward hits or cookies back to the originating server. The content provider counts a hit when a page is first retrieved, but subsequent hits to the cached copy go unreported. By masking hits, proxy web servers undermine a content provider's ability to generate revenue.

Semi-transparency

Second-generation web caching addresses two of the shortcomings of first-generation products. So-called transparent caches reduce configuration complexity, while manual hierarchies and clustering improve scalability.

With transparent caching, a router or LAN switch inspects packet headers and redirects HTTP traffic to the web cache. Browsers no longer need to point to proxy servers, so the challenge of getting users to set up their browsers correctly is eliminated. But the router or switch must be configured for redirection, making the technique only semi-transparent; browsers can't "see" the cache, but the router or switch can.

Although semi-transparent caches are easier to administer, they still share some fundamental weaknesses with proxy servers. With both approaches, any web object not in the cache passes through the ISP's router or LAN switch twice—once when it travels from the originating server to the cache and again as the cache forwards it back to the browser (Figure 2). Since a typical second-generation cache hits on less than 40% of all requests, over 60% of all web objects pass through the router twice. Caching may lessen backbone traffic, but end user response times can actually increase due to router or switch congestion.

Cache clustering

Besides overloading nearby routers and switches, proxy servers and semi-transparent caches can themselves become bottlenecks. An obvious solution is to use faster CPUs. But Moore's Law predicts that processor performance will double every 18 months, while Internet traffic is doubling every six months[4] (some even estimate it to double every 100 days)[5]. Obviously, the single CPU approach can't keep up.

Some web caching products offer clustered configurations that spread the load across multiple processors. This is effective up to a point, but it still can't match the growth rate of the Internet since the overhead of tight coupling limits throughput to 10 to 15 times that of a single CPU. Besides, tightly coupled multiprocessors are subject to multiple failure scenarios. Virtually every query to a clustered web cache involves more than one CPU. If any of the CPUs fails, the cluster is rendered useless for a significant subset of requests. Clearly a better method is needed for increasing scalability.

Manual hierarchies

Some second-generation products employ a manually configured hierarchy of caches to garner additional hits and to spread the load across the network. Typically, primary caches at ISP central sites are supported by secondary caches at one or more NAPs (Figure 3). In some cases there may be tertiary caches as well.

If a primary cache, which can be either a proxy or a semi-transparent cache, doesn't have a copy of a requested object, it queries its secondary caches. The secondaries search their storage and reply to the primary. If a yes comes back, the primary fetches the object from the appropriate secondary, copies it into local storage, and forwards the object to the browser that started the process.

If the answer is no, one of two procedures ensues. If so instructed by the primary, a secondary cache retrieves the web object from the originating server and notifies the primary when it has the object. The

---

[4] Above-mentioned MAE-East traffic measurements, and, "NetCraft web server survey," http://www.netcraft.co.uk/survey/, May 1998.

[5] "The Emerging Digital Economy," The Secretariat for Electronic Commerce, U.S. Department of Commerce, April 15, 1998.

primary then requests the object from the secondary, saves a copy, and forwards it to the browser. Alternatively, the primary itself retrieves the object from the originating server without the help of a secondary.

With a hierarchy of caches, scalability is enhanced because more processing power is applied to the overall problem. In addition, the hierarchy serves a larger group of end users. With more users, the chances improve that a desired web object is already cached. Where a primary cache might achieve a 40% hit rate, a primary and secondary together could get as high as 60%.

On the other hand, the manual hierarchy adds delay to many transactions. Even if an object is in a secondary cache, it takes two roundtrip messages to retrieve it: first the query/response from primary to secondary, then the actual retrieval. And if the object isn't in either cache, the delay of the query/response is still incurred. Because of this roundtrip delay problem, manual hierarchies can't go more than three deep, limiting their effectiveness in large internetworks.

Manual hierarchies also increase the burden on network operators. In the same way that browsers are configured to point at proxy servers, primary caches must be configured to point at secondaries. The forced HTTP routing problem is magnified, making it nearly impossible to ensure efficient routing. How can an ISP possibly keep track of all the web servers in the Internet?

Second-generation web caching offers some improvement over proxy servers. Scalability is better, both at individual sites and across the network. And configuration management is confined to ISP equipment. But several major first-generation problems are not addressed, either by semi-transparent caches or by manual hierarchies. Forced routing still results in inefficient network utilization and overloaded routers or switches. Scalability is still constrained by the limits of clustering and by the roundtrip delay of hierarchical systems. And nothing has been done to ensure the freshness of web pages or the accuracy of hit counts.

The DynaCache solution

With the DynaCache web server, InfoLibria introduces the third-generation of web caching. DynaCache architecture includes several innovations which eliminate the shortcomings of earlier generations:

- Loosely coupled distributed caching provides network-wide scalability without the limitations of manual hierarchies or clustering.
- In-line processing cuts miss latency to mere milliseconds and reduces router and switch loading instead of doubling it.
- Full transparency eliminates manual configuration and lets each web object travel the most efficient path available.
- A unique failsafe mechanism guarantees that a DynaCache web server is never a single point of failure.

- Real-time hit forwarding furnishes content providers with complete and accurate usage information.
- An efficient freshness distribution protocol ensures that DynaCache systems deliver only the most up-to-date web objects.

DynaCache web servers can be deployed throughout the Internet at local ISPs, regional sites, and NAPs (Figure 4). The servers automatically create a loosely coupled system that cooperates to move web content close to the users who need it.

A DynaCache web server comprises three major components: a rack-mounted server, a proprietary software kernel tailored for distributed processing, and the patent-pending DynaLink redirector (DLR). The DLR, which attaches the server to the network, is designed to support a variety of network media, including Ethernet, Fast Ethernet, Gigabit Ethernet, and SONET/SDH, for flexible deployment in diverse locations.

Fast, transparent processing

The DLR is a simple device that diverts network traffic through the DynaCache server (Figure 5). The server takes a quick look at each packet. HTTP packets are held for processing, while non-HTTP traffic is sent on its way at wire speed. Since the DynaCache server discovers web traffic on its own, it is fully transparent to all other network devices. There is no need to configure web browsers with proxy addresses or to command routers to redirect HTTP packets.

When the DynaCache server sees a web request, it checks its storage for a match. If the desired object isn't there, the server immediately forwards the request downstream, adding only a few milliseconds to overall network latency. The missed request may be fulfilled by a DynaCache server deeper in the network, or it may go all the way to the originating server. By not querying each other on misses, DynaCache servers avoid the roundtrip delay of manual hierarchies.

If it is busy, a DynaCache server can choose to pass web requests downstream. Again, the requests may be handled by other DynaCache servers, or they may go all the way to the originating server. In either case, the DynaCache system adapts automatically to network loading and never creates bottlenecks.

Failsafe operation

Besides supporting transparency, the DynaLink redirector is key to failsafe operation. The DLR continually monitors traffic passing to and from the server. If the server doesn't behave appropriately, the redirector automatically closes an internal connection and subsequent traffic bypasses the server (Figure 6).

The internal connection is an electromechanical switch, rather than a software mechanism or a silicon device. The switch defaults to the bypass position, so that even cutting the redirector's power cord cannot disrupt network traffic. In other words, a DynaCache server is never a single point of failure.

Automatic optimization and real-time hit forwarding

When a DynaCache server matches an HTTP request, it returns a copy of the object to the requesting browser. If there are DynaCache servers on the return path, they save a copy of the object as it passes by. Thus, frequently accessed objects quickly migrate towards end users (Figure 7), and a large percentage of web requests are fulfilled before they reach the Internet backbone. Indeed, DynaCache servers act as a shield, protecting the core of the Internet from overloading with popular web traffic.

When a DynaCache server fulfills a request, it also sends a hit report to the originating server. If the browser sent a cookie with the request, the cookie piggybacks on the hit report. The hit report and cookie travel to the content provider in real time, arriving when the web page request would have arrived. Thus the content provider gets complete, timely usage information without having to service every request.

Without forced HTTP routing, both web requests and returning objects follow paths laid out by internetwork routers—the same paths that they would use if there were no caches in the network. Thus the DynaCache system automatically uses the most efficient paths available at each moment. And network operators are freed from the configuration burden—and sub-optimal routing—of proxy servers and manual hierarchies.

Keeping content fresh

When a hit report arrives at an originating server, the server doesn’t send back the requested object. But it does tell the DynaCache server if the object has changed. If it has, the DynaCache server purges the stale version and, if the object is popular, fetches a fresh copy.

In addition, each DynaCache server shares freshness information with its neighbors. The neighbors share with their neighbors, and freshness updates spread quickly through the network (Figure 8). This mechanism, modeled on time-tested network news distribution protocols, keeps caches up-to-date without generating excess network overhead and without creating an explicit linkage between servers.

Neighboring servers find each other through a simple discovery process that works efficiently even when DynaCache servers are sparsely deployed. The process is continual, so servers automatically notice when new neighbors join or old neighbors leave. And since there are no static links between servers, they benefit from the self-healing capacity of router-based networking.

High availability, high scalability

DynaCache web servers comprise a loosely coupled distributed system that scales with the network. As with routers—and unlike clustered solutions—the overall throughput of the system grows without

increasing overhead on individual devices. And since each DynaCache server is self-sufficient, the loss of one unit does not compromise the operation of the others.

Because DynaCache servers don't query each other, there is no query/response delay and no practical limit to the number of servers that can work together. Unlike manual hierarchies, which can be only three deep, DynaCache servers can be placed wherever there are network hotspots. All of the DynaCache servers in the network cooperate transparently to move information closer to end users.

Since DynaCache servers are attached in-line, a newly fetched web page passes through each router or switch just once (Figure 9). DynaCache servers speed up web access and off-load backbone circuits without causing router or switch congestion in return.

Conclusion

Earlier generations of web caching may improve end user response time or off-load backbone links, but they all sacrifice reliability, scalability, and content integrity in return. Third-generation DynaCache web servers from InfoLibria are the first solution to deliver all of the benefits with none of the drawbacks.