



LAN Based Web Caching Q&A - Part Two

This is the second part of the Web Caching Q&A. It takes a more in-depth look at caching technology, and will attempt to go beyond the LAN and out onto the Internet. If you are in doubt about any of the topics relating to caching technology, Part One of the Web Caching Q&A provides an introduction to the subject.

Part Two: Questions

1. What are the hardware requirements of a web caching server?
2. How does one fine tune a web caching server?
3. What is the caching directive and how does it work?
4. What should be done if web servers do not properly implement the caching directive?
5. A word about history mechanisms.
6. Doesn't caching actually slow down content delivery because it must first store the data before delivering it?
7. How does caching affect multimedia or streamed content?
8. What reports and statistics do web caching servers provide?
9. What is a hierarchical caching system?
10. How does hierarchical caching work?
11. Is hierarchical caching of use on a LAN?
12. How can I be sure that the pages of my web site are never cached by caching servers on the Internet or on users' LANs?
13. How can a caching server know if the content it receives in a response is actually fresher than the content it currently has in its cache?
14. If a web page is delivered over multiple paths, and through multiple caching servers, how does the caching mechanism ensure that the different page elements are coherent among themselves?
15. When I place an order with an on-line e-commerce web site, will confidential information get saved on a web cache?
16. Do caching servers store copies of cookies?
17. Do web caching servers deprive some commercial portal web sites of hits and revenues?
18. Can web caching be used to pre-emptively download frequently viewed pages at off hours?
19. Conclusion

Part Two: Answers

1. What are the hardware requirements of a web caching server?

Web caching servers have varying hardware requirements. Before making a decision as to which caching server to implement it may be advisable to compare the hardware requirements of various vendors. There are some common baselines however.

The primary task of a caching server is to store items and to retrieve them for ulterior delivery. Disk I/O is therefore of paramount importance, and pure CPU power is secondary. It is critical to choose the fastest hard disk available for the hardware configuration to be used, both in terms

Content of this page in its entirety is protected by US & UK Copyright
© 2002 Vicomsoft Ltd

Reproduction in electronic and written form is expressly forbidden without written permission.



of access time and data throughput. Also, the larger the hard disk the better. The more disk space there is, the longer cached items can be kept, and the less frequently they need to be refreshed from the originating server. If the disk is full, the caching server will be required to discard old items to make room for new ones. These items may be discarded while they are still fresh. With adequate disk space this will happen less often and performance gains will be realized.

While disk performance is a key element, it is still necessary to have enough processor power and bus capacity to push the data out the network interface card (NIC). Optimal performance will only be achieved if the host computer can saturate the available LAN bandwidth.

2. How does one fine tune a web caching server?

A number of parameters are involved in the tuning of a caching server. As mentioned above, the total size of the cache is important. Having a large cache will not only allow items to be kept longer, it will allow larger items to be cached.

Some users download very large items such as video files. It would not take many of these to fill up a cache. Regardless of how much disk storage is available some upper limit must be placed on the size of files that may be cached. A common value for this parameter is about two percent of the available disk space. If users download objects greater than the maximum cached object size, none of these objects will be cached. Subsequent requests for them will result in content being fetched from the source. If this occurs too frequently the result will be unnecessary use of bandwidth and slow download times. It is important to know and understand usage patterns in order to best decide the size of items to be cached. Again, the larger the maximum cached object size, the more disk space will be needed.

Because caching servers often store very large numbers of relatively small files, disk fragmentation may occur. Regular disk maintenance and defragmentation can have a positive effect on cache performance.

The overall efficiency of the caching mechanism will also be affected by the way that incomplete or missing caching directives are handled (see below).

3. What is the caching directive and how does it work?

Web pages are delivered from servers to browsers using the HTTP (Hypertext Transfer Protocol) protocol of the TCP/IP protocol suite. Each HTTP packet has a header containing a number of fields. Two of these fields are used by caching servers to decide how long to cache content, or even if it should be cached at all. The "Expires" field contains a date and time after which the data should be considered stale or invalid. Once the data is invalid, under normal circumstances it should not be served until it becomes valid once again. If the data is requested, the caching server should first revalidate it by comparing the last modified date of the cached data with the last modified date of the data on the originating server. If the two dates are the same, the data may be revalidated and served to the requester. If not it should be refreshed with new data from the originating server.



If the originating server is uncertain about how long the data will remain valid, it can allow caching servers to cache the data, but indicate that it must not be served without revalidation. To accomplish this it includes a date from the past, or a zero in the "Expires" field. The caching server will then cache the data but consider it to be immediately invalid. Subsequent requests for the data will cause the caching server to revalidate the data before serving it. If the originating server indicates that the cached data is still fresh, it may still be served from the cache, saving bandwidth and improving network efficiency. If not it will simply be fetched from the originating server again.

In the above example, data is cached although it is considered to be stale already. This technique will not prevent the caching server from caching the data, it will only prevent it from serving the data on subsequent requests without revalidation. There are times when an originating server does not want data to be cached at all. To this end there is another field called "Cache-control". If this field contains a value of "no-cache" then caching servers should not cache the data.

4. What should be done if web servers do not properly implement the caching directive?

Some web servers provide incomplete or inaccurate caching directives. Some do not provide caching directives at all. In such cases it is possible to instruct the caching server to override the directives if there are any, or to apply specific rules if there are not. For example an administrator may wish to allow a maximum age of 24 hours to `www.any-site.com` even though the caching directive indicates a maximum age of one week. There are many legitimate reasons for doing this. If a news site does not include caching directives, a web caching server will likely cache it by default. Users may then find themselves reading yesterday's news. To solve this, the caching server should allow the administrator to override the originating server's caching directive. Of course there will be a performance penalty each time it becomes necessary to fetch a page from the source rather than from the cache. It is up to the cache administrator to see that an optimal balance is maintained between performance and content consistency.

5. A word about history mechanisms.

Many user agents incorporate a history mechanism. In browsers this is usually in the form of a "back" function or possibly a "history" window. This allows users to revisit recently displayed pages quickly and in most cases without waiting for delivery from the Internet. A history mechanism involves the use of a cache. All of the rules regarding caching servers however do not necessarily apply to history mechanisms. The purpose of a history mechanism is for the user to be able to see a history of what they have already viewed. This may mean that they want to see exactly what they had previously viewed rather than a new updated version of the page that they had previously viewed. It is therefore perfectly legitimate for a history mechanism to retrieve and deliver stale content, whether to an individual user or in a collaborative environment. Caching servers are not history mechanisms, but a history mechanism may use a caching server to retrieve historical content. It is up to the designer of the mechanism to offer all of the reasonable alternatives to the end user. Current standards allow a history mechanism to redisplay any content, dynamic or static, immediately and without revalidation with the originating server, although some browsers may not yet fully implement these standards.



6. Doesn't caching actually slow down content delivery because it must first store the data before delivering it?

If data were first stored by the caching server and then delivered, it would be slower than normal browsing. However, not all caching servers cache first and deliver later. Advanced web caching servers enjoy an innovation known as concurrent caching and delivery. This feature allows the caching server to stream content to multiple users' browsers while it is simultaneously being written to the cache. By the time the content has been stored to disk, it has already been delivered to the requester with no additional delays introduced by the caching mechanism.

However, even if the caching server does not support concurrent caching and delivery, what really matters to end users is the average time it takes to retrieve content and display it on their screen. If a user views 100 pages on a given day, and the total time it takes to deliver and display those pages is 15 minutes, a solution that could reduce this time to five minutes would be appreciated. The user would not worry that *certain* pages actually took a fraction of a second longer to display. This is what web caching does. It dramatically speeds delivery of *some* of the pages viewed, thus improving *overall* efficiency by a noticeable amount.

There are other factors that come into play as well. Performance gains achieved through optimal use of bandwidth may well outweigh any minimal losses incurred by the caching mechanism.

7. How does caching affect multimedia or streamed content?

The expression "streaming media" is used to describe audio and video content that is delivered in real time directly from the source to the end user. When this happens, the user experience is similar to watching television or listening to the radio. As such, by definition streaming media cannot be cached. First, the continuous uninterrupted stream of data would soon fill up the user's hard disk. Secondly, if cached, they would no longer be "streaming media" but rather stored and played back. Caching can therefore do nothing to improve their delivery. Unfortunately however, caching may adversely affect it. Proxy servers that do not offer support for TCP/IP protocols other than HTTP may prevent streaming media from reaching the client computer. This is because streaming generally uses different protocols than HTTP, and proxy servers do not always support them. A proxy caching server may therefore block all forms of streaming media. Some non-proxy caching servers allow streaming media to pass through to a single client, in which case a single user on the LAN would be entitled to view streaming media at any one time. Other non-proxy caching servers may deliver streaming media to multiple clients. In this case operation would be entirely transparent.

Not all multimedia content is streamed. Streaming media should not be confused with multimedia content stored on disk to be retrieved and played later. When such multimedia content comes from the Internet as part of a web page, it is delivered via the HTTP protocol and therefore is cached just like any other web content. Repeat requests will be met with improved delivery. Performance gains where large video files are involved can be truly dramatic with delivery times dropping from minutes to seconds.



The effect of web caching servers on multimedia content therefore will be beneficial when the media are delivered as part of a web page using the HTTP protocol. The effect of web caching servers on multimedia content that is delivered using protocols other than HTTP will be neutral so long as the caching server allows their delivery to the end user.

8. What reports and statistics do web caching servers provide?

Internally, web caching servers function somewhat like web servers in that they fetch web pages from their own local storage area and serve them to web browsers. Because of this it is possible for them to keep statistical logs that can be used to analyse traffic patterns. An industry standard for this type of log is the Extended Common Log Format (CLF) that lists all served web items.

Log files in this format may be analysed with any CLF compliant program. One such popular log analyser utility is Analog. The use of Analog to analyse CLF files will allow the administrator to determine what percentage of pages are served from the cache, for example. This percentage can be broken down by domain, type of browser etc. Intelligent use of caching statistics can help forecast server load and bandwidth usage and determine caching parameters to achieve optimal performance.

9. What is a hierarchical caching system?

Up until now all of our discussions have taken for granted a scenario involving a single web caching server between the end user and the content source. What we have not yet made clear is that the caching server may itself request pages from another caching server. If this server is higher up in the hierarchy, it is known as a parent. Each request is fed upstream closer and closer to the source until a parent caching server is found that has a fresh copy of the data in its cache, or until the request ultimately reaches the source of the content. This structure is referred to as a cache hierarchy or mesh. Hierarchical caching structures are commonplace on the Internet. National and international caches serve very large areas and may receive hundreds of thousands of page requests per day.

10. How does hierarchical caching work?

Purely hierarchical systems do have limitations, notably if they are too deep. If there are too many caching servers between the requester and the source, the net delay introduced by all of the relayed requests and responses may be self-defeating. Further, the number of messages sent among caches could actually contribute to increasing rather than decreasing network congestion. Simply decreasing the number of caching servers at large does not solve the problem, as the distance between servers would increase, causing a corresponding increase in the hop count of each request and response, placing unnecessary burden on network routers which would, again, be self-defeating. The solution is to increase the number of similar caches at national and regional levels having a similar status. That is, they do not all have a parent-child relationship. They can also have a peer-to-peer relationship and as such are known as siblings. In a caching mesh one cache may ask a sibling if it has a copy of specific content that is fresher than its own copy. This avoids repeatedly asking the parent cache every time revalidation is needed or uncached content is requested. The result is a reduction in the load on parent caching servers and a reduction in network traffic due to requests and responses. Since



a cache has its own list of nearby siblings, it can revalidate old content or acquire new content following the most efficient path.

11. Is hierarchical caching of use on a LAN?

Hierarchical caching is of little use on a LAN as delivery between any two points on the LAN is extremely efficient. As caching appliances become commonplace in corporations, and Intranet technology causes web based solutions to supplant older methods of information retrieval and delivery, hierarchical caching may emerge as a means of reducing traffic across zones and through enterprise routers. The advantages of using hierarchical caching on enterprise WANs are easy to demonstrate as WAN bandwidth is limited and more expensive than LAN bandwidth.

12. How can I be sure that the pages of my web site are never cached by caching servers on the Internet or on users' LANs?

Web pages are made up of HTML (HyperText Markup Language) but they are delivered from web servers to web browsers via a protocol known as HTTP (Hypertext Transfer Protocol). Most web users have some idea of what HTML is, but few understand HTTP. The former is a page description language, while the latter is a protocol. A protocol is a set of rules used by two computing devices to determine how to communicate. It includes, among other things, a predetermined message format to ensure that each device understands the other. HTTP requests are sent from the web browser to the server and responses are sent from the web server to the browser embedded in HTTP packets. Each HTTP packet has a header that contains information of use to software programs and of no interest to humans. The average user never sees an HTTP header (the HTTP header is not to be confused with the <HEAD> section of an HTML document). What we are discussing here is the HTTP protocol header that is read by servers and browsers to determine how to manage the information in the HTTP packet. For example the Content-Type header field may state that the HTTP packet contains text. This is important information to the web browser which needs to know if the data is an image or if it is text. By the time the browser displays the information the user knows very well that it is text. But the browser would not know unless it were told so in the HTTP header.

The HTTP header has many fields. One of them is called the Cache-control field. If a web site contains content of a nature that should not be cached, the web server should include a "no-cache" directive in the Cache-control field of the HTTP header in compliance with RFC 2068. It is the responsibility of the webmaster to configure the web server to do this for each page that should not be cached. Failure to do so is implicit consent to cache web content. If the "no-cache" directive is included, caching servers on the Internet are expected to comply. Caches on users' LANs set to comply with originating servers' caching directives will also respect the "no-cache" directive.

Any web page that is generated dynamically as a result of a database query generally has a URL containing a question mark, and therefore will not be cached unless administrators of private LAN based caching servers specifically override standard caching behavior. There would be no harm in including the no-cache directive in the HTTP headers of such pages however.

13. How can a caching server know if the content it receives in a response is actually fresher than the content it currently has in its cache?

Hosts running web servers or web caching servers should use NTP or similar protocol to ensure their clocks are reasonably accurate. Not everyone does. Even if they do, when a server receives a signal telling it what time it is, it is difficult for it to know just how long the signal spent in transit. The net result of all of this is that there is some discrepancy among server clocks.

HTTP headers may include a date-time stamp. As long as all of the web caching servers in the path followed by the content include a date-time stamp, or all of the system clocks are reasonably synchronized, identical cached copies of a resource should indicate the same age. But incidents may arise where this is not the case. If that should happen, a cache could request a resource and receive a response with an older header date than the one it already has. In this case it may ignore the response and use the copy it has. Alternatively it may reiterate the request indicating zero tolerance for content age.

14. If a web page is delivered over multiple paths, and through multiple caching servers, how does the caching mechanism ensure that the different page elements are coherent among themselves?

The set of circumstances that can produce the situation described above can also have some other interesting side effects. Because data can follow multiple paths between server and client, and some web pages are even made up of elements from more than one server, the page elements that make up a page may have come from different caching servers between the browser and the originating web server.

If a client or a caching server receives a response with a date header that appears to be older than the one it currently holds, it simply reiterates the request using the Cache-control header to indicate its intolerance of cached content. This will cause a fresh copy to be fetched from the server and all of the caches in the chain will be forced to refresh their copies of the resource.

15. When I place an order with an on-line e-commerce web site, will confidential information get saved on a web cache?

There are certain cases where a web caching server will not cache any content under any circumstances. When a user opens a session with an e-commerce site with the intention of making an on-line purchase with their credit card, their browser invariably opens a secure encrypted session with the web server using the secure protocol HTTPS. It will usually indicate this by means of visual cue, or by prompting the user for approval. When this happens, all of the packets exchanged between the user's browser and the server use the protocol HTTPS rather than HTTP. Web caching servers do not cache HTTPS.

16. Do caching servers store copies of cookies?

HTTP is a stateless, connectionless protocol. Web browsers do not open connections with web servers, they send requests and receive responses. Every element of each web page is the object of an individual request from the browser and an individual response from the server. There is no link from one request to the next allowing the server to identify the client. Think of it



this way: a web server is like a mail order company that keeps no records. A browser sends it an order (HTTP request) asking for a web page. It fills the order (sends the web page) but retains no information about the customer (browser). The browser reads the page and discovers that it should be displayed with an image called "flowers.hawaii.gif" for example. So it sends another HTTP request for the image called "flowers.hawaii.gif". The web server obediently sends the photo but it has no idea that the request came from the same browser that asked for the page two seconds earlier. This process is repeated for as many items as there are items on the page. This type of protocol is sufficient for displaying basic text and graphics, but precludes any form of interactivity between client and server (i.e. the user and the web site). Cookies were introduced to solve this problem.

A cookie is a small piece of data that a web server can send to a web browser. Web browsers store cookies on the user's hard disk for as long as they are instructed to (cookies have expiry dates). Cookies contain information that can be sent back to the web server at a later time. In its simplest form, a cookie just has an ID number in it. Think of a cookie as a mail order customer ID. When a browser requests a page from a certain web site for the first time, the web server sends it a cookie. The next time the browser requests information, it sends the cookie back to the web server in much the same way that a mail order customer would quote their customer ID when placing an order. The web site is then able to identify the browser and return web pages containing customized information.

The cookie mechanism allows the web server to identify the user from one request to the next. A user may go shopping at an online bookstore, and put a number of books in their shopping basket. At the checkout the web server knows what the basket contains. It knows because it sent a cookie which was returned to the server by the browser with every request. This enabled the server to identify the user, and display pages containing the names of the books in the shopping basket. Cookies are also used by web servers to provide custom news pages showing selected stories to regular visitors who register with them. The principle is always the same. The cookie is like a customer ID that allows the web server to "know" who is visiting the site, and respond accordingly.

If a LAN based web caching server were to cache cookies for future delivery to multiple clients, this would make it impossible for the originating server to correctly identify users, as multiple clients would have identical copies of the same cookie. All users on the same LAN accessing the news site in the above example would be presented with the same news stories, and custom content would not be generated. Therefore under normal circumstances cookies are delivered end-to-end and not cached. Specific web caching servers however may allow administrators to override this default behavior.

17. Do web caching servers deprive some commercial portal web sites of hits and revenues?

This is an interesting question that arises occasionally and unfortunately is often subject to misinformation. We have seen claims alleging that caching is unfair and that it deprives some content providers of hit counts and revenue. It has to be recognized that caching is already in use throughout the internet and on multiple levels. It is here to stay and its use will increase. Whilst one should not try to justify doing something based on the argument that 'other are



already doing it', the fact that something is widely done and available must be taken into consideration.

Caching at various levels of the Internet improves the efficiency of the Internet generally and as a result improves the browsing experience for millions of users, providing them with faster access to data from host sites. The improved browsing experience results in increased page views which in turn will benefit content providers.

If a specific page from a content provider were cached on a LAN and subsequently served up from the cache when viewed by ten employees, then the number of people seeing it (and its associated advertising) would be the same as if the LAN cache were not present. The differences, from the perspective of a content provider, would be (a) that the content provider would not have as many 'hits' for the cached data as for individual accesses and (b) that cached advertising would show the same advertising to those viewing from the cache rather than rotating advertising.

Dealing with the subject of 'hits', it should be noted that web servers provide statistics for administrators to analyse. Information about the number of pages served, and the geographical location of international visitors are among the data that interest web site administrators. Such information may be used to forecast server load, for example, but may also be used to acquire a rough idea of the effectiveness of marketing and advertising.

Caching servers reduce server load so in this respect should be welcomed by web site administrators. If however the administrator is using simplistic statistics to measure the effectiveness of a marketing campaign for example, then it is possible for caching servers to distort the figures. Simplistic analysis should be replaced with advanced and relative analysis that will remove the statistical influence of caching. For example a twofold increase in traffic can be seen as such, but the actual number of visitors may not be known because of the existence of caching servers and proxies. Web site administrators know that caching exists, and factor this into their calculations.

Traditionally, those selling advertising in print media have charged for ad space based on an agreed sum per thousand impressions. (Rate Card based on Cost-per-thousand impressions- 'CPM'). As Internet advertising developed, and in the absence of an alternative methodology, it was natural for electronic publishers to transpose the 'cpm' business model to their Internet advertising. Since Web caching servers may cache some banner ads, then the statistics from the web server of the number of times a banner is displayed may be less than the number people who actually see the advertisement. The advertiser benefits from the visibility gained by the 'unpaid' additional "impressions" that the ad enjoys, but publisher will only be able to charge based on the impressions it can prove. It is this that has caused some concern to some web site owners. There are numerous ways for vendors of Internet advertising to address this. As we have mentioned before, if they use the "no-cache" directive, their ads from won't be cached. Another method, and one being increasingly favoured by advertisers, is charging on a per click-thru basis. Under such a scheme advertisers pay a fixed sum for each user that clicks on a banner ad and visits the advertiser's web site. This revenue model would be unaffected by web caching and would satisfy both the advertiser and the publisher.



In RFC 2227 the Network Working Group points out that publishers of print media know how many copies of a publication are circulated, but not how many people read each copy. They can only attempt to find out through the use of polls and marketing activities. Similarly, content providers can know how many times a page is served from the originating server but cannot be certain of the number of readers each page enjoys.

In summary then, caching is an integral part of all computing devices from CPU, to hard disk, to the World Wide Web, all of which depend on it for efficient flow of data. It is not reasonable to seek to reduce the effectiveness of the Internet architecture or its underlying mechanisms because one wishes to apply a transposed cpm business model (see also: <http://www.pbs.org/cringely/pulpit/pulpit19990923.html>). Ultimately good Internet user experience will benefit everyone including both publishers and advertisers.

18. Can web caching be used to pre-emptively download frequently viewed pages at off hours?

Some web browsers have a subscription function that allows users to automate downloads of certain web pages. The user simply indicates a frequency and a time of day for their preferred pages to be downloaded from the originating site. This subscription feature is generally favored by dial-up users wishing to browse their preferred site without constantly redialing their ISP. However, automated downloads can also be combined with a LAN based web caching server to reduce bandwidth usage at peak periods.

Consider a company with several people who read the same web pages on a daily basis. The administrator could program a browser to download the most popular pages daily at 5:00 AM, causing them to be cached by the LAN based web caching server. LAN users viewing the pages at peak traffic times later in the day would avoid using bandwidth to the Internet, experience improved response times and lower overall congestion for other users.

19. Conclusion

The view is commonly held among Internet architects that web caching is beneficial if not essential to network efficiency. As was mentioned in part one, growth of the Internet is outpacing Moore's Law. The complexity of a network increases roughly in proportion to the square of the number of nodes. The resulting exponential increase in complexity is multiplied by already exponential growth. Parkinson's Law is against us. No experimental, mathematical or theoretical model of Internet growth will allow us to be optimistic about bandwidth availability solving congestion problems. These problems can only be solved through more intelligent bandwidth and resource management strategies. Without web caching, such strategies would be crippled.