Hal Berghel

Responsible Web Caching

Putting the distribution control of the Web's intellectual content in the hands of those who created it.

Web caching is not a burning issue in IT. A few people are motivated to improve caching techniques for the sake of network efficiency. But the overwhelming majority of IT professionals would sooner have their teeth scraped than review the literature on Web caching. Like other arcane areas in networking (load balancing, TCP/IP protocols, Domain Name Servers (DNSs), and network topologies, to name a few), most of us are content to know these issues are important to our communication practices, and that bright, industrious people are looking after our interests by studying them. Web caching, however, falls into another category altogether. Like cookies, Web caching protocols are examples of how the best of intentions can result in implementations with highly undesirable side effects. One aspect of modern Web caching is just socially irresponsible, pure and simple.

Benign Caching

In the world of networks, caching is an important optimization tool, sharing this trait with replication and mirroring. Networks generate a lot of background "chatter" to keep the data moving along smoothly, and if it weren't for these optimization tools, the volume of chatter would overwhelm the useful data and communication would come to a halt.

DNS performance is a good example. When we access a computer on the Internet, we are actually targeting a unique computer with a unique 32-bit IP address broken into four octets. Thus, "how.what.why.when.where.com" would be resolved to an address like "138.21.4.218." This name resolution takes place in a DNS—more accurately, translation takes place somewhere within a hierarchy of DNSs that collectively make up a large distributed database. DNSs were set up as a distributed database because the pioneers of Internet functionality discovered in the early days that managing DNS names was an inherently nonscalable activity.

So, when we attempt a connection with "how.what.why.when.where.com" we must rely on authoritative address resolution into IP addresses or address fragments (domains). But, there will likely be several DNSs that have authority over the respective domain fragments involved—one authoritative root server for the top-level domain (.com), another DNS for the next lower domain (.where), and so forth. This model will work fine as long as these domains are linked together within the hierarchy, so that each DNS knows how to reach authoritative name servers for the parent and child nodes. In the example, the complete address resolution of "how.what.why.when.where.com" would entail the iterative process of getting the address for the authoritative DNS for "where.com" from the ".com" root server, then getting the authoritative DNS for "when.where.com" from the DNS for "where. com" and so forth down the tree. However, these DNSs may be scattered all over the Internet. Herein lies the rub. Without optimization, our authoritative root server for ".com" is going to receive a request every time someone tries to access any computer in that domain; it couldn't handle all this traffic.

MASAKO EBATA

Enter caching. Caching shares "bindings" of computer names with their corresponding IP addresses after these have been resolved with other DNSs that have accessed the information. When I access "how.why.what.when. where.com," the local authoritative name server stores the binding of "how.what.why.when.where.com" with "138.21.4.218" in its local cache. The principle of locality of references suggests that if I access this computer once, I'm likely to do it again, so caching the binding will take unnecessary address resolution traffic off the network above the level of the local server. I just have to add an instruction to my network operating system to look to the local cache for address resolution before passing the request up the hierarchy of the DNS.

DNS caching falls into the category of benign caching. Similar benign caching takes place at the lower, physical network layer through the Address Resolution Protocol (ARP); only there the bindings are from IP addresses to actual physical addresses of the computer built into the hardware of the motherboard or network card (type "ipconfig /all>" at a command prompt to find yours).

Malignant Caching

Web caching goes beyond the benign. In its extreme form, it routinely violates the principles modern societies routinely use to manage intellectual property up to and including overt copyright infringement. For want of a better term, I'll refer to this as the "malignant" strain of Web caching.

This is not to deny there are benign aspects of Web caching. In its most benign form, Web caching achieves efficiency through a document delivery environment built upon content and location transparency. Other things equal, if we can get current versions of sought-after information more efficiently, what's not to like? Of course the hook is "other things equal." More on that later.

Two examples of increased efficiency of Web document access that satisfy the simultaneous objectives of increased efficiency and content and location transparency are server caches and browser caches. In the first case, servers may cache frequently accessed documents in primary memory to avoid slow disk accesses, or server farms may place high-demand data on faster, dedicated servers for load balancing.

In the second case, the documents are actually served on the client itself. An example of this is a browser cache. The browser stores frequently accessed data on the client's hard drive so subsequent access can entirely avoid network access altogether. As with server caching there is no issue with subsequent reuse or redistribution of the resource or document that is inconsistent with the owner/author objectives. The client's browser cache behaves in much the same way a local DNS would use name resolution bindings. Server caching and browser caching are both ephemeral and directly linked to the original source.

Such is not the case with the potentially pernicious form of Web caching on proxy servers. Here matters become murky. Proxy servers are network middlemen that stand between computers. Typically, a proxy server would be located between a client and server. In such a case, the proxy would interact with the server on the client's behalf, and vice versa. Besides proxy caches, other examples of proxy servers are proxy firewalls, anonymizers, and remailers.

Proxy servers simultaneously satisfy the two conditions of efficient and content/location transparency, but are neither ephemeral nor directly linked to the original source. The proxied cache becomes a surrogate for the original source. This surrogate does not come without ethical implications.

Proxy Caching

The original version of Web support in HTTP 1.0 was fairly anemic. While the implementation details are only of historical interest at this point, suffice it to say the three relevant headers were included to either determine whether requested documents were "fresh" or "stale," with the end of circumventing the cache in favor of the originating host if the documents in question were out-of-date, or to prevent documents from being cached at all.

The overwhelming majority of IT professionals would sooner have their teeth scraped than review the literature on Web caching.

HTTP 1.1's treatment of Web caching is much more sophisticated. Considerable flexibility has been added for making finer-grained distinctions between "fresh" and "stale" documents, including the use of heuristics rather than server-supplied file time-and-date stamp comparisons. A validation model was included that added a "validator token" to the cached document. With a cache-resident validator, document currency could be confirmed by the proxy cache by simply running the validator from the originating server. The currency is reported by the response status code. In fact, there are two types of validators, and correspondingly, two degrees of validation: strong and weak, depending upon the degree of "freshness" one needs. But by far the most important difference in the HTTP 1.1 treatment of caches was the addition of the cache-control header field.

The idea between the cache-control header field was to provide a general mechanism for caches to communicate efficiently and effectively with other computers. The point that should not be overlooked is by design the cache-control header field provides directives focused on regulating requests and responses along the caching food chain. The general structure of the header is:

Cache-control : directive
[optional parameters]

where the directive consists of keywords relating to aspects of requests and responses. Since the more controversial aspect of cache-control deals with responses, we'll limit our discussion accordingly:

- **Cache-control = "public"**
  This means responses from this server may be stored without restriction.
- **Cache-control = "max-age = 43,200"**
  This means the document contained in this response should be considered "stale" after 43,200 seconds (approx. 30 days).
- **Cache-control = "must re-validate"**
  This means the caching service (proxy or browser) must revalidate the document after it becomes "stale" from the originating server, or report an error message. It must not pass on the "stale" version.

You get the idea. Now, the worrisome directives:

- **Cache-control = "private"**
  This means only private cache services (not shared cache services) may cache the contents of this response.
- **Cache-control = "no-cache"**
  This means responses can be cached, but the cached copy may not be reused for subsequent requests without revalidating the cached copy with the originating server. (The baroque logic behind this directive is a work of art. When we say "don't peek," does that normally mean "peek only once"?)
- **Cache-control = "no-store"**
  This means this response may not be cached on nonvolatile storage.

The Rub

It doesn't take a rocket scientist to review the list of Internet Cache Protocol (ICP) cache-control directives and determine its approach to intellectual property management falls somewhere between irresponsible and draconian. Proponents of Web caching in its present form subscribe to the "anything on the Web is community property" school of thought. This is made abundantly clear in the crude way document ownership is skirted in the last three directives previously discussed.

Let me suggest some topics directly relevant to the issue of

ownership, together with what many of us consider reasonable concomitant rights and responsibilities, for which there are no provisions in the ICP:

- Does the cache operator own the resource/document being accessed?
- Is this resource/document copyrighted, and if so what is the nature of the license for subsequent redistribution..
- Is there a "definitive" version of this document (the published and copyrighted version)? If so, where is it located?
- Does the cached document have a digital object identifier? If so, what is it?
- Did the owner/copyright holder restrict the use of this resource/document (for example, for classroom use, for use by nonprofit corporations, for use other than commercial purposes)?
- Is it realistic to expect the owner/author to familiarize himself or herself with the nuances of the cache-control directives before placing anything on the Web?
- Should the owner/author be expected to define the caching parameters that take into account all possible effects (for example, would the typical author interpret "no cache" to mean "don't cache without revalidating")?
- Did the owner/author relinquish control when a resource/document may be withdrawn from public circulation by placing it on the Web? (Once the document is cached, there's no simple and immediate way of withdrawing all cached copies from the Web.)
- Is the owner/author entitled to maintain accurate usage statistics and logs regarding resources of their creation? (These statistics are more difficult to gather from caches, demand additional work, may (as in the cost of embedded Web bugs) have privacy implications, carry with them a definite cost to the owner/author/information provider, and require the universal cooperation of cache maintainers to work. Cache metering is not catching on for these reasons.
- Is it the responsibility of the owner/author to ensure version control mechanisms are in place to prevent outdated versions from being circulated by cache services?
- Is it the responsibility of the owner/author to continuously monitor every proxy cache operation and issue a "take-down" notice each time some of his or her intellectual property is discovered in some cache?
- Is the owner/author of a Web document/resource entitled to reasonable royalties or profit-sharing that accrue to downstream Web caching services that host his or her work?

I predict answers to these questions will determine how sympathetic one is to the objectives of proxy cache services. Regrettably, few people seem to even ask these questions, much less try to develop viable answers.

Caching or Bust

Those of us who oppose Web caching in its present form run the risk of being labeled "cache busters." But the label just doesn't fit. I know of no one who opposes the technology of caching as such. But, like cookies, caching technology is being misused. Members of the Web community that respect the ownership of intellectual property may be forced into cache busting because there are no other reasonable alternatives in the cache-control directives to protect their intellectual property. It is worth remembering that under HTTP 1.1, "no store" is the only option available for those who want to control the dissemination of their intellectual property. "No store" is far too coarse to be useful to the Web community in general.

The "no-store" directive shares this deficiency with the robot exclusion standard built on the premise that the primary objective of agent indexing controls should be the convenience of the system administrators. The idea that the millions of Web developers would organize their sites on the basis of what should be indexed by search agents rather than natural semantic clusters is preposterous. But that's the only way the robot exclusion standard can work. So it is with the binary "no-store" directive. Web developers are given the choice of either preventing any caching or giving up complete control of their intellectual property. I'm sorry, but the real world doesn't limit us to a succession of equally unpleasant choices in a way the "no-store" directive does.

The defenders of intellectual

property rights face formidable foes. The Web cache community seems to embrace the position there should be no expectation of proprietary ownership for anything placed on the Web. Usually this takes the form of remarks like "the Web is about copying—if you don't want your work copied, don't put it on the Web." This is an exceedingly simplistic and narrow-minded view of the Web.

While the world recognizes a difference between perusing the intellectual property of others on the one hand, and making and distributing copies of it on the other, this distinction seems to have completely escaped the attention of the ICP creators. When teachers provide their students with perusal copies of written work, designs, computer programs, artwork, and so forth, there is no implication that they surrender any ownership rights. Nor is there any implication that the teacher has given the students a license to copy, distribute, or sell copies. To claim there is such an implication would be patently absurd. Artists don't automatically surrender rights when they agree to have their paintings displayed in a gallery. Neither does the owner/author of Web content.

The Google search engine is a clear example of this abuse. One of the things that made Google distinct from other Web search engines was that it cached the content it indexed, without, mind you, any regard to the questions I posed in the previous section. Google generates revenue from having a large index of cached intellectual property it doesn't own and, for the most part, hasn't received permission to cache and redistribute. Some have a problem with this—publishers in particular. In one of life's little ironies, publishers and authors are not necessarily on the same page on this issue.

To illustrate, I'll use the example of publishing an article in *Communications*. Let's think of a linear model of producing a publication (for description of the more realistic, nonlinear model, see my "Cyberpublishing Manifesto" (*Communications*, Mar. 2001). On this account, an author completes some research or comes up with a new idea and commits it to writing. Continued reflection and input from external sources produces *k* iterations of this document, only the last of which is submitted for editorial consideration. *Communications* sends the

Berghel's URL Pearls

- Generic information about Web caching, including recent news flashes, may be found on the Internet Caching Resource Center Web site; www.caching.com.
- ICP specifications may be found at icp.ircache.net.
- The International Web Content Caching and Distribution Workshops have been held annually since 1996. Program information and proceedings can be found at www.iwcw.org. The focus of these workshops is technical, with no evidence there is much attention paid to underlying ethical issues.
- Duane Wessel's Web Cache can be found at www.web-cache.com. Wessels penned what is to my knowledge the most complete reference book on Web Caching (O'Reilly, 2001). He also coauthored Version 2 of the RFC for Internet caching protocol specifications. Wessels pays lip service in both his book and Web site to the ethical and privacy issues of Web caching.
- Many caching programs for a wide variety of Unix and Windows NT/2000/XP servers are available, including:
  - The Open Source Squid—Web Proxy Cache (www.squid-cache.org);
  - Netscape's Proxy Server (wp.netscape.com/proxy/v3.5/);
  - Microsoft's caching software is built into their Internet security and acceleration server product (see www.microsoft.com/isaserver); and
  - The CiscoCache Engine Series (www.cisco.com/warp/public/cc/pd/cxsr/500/index.shtml)
- The ACM copyright policy is online at www.acm.org/pubs/copyright_policy.
- Digital Object Identifiers are described at www.doi.org. C

Coming in 2003

JANUARY
Digital Government

FEBRUARY
Peer-to-Peer Computing

MARCH
Attentive User Interfaces

APRIL
Digital Rights Management

MAY
Wireless Networking Security

JUNE
Marketing Digital Products

See the Communications Web site (www.acm.org/cacm) for the complete 2003 Editorial Calendar

material to reviewers, adds editorial value, requests revisions from the author, formally accepts the *k*+1st version, copyrights it, assigns a unique document object identifier (DOI), and publishes it as the "definitive" version. Along the way, a subset of the first *k* versions are likely to have found their way onto the Web.

The ACM copyright policy (by anyone's measure a very author-friendly policy) is clear on several critical points relating to this process. For one, if any of the first *k* versions are substantially the same as the accepted version, the ACM copyright notice and a link to the definitive, copyrighted version must be attached after the article is accepted. Google doesn't recognize this copyright arrangement at all. In fact, under the current contractual arrangement with ACM, Google agrees only not to serve up "full text" versions of definitive copies. There is no mention at all of prior versions that bear the ACM imprimatur, never mind any consideration given to the earlier versions that may be copyrighted by the author. Under the current Web caching scheme, the author has no protection against copyright infringement.

The Devil's in the Details

The optimization side of Web caching is indispensable to the successful deployment of the Internet. The intellectual property side is a disaster.

The solution to the problem is the engagement of people who are sensitive to intellectual property issues, not just those interested in improving network efficiency, in a complete overhaul of the ICP. The new version should explicitly contain cache-control directives that are socially responsible when it comes to authorship, providing appropriate credit, delineating just how much latitude the owner/author has licensed regarding subsequent distribution, whether royalties are required, definitive versions, copyright, and so forth.

As a modest first step, consider the following directives:

```
Credits required = list
Copyrighted = "yes/no"
Copyright holder = name
DOI = value
Serveup = "full text | extract |
    keywords | URL" [inclusive or]
Extract criteria = "word limit =
    value," "document abstract =
    yes/no," etc.
Royalty required = "yes/no"
Methods of royalty payment = list
…
```

The idea is to put the control over the distribution of the intellectual content of the Web in the hands of those who created it in the first place. C

Hal Berghel (www.acm.org/hlb) is a frequent contributor to the literature on cyberspace and professor and chair of computer science at the University of Nevada at Las Vegas.