The
Internet Protocol Journal

September 1999 Volume 2, Number 3

A Quarterly Technical Publication for Internet and Intranet Professionals

In This Issue

You can download IPJ back issues and find subscription information at:
www.cisco.com/ipj

From The Editor

More and more of the data traffic on the Internet is due to World Wide Web activity. Given the often-complex graphics contents of Web pages, this traffic represents a significant amount of data and leads to an overall requirement for more bandwidth across the system. But building "bigger pipes" is not the only way to achieve better performance. Generally speaking, Web pages are relatively static objects that reside in *one* location and are accessed repeatedly by *many* users, often from "far away." If the contents of the most frequently accessed pages can be stored by a proxy residing more "local" with respect to the end user, significant reductions in download delay can be accomplished. Since the Internet comprises many expensive international circuits, such local mirroring of content is also highly desirable from the point of view of the Internet Service Providers. Storing information in a proxy server is called *caching*, and it is the subject of our first article. Geoff Huston explains the motivation behind—and the different approaches to—caching.

The most popular Local-Area Network (LAN) technology is *Ethernet*. Invented in 1973 by Bob Metcalfe as a 3-Mbps technology, Ethernet has evolved to the now-familiar 10Base-T and 100Base-T standards. Standardized in 1998, *Gigabit Ethernet* is the subject of our second article. Bill Stallings gives an overview of the Gigabit Ethernet standards and their application in enterprise networks. There is already discussion about 10-Gigabit Ethernet and even 100-Gigabit Ethernet. We will keep you posted on these developments.

Some readers have suggested that we publish a few short articles on limited topics. In this issue we bring you the first in what we hope will become a series of articles under the general heading "One Byte at a Time." The article is by Tom Thomas and he discusses *active* and *passive* modes of the File Transfer Protocol (FTP). If you have suggestions for future topics in this series, please contact us at **ipj@cisco.com**

The so-called "Millennium Bug" or "Y2K Problem" has been well reported in all the media. Our *Fragments* section gives some specific information relating to Y2K and the Internet.

—Ole J. Jacobsen, Editor and Publisher
ole@cisco.com

Web Caching

by Geoff Huston, Telstra

Web browsing dominates today's Internet. More than two-thirds of the traffic on the Internet today is generated by the Web. In looking at how to improve the quality of service delivered by the Internet, a very productive way to start is examining the performance of Web transactions. It is here that Web caching can play a valuable role in improving service quality for a large range of Internet users.

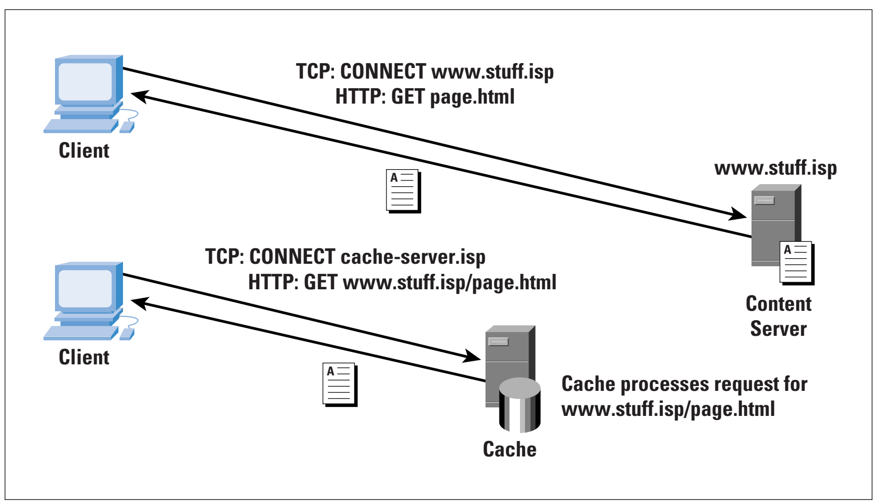
There are two types of Web caches—a *browser cache* and a *proxy cache*. A browser cache is part of all popular Web browsers. The browser keeps a local copy of all recently displayed pages, and when the user returns to one of these pages, the local copy is reused. By contrast, a proxy cache is a shared network device that can undertake Web transactions on behalf of a client, and, like the browser, the proxy cache stores the content. Subsequent requests for this content, by this or any other client of the cache, will trigger the cache to deliver the locally stored copy of the content, avoiding a repeat of the download from the original content source. In this article we look at proxy caches in further detail, particularly at the aspects of deployment of proxy caches in Internet Service Provider (ISP) networks.

What Is Proxy Web Caching?

When a browser wishes to retrieve a URL, it takes the host name component and translates that name to an IP address. A HTTP session is opened against that address, and the client requests the URL from the server.

When using a proxy cache, not much is altered in the transaction. The client opens a HTTP session with the proxy cache, and directs the URL request to the proxy cache instead (Figure 1).

Figure 1: A Proxy Web Transaction



If the cache contains the referenced URL it is checked for freshness by comparing with the "Expires:" date field of the content, if it exists, or by some locally defined freshness factor. Stale objects are revalidated with the server, and if the server revalidates the content, the object is re-marked as fresh. Fresh objects are delivered to the client as a *cache hit*.

If the cache does not have a local copy of the URL, or the object is stale, this is a *cache miss*. In this case the cache acts as an agent for the client, opens its own session to the server named in the URL, and attempts a direct transfer to the cache.

The Pros and Cons of End-to-End Web Access

The original design principle of the Internet architecture is that of the end-to-end model[2, 3]. Within this model the network is a passive instrument that undertakes a best effort to forward packets to the specified destination. Each packet generated by a host is assumed to be forwarded to the addressed destination, and any response to the datagram is assumed to come from that destination address.

The World Wide Web transaction protocol, the *Hypertext Transfer Protocol* (HTTP)[4, 5], is constructed upon this model, where a client's Web fetch causes a TCP session to be opened with the specified target host. The ensuing HTTP conversation identifies the requested data on the destination host, and this data is then passed back to the client. This delivery model is best expressed as a *just-in-time delivery model,* where the data is passed to the client on demand.

This delivery model has many significant advantages. The content server can modify the content, and all subsequent client requests are provided with the updated information, so that updates are immediately reflected in the delivered data. The content server is also able to track all content requests, allowing the content provider to track which particular content is being requested, the identity of each requestor, and how often each content item is referenced. The content provider can also differentiate between various clients, and, using some form of security model, the content provider can authenticate the client and deliver privileged information to certain clients. In this model the content provider can also differentiate between clients, delivering certain information to some clients, and *different* information to other clients of the content server.

Many web systems have been constructed based on the capability of this end-to-end delivery model. Continuously updating Web pages that use either *server push* or *client pull* to regularly update the content on the client's display are used to display stock market prices, weather maps, or network management screens. Client identification can be used to create combined public and virtual private information servers, where a class of identified users can be directed to internal content environments, while other clients are passed to a default public content environment. Such systems form the basis of extranet environments, and can also be used to form part of a virtual private network.

Where information has a defined locality, this tool is very useful. Security and authentication is also used to provide services where the transaction requires some level of privacy. Electronic trading systems, credit card transactions, and related financial systems on the Web make use of such client authentication capabilities. The individual transaction can be encrypted using socket-level encryption,[13] or the entire TCP session can be encrypted using an IP session-level encryption tool such as IP Security (IPSec).

For all these benefits available in an end-to-end model of Web content delivery, there are some balancing drawbacks. A server providing very popular content is placed under considerable stress, both in the number of simultaneous client connections active at any time and in the total volume of data being delivered from the server in the surrounding network. This load is expressed both as a server system load, and as load on the surrounding network. Improving the performance of such systems may entail improving the server throughout, increasing the number of servers through the use of server farms and a traffic manager, and improving the capacity of the local network to deliver the increased volume. However, all these measures may not address all the problems in maintaining quality of the content delivery. Modem-based client systems, and low-bandwidth wireless-based client systems are constrained by a combination of the restricted bandwidth of this last hop and the associated imposed end-to-end delay in conversing with the server. Improving the capacity of the server may not necessarily reduce the number of simultaneously active client connections. Reducing the delay between the client and the point of delivery of the content will improve the performance of content delivery.

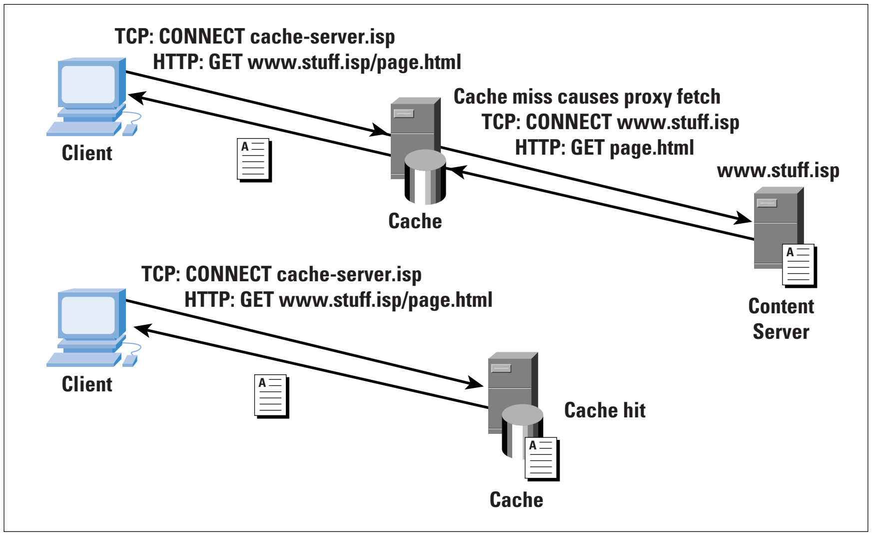
In addition, the network itself may not be efficiently utilized. Web traffic does have considerable levels of duplication, where a set of clients request copies of the same content, and the network carries duplicates of the data to each client. For a network provider, where transmission capacity is a business cost, importing the content just once, and then passing local copies of this content to each client, is one method of improving the carriage efficiency of the network.

In terms of the ability to improve the service performance of delivery of content to a global network of clients, and in terms of the ability to improve the carriage efficiency of the network, caching of content makes some sense to the content provider, to the ISP, and to the end client.

The Pros and Cons of Web Proxy Caching

The same benefits of improved performance and reduced outbound traffic loads can be realized for World Wide Web traffic through the deployment of Web caches. Web caches are basically no different from any other form of caching. The client request is passed through a *cache agent,* which makes the request to the original source as a proxy for the client. The response of the server is retained in a local cache, and a copy is passed to the client. If the same request is passed to the cache agent soon after the original request was serviced, the response can be generated from the cache without further reference to the original source. The operation of a Web cache is shown in Figure 2.

Figure 2: A Web Cache



Measurements of ISP traffic profiles indicate that some 70 percent of a typical ISP's traffic is Web-based traffic. An analysis of Web requests indicates that the typical level of similarity of requests (for the same object as one previously requested) can be as high as 50 percent of all Web-based traffic.

There are two hit-rate measures, a *page hit rate* and a *byte hit rate*. A page hit rate measures the proportion of individual HTTP requests that can be served from the cache, irrespective of the size of the page. A byte hit rate measures the ratio of the number of bytes delivered from the cache in hits against the number of bytes in misses. Experience to date has indicated that page hit rates of somewhere between 40 to 55 percent are achievable for a well-configured cache. In such circumstances the associated byte hit rate is between 20 and 35 percent. The major contributor to the hit rate is in image files.

For many ISPs, particularly those operating outside of North America, transmission costs dominate the cost profile of the ISP's operation. If the cache performed at even 60 percent of a theoretical maximum caching performance, the ISP could reduce its external traffic volume requirements by some 13 percent. When the costs of caching are compared to the costs of transmission, this difference can be a significant one in the cost base of the ISP's operation.

For example, if the average cost of transmission is $150 per gigabyte, and the ISP has a typical carriage profile of purchasing 1000 gigabytes per month from an upstream ISP with a 70-percent Web traffic profile, then a cache operating at a 25-percent byte hit rate can save the ISP a recurrent expenditure of $26,250 per month. If the cache costs $100,000 as a capital expenditure and $2000 per month in operational costs to support the service, then a business case analysis would see the cache activity return some $18,000 per month to the business, net of annualized capital and operational expenditures.

The other benefit is to the client, where the reduced network delay between the client and the local cache results in an increase in speed of Web page delivery for cached content.

The average size of a Web transaction is some 16 data packets within the TCP flow. Within a TCP slow-start flow-control process, the first cycle will transmit one packet and wait for an ACK. The reception of the ACK will trigger transmission of two more packets in the second round-trip cycle, and then the sender will await two ACKs. Reception of these two ACKs will trigger a further four packets in the third cycle and eight in the next cycle, and the remaining single packet in the fifth cycle. Therefore, allowing for optimal behaviour of the TCP slow-start algorithm, this average Web transaction takes some five round-trip times. If a user is located some distance away from the Web page, and the round-trip time to the source is 300 ms, the propagation delay of the page load will be 1.5 seconds. In comparison, if the round-trip time to the local Web cache is 2 ms, then the propagation delay of the page load will be 10 ms. These latency figures assume an uncongested network in both cases. In this case, as long as the Web cache search can complete within 1 second, the cache will appear to be far faster to the user.

A slightly different analysis is possible when comparing the performance of a cache configured at the headend of a cable-IP system versus the performance of direct access. The difference in latency in this case is due to both the closer positioning of the cache to the user and the greatly increased effective bandwidth from the cache to the user. A cache download can operate at speeds of megabits per second, as compared to kilobits or tens of kilobits per second when using dialup modem or ISDN services. For a 100K image download, the dial user may experience a 60-second delay, and the same delivery from a local cache via cable-IP may take less than half a second.

The trade-off with caching is that of balancing the the cost of carriage capacity, both in terms of monetary cost of the carriage and the performance cost of the transaction time of the application, against the cost of the use of caching. For non-North American ISPs, in which there is typically a large cache hit rate against North American server locations, the benefits of widespread use of caching are quite substantial. For cable-IP operators, the benefits of local cache operation lie in the ability to exploit the benefits of the very-high-speed final hop from the headend to the end user. For other ISPs, the benefits of caching may be less dramatic, but nevertheless, there are tangible positive outcomes of caching in terms of performance and cost that can be exploited.

As with direct-access models, this approach also has drawbacks. We have already noted the various ways in which the end-to-end model of Web content delivery has been exploited to provide time-based content, client-based content, and secure delivery of content. Caches insert themselves within the end-to-end semantics of the original transaction model, and intercept the transaction by presenting a proxy of the original endpoint. The content delivered from the cache is the content based on the time the cache undertook its request to the server, and the content delivered from the server is based on the server's view of the identity of the cache, rather than the identity of the end client.

With cached content in operation, the cached-content server no longer has an accurate picture of the number of times an item of content is viewed, and by whom. The server cannot authenticate the client, nor can the server deliver any information that is based on the supposed identity of the client. Equally, the client has potential problems, because the client may not be aware that the content has been delivered by the proxy cache. The content may not properly reflect the client's identity, and the information may be based on the security trust model of the server to the cache, rather than the server to the end client, and again the client may not be aware of such a change in security domains. If the content is time-dependent, the content will reflect the time at which the cache retrieved the content, rather than the time the client made the request.

All of this tends to suggest that caching is not a universally applicable tool. Part of the challenge in deploying cache servers is to understand the models of cache deployment and Web content delivery, and ensure that the cache does not intrude in ways that distort the integrity of content delivered to the end user.

Web Cache Hits Versus Web Server Hits

One of the biggest tensions is the balance between the cache operator's desire to maximize the hit rate of the cache system and the desire of many Web page publishers to maintain an accurate count on the number of hits of the page and from where those hits occur. In most cases, it is the requests that are of interest here, rather than the control of delivery of the content. The Web publisher is not necessarily interested in absorbing the hits for Web content. Indeed, many Web publishers see value in distributing the load of content delivery of fixed-content material further out toward the client base, rather than the Web publisher bearing the cost of the distribution load from the local site.

Static pages, composed of plain text and images, are readily cached. As a consequence, the original page publisher may not obtain an accurate count of the number of times the page was displayed by users if the Web server's log was analysed. Some Web page designers place information in the Web page directives; this information directs the Web cache server not to reuse a cached page. The most common way of doing this is to set the "Expires:" Web page information header to the current date and time, so the next time the page is referenced, a new fetch will be undertaken. One of the more common hacks to cache servers to attempt to improve the hit rate is to allow this directive to be ignored.

This server hit-count problem has plagued cache deployment for many years now. Although there are real requirements in the areas of authentication and security, time-based content, and client-based content that mandate certain types of content being flagged as non-cacheable, much of the data that is marked as non-cacheable has been marked in this way simply for the server to capture the identity of the client. Such "cache-busting" practices are unnecessarily wasteful of network resources, and can overload the content server. There is an Internet Proposed Standard extension to HTTP[6] intended to provide a "Meter" header, where a cache can communicate demographic information relating to client "hits" back to the original content server. The extension also proposes usage limiting, where a server can provide content with a limit on the number of times the information can be used by the proxy cache before revalidating the content with the server.

Web-Caching Models

There are many models of how to invoke a proxy cache.

Explicit Caching

Some proxy cache systems are deployed as a user-invoked option, in which the user nominates a cache server to the browser as a proxy agent, and the browser then directs all Web requests to the proxy cache. At any stage, the user can instruct the browser to turn off the use of the proxy cache, and request the browser to undertake the transaction directly with the client. Modern browsers when configured with a proxy cache may also use the approach of attempting direct access when a request via a proxy cache results in a fetch error. In the proxy cache mode of operation, the destination address of the underlying transport session is then the address of the cache server, while the HTTP content of the transaction remains unaltered. Such caches can be deployed within a client's local network, with the intent of minimizing the amount of traffic passed to the external provider ISP. Additionally, The ISP can operate such a voluntary cache for use by its clients. If the ISP operates in this mode, the benefits to the user in using the cache need to be clearly stated and understood by both the client and the ISP, and the client must be made aware of the location of the cache in configuring his or her local browser.

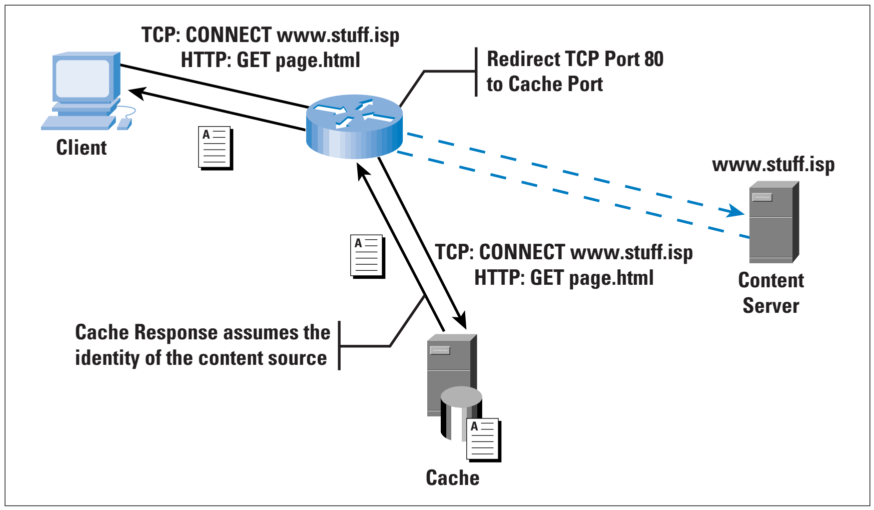
Forced Explicit Caching

Some ISPs, notably in the dialup service provider sector, operate in a highly cost-competitive market. In such a market service performance and service price are critical business factors, and the provider may choose to operate its network in a forced-cache mode. Here, all Web traffic on TCP port 80 (the port used by the HTTP Web transport protocol) is blocked from direct outbound access, and the ISP's clients are forced to configure their browsers to use the provider's cache for external Web access. This technique is commonly termed *forced caching*.

Transparent Caching

The use of a cache for all Web traffic also can be undertaken by the ISP, without the explicit configuration of the identity of the proxy cache into the user's browser. Irrespective of precisely how this setup is engineered, and there are numerous ways of engineering it, this technique is termed *transparent caching*. With transparent caching the user, and the user's browser, may not be explicitly aware that caching is being undertaken when processing the user's requests. Here the network has to intercept HTTP packets destined to remote Web servers, and present these packets to the proxy cache. Once the page is located, either as a cache hit or a cache miss, the cache must then respond to the original requestor by assuming the identity of the original destination (Figure 3).

Figure 3: Transparent Caching



It should be noted that no mechanism to date of explicit or transparent caching is completely transparent to both the Web client and the Web server. Where the Web server uses an end-to-end security access model the transparent cache may fail, because the cache will present its address as the source of the request, rather than that of a client. This scenario may result in a page-denied error to the cache request, whereas the client could have completed the transaction directly with the server. In those situations where the use of the cache is mandated, either through filters and a forcing function, or through transparent network redirection, there is no user-visible workaround to the error, and the level of user frustration with the entire cache service rises dramatically.

Under some circumstances it may be possible to work around transparent cache fetch errors. One approach is for a cache fetch error to trigger the cache subsystem to establish an HTTP session with the content server using the source address of the client, and then pass the original HTTP GET request to the server. The server's response is then passed to the client using a TCP bridge. (A TCP bridge is where the connecting device is required to translate the sequence numbers of the TCP headers between the two TCP sessions). Having the cache subsystem intercept the server's packets addressed to the client does require careful coordination with the cache router, and TCP bridging is also quite complex in its operation, so such solutions tend to be somewhat unstable under load stress. An alternative approach is for the cache to pass a TCP RST back to the client, and instruct the cache router to insert a temporary entry in its redirection filter so that any subsequent TCP port 80 connection from the client to the server's address is not redirected to the cache.

If the sole benefit to the client is improved speed of response, then the ISP must understand that the performance of the Web cache systems must be continually tuned to be highly responsive to Web requests under all load conditions experienced by the ISP. Performance of cache hits must be maintained at a level consistently faster than the alternative of direct client access to the original client site. Performance of cache misses must be at a level that is not visibly slower than that of direct access to the original site. If the user's perception of performance of the cache drops, the benefit to the user also drops. In the case of user-selected caching, the users will turn off the cache option in their browser and return to a mode of direct access.

The business model of a cache is that the capital and operational costs associated with localizing traffic to the cache result in cost reductions to the ISP, when compared to the operation of a noncached network. These cost reductions can be passed on to all users through operation of the entire service at a lower price point or selectively passed on to those clients who make use of the cache through some form of cache-use tariff. The generic model of applying the cost reduction to the ISP's service tariff is certainly an advantage in a price-competitive marketplace. However, unless the performance of the cache is consistently very high, and the transparency of the cache is close to perfect, each individual user may attempt to use direct-access methods.

The alternate business model is to pass on the marginal cost savings to those clients who make use of the cache, and at a level that corresponds to the client's use of the cache and its effectiveness in operating at a high cache hit rate. If, for example, the ISP uses a charging model that includes a tariff component based on the amount of data delivered to the client during the accounting period, this tariff component could be adjusted by the amount of use the client made of the cache system and the relative operating efficiency of the cache in generating cache hits.

As an example, if traffic is tariffed at $100 per gigabyte as delivered to the customer, a discounted value can be derived for traffic delivered from the Web cache. If the average cache byte hit rate is 30 percent, then after factoring in the costs of capital equipment and operational support, the traffic from the cache could be tariffed at $80 per gigabyte. Here, the benefit of using the Web cache is passed directly to those clients who make use of the cache, who both enjoy lower tariffs in direct proportion to their use of the cache and derive superior performance through using the cache. The accounting for this marketing model is certainly a more involved process, involving additional accounting systems and processing to undertake an accurate per-client view of cache usage.

It is becoming increasingly evident that a robust business model associated with a model of discretionary use of a Web cache is that of access to a lower unit price of traffic. In this way, the user sees the incentive of immediate financial benefit in choosing to use the cache system. When the provider deploys transparent or forced caching, translating the benefits of caching into an overall reduced tariff structure for all clients is a more robust business model.

Web-Cache Systems

Cache systems can take a variety of forms. The original Web server from CERN, the original location of the development of Web software, allowed a mode of proxy behaviour. This cache server model was developed significantly in the Harvest Project, a research project at the University of Colorado. As an evolutionary path, the *Harvest* cache server is being further developed within the scope of the development of the *Squid* cache server software and the associated *Internet Caching Protocol* (ICP).

Currently numerous freely available proxy cache systems are available, such as Squid, and many systems are available commercially, such as the Cisco Systems *Cache Engine*. Some of these systems are software packages that operate on a conventional operating system platform, while some use a customized platform kernel, which is optimized for the demands of a cache-delivery environment.

Many of the characteristics of Web caching systems are relevant to the performance of the caching environment. The first is the *size* of the cache server. The relationship between the size of the cache and its hit rate is not a linear relationship. For typical patterns of Web use generated from a relatively large user population, a cache of 1 gigabyte or so will yield reasonable hit rates. Further increase of the cache size will yield incremental improvements in the cache hit rate, where the incremental rate is best described by a negative exponential relationship. Thus, caching systems with 10 gigabytes of storage do not produce performance characteristics markedly different from larger 100-gigabyte caching systems. No objective best size of cache system can be determined, because local environments differ, but every environment exhibits the law of diminishing returns, in which the addition of further cache capacity yields no tangible difference in the cache effectiveness. Large caches take some time, in the order of days or even weeks, to build up a sufficiently large repository of cached data to produce an improved cache hit rate. Generally, 10- to 100-gigabyte cache systems provide extremely effective cache performance, as long as the cache is allowed to stabilize for some weeks following startup. Memory demands in a cache also need to be carefully configured. The URL index of the storage system is stored in memory in most cache architectures in order to perform fast cache lookups, so that the more disk storage configured, the larger the memory requirements.

The next parameter is the *number of simultaneous cache requests* that the cache server can manage efficiently. Note that this metric is different from the number of requests per second that the cache server can manage. The number of simultaneous sessions that the cache server can support is related to the amount of resources allocated to the cache request and the total resource capacity of the box.

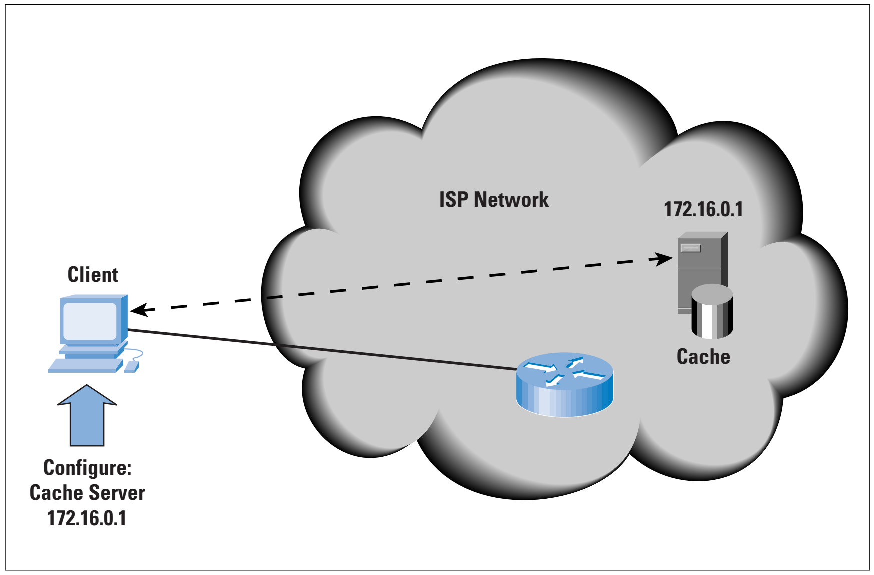
The environment of deployment is very relevant to the performance of the cache environment. The related metric to the number of simultaneous requests that can be managed is the average time to process a request. Combining these two metrics provides the number of requests per second that the cache system can process. The same unit will have a different performance metric of requests per second when deployed in different parts of the Internet. If the cache system is deployed with a satellite-based feed, then the average time to process a cache miss is considerably longer because of the higher latency of the satellite path. This scenario leaves the process of managing the original request open for a longer period, blocking other requests from using this process slot. If the same unit is deployed in a location where cache misses take fractions of a second to process, the process slot can be quickly reused. Each active client connection also consumes memory, and the client connection will remain open for as long as it takes to complete the Web transaction, either for a hit or a miss. The greater the mean round-trip delay for a miss, the greater the number of concurrent active sessions held in the cache. Similarly, the greater the number of low-speed modem or wireless-based clients, the greater the number of concurrent active sessions in the cache. Whether the client operates in transparent mode or in explicit proxy caching mode is also an important consideration. Browser clients use an explicit proxy cache with a persistent connection, while if the cache is a transparent cache, the cache will see clients bring up and drop HTTP connections each time the base URL changes. This session reestablishment, together with the additional Domain Name System (DNS) resolution load imposed on the client, can add up to half a second to the transparent cache response time as compared to the explicit cache response.

Web Cache Deployment Models

In this section we first examine scaling issues for explicitly referenced cache configurations, and then look at the changes to the model introduced through transparent caching.

The simplest deployment model of an explicit cache is that of deployment of a single cache system as a browser-selectable resource. This system can be deployed within an ISP's server environment with a TCP port-80 interface opened for client access. Such a deployment model is shown in Figure 4.

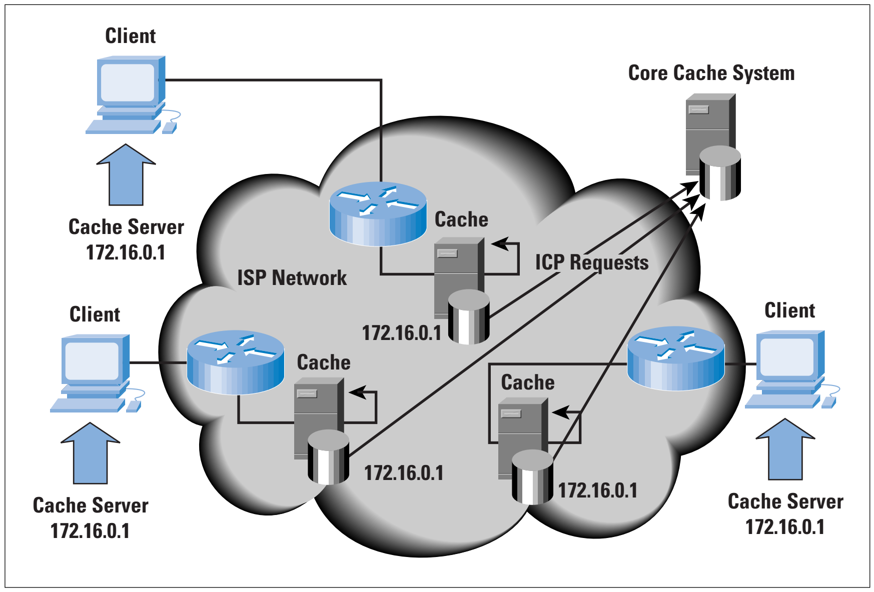
Figure 4: A Selectable Web Cache



Single Web proxy cache systems can be placed under some significant load, and an overloaded and poorly performing cache is perhaps worse than no cache at all. However, scaling this deployment model can prove challenging. Where an ISP operates multiple access points, or points of presence (POPs), one scaling solution is to deploy a server at each POP and use the same IP address for each server. This solution allows the ISP to provide a consistent configuration to all clients and to augment capacity at any location seamlessly. If the cache itself is responsible for advertising the common IP address into the routing system, the caches can also act in a mutual backup role. Failure of a single server will shut down the local route advertisement. Traffic directed to this address will then be carried by the routing system to the next closest proxy cache. There may be some level of TCP session resets for sessions that were active on the failed unit, but in all other respects the switchover is seamless to the client base, and the recovery of an operational state among a set of such servers can be left to the routing system. This deployment model is indicated in Figure 5. Such servers can be configured as a set of local satellite systems to a larger caching core, using an *Internet Cache Protocol* (ICP) configuration to set up a caching hierarchy.

ICP is a lightweight message format for communicating between Web caches[7]. The message format is a simple two- packet exchange, where a Web cache passes a URL query to another cache. The response is either a hit or a miss, indicating the presence of the URL object on the remote cache. On top of this protocol can be constructed cache hierarchies, to allow multiple neighboring caches to pool their resources effectively.

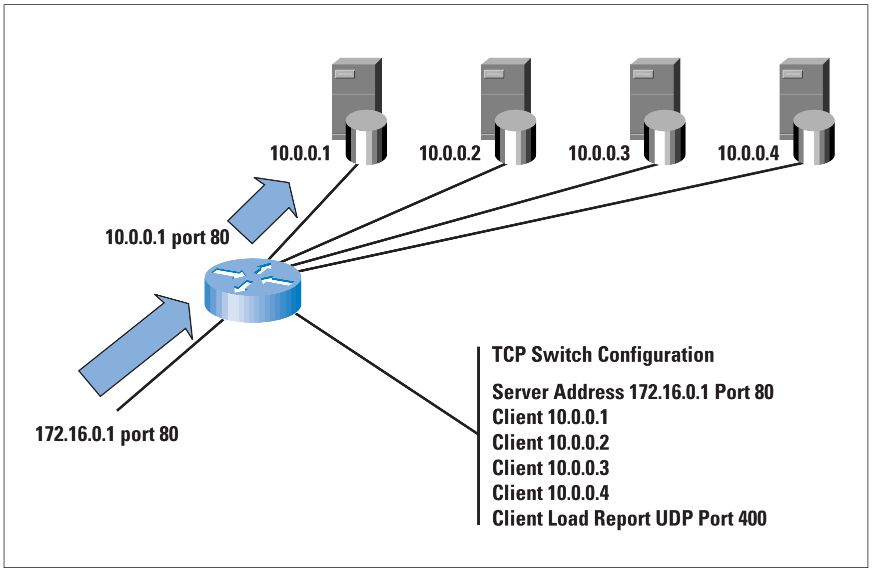
Figure 5: Replicated Web Caches



The proposed mode of configuration of caches is into a tree-structured hierarchy[8]. In such a hierarchy every participating cache is organized with a connection of neighboring peers and an ICP parent. When a cache request cannot be serviced from the local cache, the cache first uses a set of local configuration rules to determine if the server is local. If so, the cache queries the server directly for the content. If the server is remote, the cache issues a set of simultaneous ICP queries to all its cache peers. If any peer responds with an ICP hit, the cache then requests the peer to provide the referenced content. If all peers respond negatively to the ICP query, or a two-second timeout elapses, the cache then requests the URL from its designated parent. The parent may use a peer referral, or the parent may refer the query to its parent, or perform a cache retrieval on behalf of the original request. The intent of this mode of operation is to use a lightweight query response protocol to allow a local collection of caches to pool their cached data. ICP has also been used with additional policy constraints, although the protocol itself is not capable of describing or carrying overly complex retrieval policies. Other intercache protocols are available, including the *Hyper Text Caching Protocol* (HTCP) and the *Cache Array Routing Protocol* (CARP), which offer functionality in terms of intercache cooperation similar to that of ICP[9].

Another scaling measure is to alter the single server to multiple servers, using a TCP-based, load-sharing mechanism in the switching system to ensure that the servers are evenly loaded. This setup is shown in Figure 6. Such a simple load-sharing system may even the load on each server, but it will cause each server to act independently of its sibling servers. It is essential in such an environment to use ICP to coordinate the servers so that they will refer to each other before initiating a new fetch from the content server.

Figure 6: Load-Balancing Web Caches



In such a configuration each cache will contain content also held in neighboring caches. Although this scenario may allow some form of server load balancing, particularly when the servers continually communicate their current load conditions to the load-balancing switch, there is still some inefficiency in the cache farm operation through the potential replication of content on each of the component caches. One direction of scaling the cache servers is to take a collection of cache servers and allow each cache server to specialize in the content it holds. However, the outer TCP destination address does not help the server determine which URL is being requested. In an explicit cache configuration, the browser is directing the TCP session to the externally advertised TCP address of the server farm. The URL information is embedded within the HTTP payload. Some developments have been made in this area, where, with a combination of TCP spoofing and TCP session bridging, a server switch can select the appropriate cache for each HTTP-referenced URL, and then logically connect the client's TCP session to a TCP session to the selected cache to deliver the URL to the client.

Transparent caching presents some further deployment challenges. The functional requirement is to pass all Web requests through a proxy cache server without the explicit knowledge of the client. Two generic techniques exist to achieve this goal:

- *Inline caches:* The first of these approaches is to pass all traffic through a two-port cache server. All non-HTTP traffic is simply passed straight through the device without alteration. HTTP traffic is intercepted and passed to a cache module. The major concern with this approach is the introduction of a single point of failure with an active network element. Any failure of the cache may well prevent all further traffic from entering or leaving the served subnetwork.

- *Redirection caches:* A technique that does not place the cache as a critical point of potential failure is to use policy redirection within the router, redirecting all port-80 traffic to the attached cache. Normally such a policy redirect would infer that the cache is located one hop away from the router, so that such a redirection is normally a local solution. Redirection to a tunnelled interface does allow some greater flexibility in this setup, and the one cache farm could, in such an approach, service a collection of redirecting routers. The failure mode of this form of operation remains a concern, because the redirection mechanism in the router would not normally be aware of the operational status of the cache.

Transparent caches need to ensure that the full URL is inserted into the HTTP level request. When the browser assumes that the request is directed to the content server, the GET request may specify a URL relative to the server. In such cases, the transparent server will need to perform a DNS lookup of the destination IP address of the TCP session in order to reconstruct the complete URL.

Although the DNS lookup does have some performance implications to transparent caches, the major issue for transparent caches is to devise a fail-safe mechanism, so that if the cache server fails for any reason, the caching redirection is disabled. One solution is to use a redirection function within the router in conjunction with a keepalive-based Web cache management protocol. This scenario is the basis of the *Web Cache Coordination Protocol* (WCCP)[10]. WCCP also adds the ability to load share across multiple cache servers through content distribution. Transparent caching assists in this task because the destination address in the IP packets can be used as the basis of the cache selector. The keepalive exchange between the router and the cache server system allows the router to cease redirecting Web traffic upon failure of the servers.

Alternative solutions rely on the cache itself participating in a local routing environment. The redirecting router uses policy-based redirection to forward all port-80 traffic to an address announced by the cache system at a high routing priority. The same address is also announced by the default path router at a low routing priority. Failure of the cache system will result in a withdrawal of the high-priority route, and while the redirection will remain in place on the router, the redirection will be in the direction of the default route.

Another challenge is to process cache misses at a speed comparable with normal noncached Web retrieval. A process of pulling the document into the cache and then serving the document to the original requestor does not meet that objective. The transparent cache has to feed the document to the requestor while simultaneously creating a stored copy for subsequent cache serving.

However, the largest challenge to the transparent cache is that it can serve only documents that are not dependent on the identity of the requestor being preserved. Web servers that use an end-to-end model of access, based on source address identification, or Web servers that attempt to present different documents to the client based on the client's source address, do not fit within the transparent caching model. There is much interest in solutions that allow a transparent cache to effectively shut down in the case of a Web retrieval error, and allow the original requestor the ability to conduct a HTTP conversation directly with the server in such situations. Although there is interest in a network-only solution, it appears at this stage that some level of assistance from the browser may be required. One model of operation is that a transparent cache records the network-level flow identification of a failed Web retrieval, and passes a retry signal back to the requesting browser, and also passes this flow identifier back to the redirector as a temporary filter entry. When the requestor retries the query, as per the signal from the cache, the redirecting router will refrain from redirecting the flow to the cache, and allow an end-to-end session to operate.

Accounting for Web Cache Use

These deployment systems allow for user-optional cache configuration. If the ISP wants to account for the use of the cache, then the cache server or the switch that feeds the cache server must play an active role in accounting collection.

If every network address is uniquely advertised to the ISP by a particular client, then the task of accounting for cache use can be performed using the logged records of the cache system itself. Because every IP address can be uniquely mapped to an ISP client, it is possible to also associate the volume of bytes delivered by the cache to the identified client.

Unfortunately, two factors make this supposition of address uniqueness somewhat weak. First, dialup address assignment implies that the association of an IP address to a client is held only within dialup accounting records in the first instance, and the binding is valid only between the times referenced in the start and stop records. This scenario can be configured into an accounting model by simultaneously processing the dial accounting records when attempting to associate a particular IP address at a particular time to a client.

The second factor is slightly more challenging. For an ISP that offers permanent access transit services, the potential exists that any particular IP address may not be uniquely routed. Normally, such multiple access environments are part of a Border Gateway Protocol (BGP)-based interaction with multiple clients. Knowing the IP address of the query agent is not enough. Ascertaining the next-hop Autonomous System (AS) number as well as the IP address is now necessary to determine the client using the cache.

The implication is that the accounting records now need to be generated on the router that is also the entry point to the cache. In addition, the router must participate in the interior BGP (iBGP) core mesh to maintain current AS path-selection choices. Given the considerable overheads that such an engineering design entails, an alternative approach is to restrict the cache accounting role to account for those cases where the cache client is readily identified. A common measure is that the lower tariff is available only to customers who are "singly homed" with the ISP. Not only is this a strong market incentive for customer loyalty, it also allows simple engineering solutions for cache accounting, because the lookup from the IP address in the cache log to a customer account is then relatively straightforward. Such measures allow a cache-use tariff to be very competitively positioned in the market.

As well as accounting issues, another component for the consideration of optional use of Web caches is that of the necessity of restricting the use of the cache to clients of the ISP. The motive for so doing is to ensure that the cache is available only to clients of the service and not to clients of peer ISPs. It may not be an issue worth the effort of solving, and the first questions ISPs should ask is, "To what extent does this happen, and what impact does it have on the operation of the Web cache systems?" In most cases, the accounting of cache usage may reveal that this issue is one of negligible proportions, and any effort expended in devising an engineering solution would far outweigh the loss to the ISP through such use of the service.

If the measurement of such usage is considered sufficient to warrant engineering solutions, then the mechanisms available to the ISP are to ensure that the Web cache access is filtered at the edges of the ISP network and to ensure that access is possible only by ISP clients, or that the address of the cache is not exported in the routing system to peer ISPs or upstream ISPs.

Further Deployment Challenges

It is highly likely that further development will occur with cache servers in the near future. Large-scale backbone IP networks that use OC-3c (155 Mbps) or OC-12c (622 Mbps) transport cores may carry tens of thousands of requests per second. Designing transparent caches that fit within a transport core at such a scale does present dramatic scaling issues in terms of cache system performance. This factor continues to elude many of today's products available on the market. The generic architecture today is to use a cache network that attempts to place the cache systems closer to the access edge of the network, where the Web request volumes are within the scale of today's cache systems.

Transparency of the cache remains an issue, and it is perhaps an area of further refinement within the specification of the underlying HTTP Web server protocol, as well as further refinement of the operation of Web browsers and transparent cache systems. A potential implementation within Web browsers may allow the user to state the acceptability of using a cache to complete a request, and allow noncache Web page retrieval attempts on cache failure, in the same way that the provider can use page expiration directives to direct a cache not to store the presented data.

References and Further Reading

[1] Huston, G. *ISP Survival Guide,* ISBN 0-471-31499-4, John Wiley & Sons, November 1998.
A more comprehensive view of the technology, business, and strategy behind the Internet service sector.

[2] Clark, D.D. "The Design Philosophy of the DARPA Internet Protocols," Proceedings of SIGCOMM 88, ACM *Computer Communications Review* (CCR), Volume 18, Number 4, August 1988, pp. 106–114 (reprinted in ACM CCR Volume 25, Number 1, January 1995, pp. 102–111).
The original paper describing the end-to-end design philosophy used within the Internet protocols.

[3] Carpenter, B., Ed. "Architectural Principles of the Internet," RFC 1958, Informational RFC, June 1996.
A summary of the design principles underlying the current Internet architecture.

[4] Berners-Lee, T., et al. "Hypertext Transfer Protocol—HTTP/1.0," RFC 1945, Informational RFC, May 1996.
The specification of Version 1.0 of the HTTP protocol.

[5] Fielding, R., et al. "Hypertext Transfer Protocol—HTTP/1.1," RFC 2616, Draft Standard RFC, June 1999.
The specification of Version 1.1 of the HTTP protocol.

[6] Mogul, J., and Leach, P. "Simple Hit-Metering and Usage-Limiting for HTTP," RFC 2227, Proposed Standard RFC, October 1997.
A proposed extension to HTTP to allow a content server to receive hit reports from a proxy cache.

[7] Wessels, D., and Claffey, K. "Internet Cache Protocol (ICP), Version 2," RFC 2186, Informational RFC, September 1997.
A description of the ICP protocol.

[8] Wessels, D., and Claffey, K. "Application of Internet Cache Protocol (ICP), Version 2," RFC 2187, Informational RFC, September 1997.
A description of the structure of cache hierarchies, and their ICP-based interaction.

[9] Melve, I. "Inter Cache Communications Protocols," Internet Draft, Work in progress, **`draft-melve-intercache-comproto-00.txt`**, November 1998.
An overview of intercache communications protocols currently available, and a collection of references that describe these protocols in further detail.

[10] Cieslak, M., and Foster, D. "Web Cache Coordination Protocol V1.0," Internet Draft, Work in progress, **`draft-ietf-wrec-web-pro-00.txt,`** June 1999.
A description of Version 1 of the WCCP protocol to support the operation of transparent caches. The protocol defines the interaction between a router and a neighboring cache system.

[11] "Squid Internet Object Cache"—Resource Web page.
`http://squid.nlanr.net`
A very useful page of resources and references related to the Squid implementation of Web caching.

[12] "Distribution of Stored Information on the Web," Online Tutorial, Ross, K., Institut Eurecom, October 1998. Available at:
`http://www.eurecom.fr/~ross/CacheTutorial/DistTutorial.html`
A good overview of proxy caching technologies, and also a good analysis of their efficiency of operation.

[13] Stallings, W. "SSL: Foundation for Web Security," *The Internet Protocol Journal*, Volume 1, Number 1, June 1998.

[This article is based in part on material in The ISP Survival Guide, by Geoff Huston, ISBN 0-471-31499-4, published by Wiley in 1998[1]. Used with permission].

GEOFF HUSTON holds a B.Sc and a M.Sc from the Australian National University. Closely involved with the development of the Internet for the past decade, particularly within Australia, he was responsible for the initial build of the Internet within the Australian academic and research sector. Huston is currently the Chief Technologist in the Internet area for Telstra. He is also an active member of the IETF, and is the chair of the Internet Society Board of Trustees. He is author of *The ISP Survival Guide*, ISBN 0-471-31499-4, and coauthor of *Quality of Service: Delivering QoS on the Internet and in Corporate Networks*, ISBN 0-471-24358-2, a collaboration with Paul Ferguson. Both books are published by John Wiley & Sons, E-mail: **`gih@telstra.net`**

Gigabit Ethernet

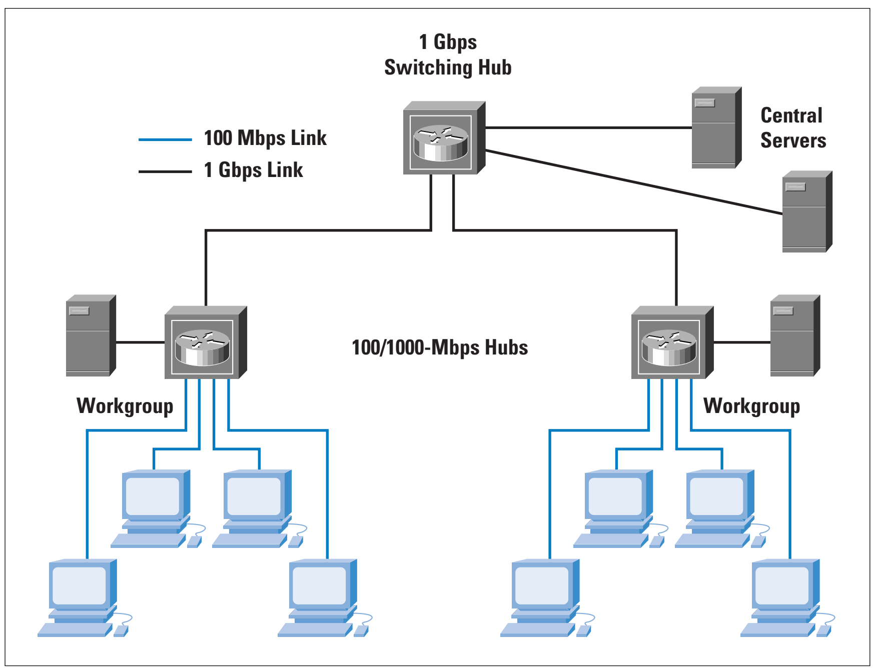
by William Stallings

In late 1995, the IEEE 802.3 committee formed a High-Speed Study Group to investigate means for conveying packets in Ethernet format at speeds in the gigabit-per-second range. A set of 1000-Mbps standards have now been issued.

The strategy for Gigabit Ethernet is the same as that for 100-Mbps Ethernet. While defining a new medium and transmission specification, Gigabit Ethernet retains the carrier sense multiple access collision detect (CSMA/CD) protocol and frame format of its 10- and 100-Mbps predecessors. So it is compatible with the slower Ethernets, providing a smooth migration path. As more organizations move to 100-Mbps Ethernet, putting huge traffic loads on backbone networks, demand for Gigabit Ethernet is intensifying.

Figure 1 shows a typical application of Gigabit Ethernet. A 1-Gbps LAN switch provides backbone connectivity for central servers and high-speed workgroup switches. Each workgroup LAN switch supports both 1-Gbps links, to connect to the backbone LAN switch and to support high-performance workgroup servers, and 100-Mbps links, to support high-performance workstations, servers, and 100-Mbps LAN switches.

Figure 1: Example Gigabit Ethernet Configuration



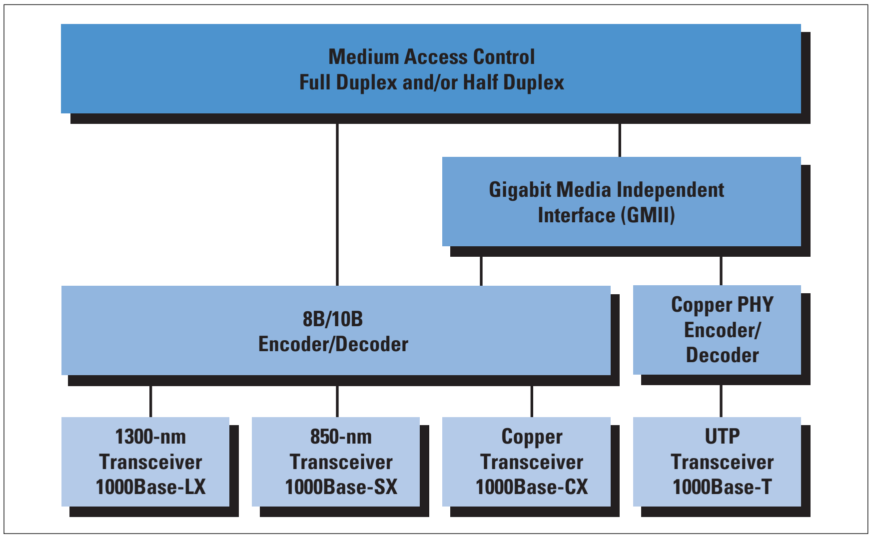
Protocol Architecture

Figure 2 shows the overall protocol architecture for Gigabit Ethernet. The Media Access Control (MAC) layer is an enhanced version of the basic 802.3 MAC algorithm. A separate gigabit medium-independent interface (GMII) has been defined and is optional for all the medium options except unshielded twisted-pair (UTP).

The GMII defines independent 8-bit-parallel transmit and receive synchronous data interfaces. It is intended as a chip-to-chip interface that lets system vendors mix MAC and physical sublayer (PHY) components from different manufacturers.

Two signal encoding schemes are defined at the physical layer. The 8B/10B scheme is used for optical fiber and shielded copper media, and the pulse amplitude modulation (PAM)-5 is used for UTP.

Figure 2: Gigabit Ethernet Layers



Media Access Layer

The 1000-Mbps specification calls for the same CSMA/CD frame format and MAC protocol as used in the 10- and 100-Mbps versions of IEEE 802.3. For traditional Ethernet hub operation, in which only one station can transmit at a time (half-duplex), the basic CSMA/CD scheme has two enhancements:

- *Carrier extension:* Carrier extension appends a set of special symbols to the end of short MAC frames so that the resulting block is at least 4096 bit-times in duration, up from the minimum 512 bit-times imposed at 10 and 100 Mbps. This extension makes the frame length of a transmission longer than the propagation time at 1 Gbps.
- *Frame bursting:* This feature allows for multiple short frames to be transmitted consecutively, up to a limit, without relinquishing control for CSMA/CD between frames. Frame bursting avoids the overhead of carrier extension when a single station has a number of small frames ready to send. extension when a single station has numerous small frames ready to send.

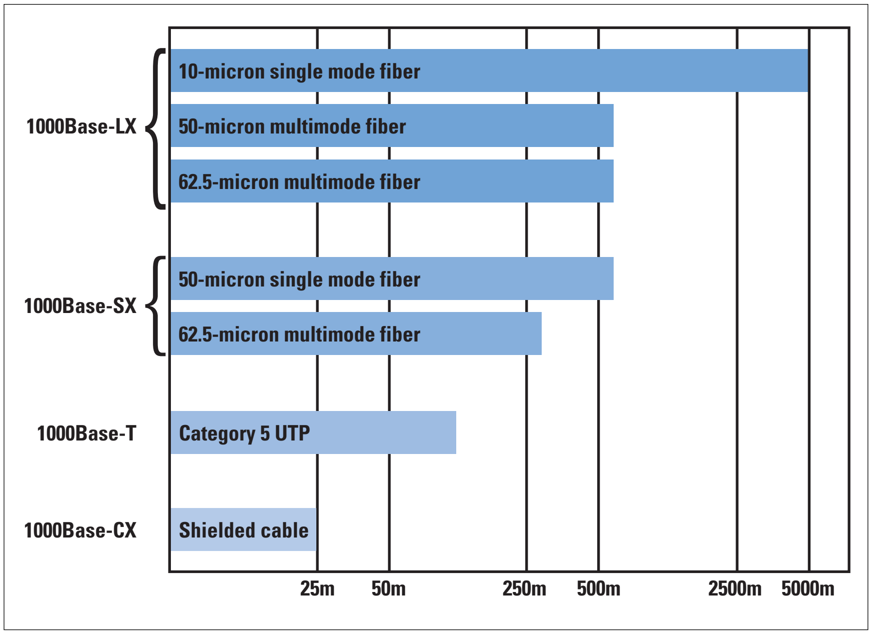
With a LAN switch (full-duplex operation), which provides dedicated rather than shared access to the medium, the carrier extension and frame bursting features are not needed. They are unnecessary because data transmission and reception at a station can occur simultaneously without interference and with no contention for a shared medium. All the gigabit products on the market use a switching technique, and so do not implement the carrier extension and frame bursting.

With a switching technique, full-duplex operation is employed, and the CSMA/CD protocol is not needed. The gigabit specification expands on the pause protocol that is defined for 100-Mbps Ethernet by allowing asymmetric flow control. Using the autonegotiation protocol, a device may indicate that it may send pause frames to its link partner but will not respond to pause frames from its partner.

Physical Layer

The current 1-Gbps specification for IEEE 802.3 includes the following physical-layer alternatives (Figure 3):

Figure 3: Gigabit Ethernet Media Options (log scale)



- *1000Base-LX:* This long-wavelength option supports duplex links of up to 550 m of 62.5-µm or 50-µm multimode fiber or up to 5 km of 10-µm single-mode fiber. Wavelengths are in the range of 1270 to 1355 nm.
- *1000Base-SX:* This short-wavelength option supports duplex links of up to 275 m using 62.5-µm multimode or up to 550 m using 50-µm multimode fiber. Wavelengths are in the range of 770 to 860 nm.
- *1000Base-CX:* This option supports 1-Gbps links among devices located within a single room or equipment rack, using copper jumpers (specialized shielded twisted-pair cable that spans no more than 25 m). Each link is composed of a separate shielded twisted-pair running in each direction.
- *1000Base-T:* This option makes use of four pairs of Category 5 unshielded twisted-pair copper wires to support devices over a range of up to 100 m.

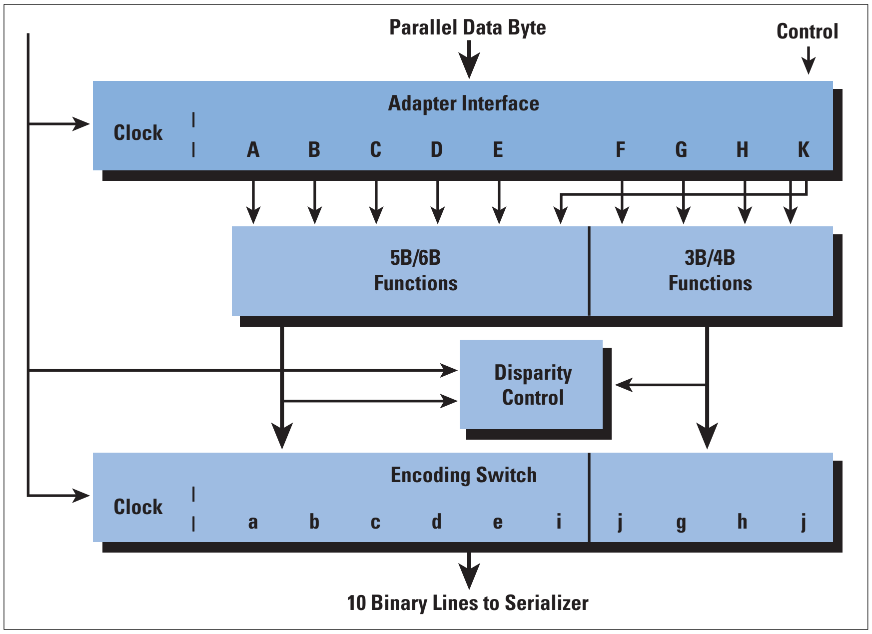
Digital Signal Encoding Techniques for Gigabit Ethernet

The encoding scheme used for all the Gigabit Ethernet options except twisted-pair is 8B/10B. This scheme is also used in Fibre Channel. With 8B/10B, each 8 bits of data is converted into 10 bits for transmission. The 8B/10B scheme was developed and patented by IBM for use in its 200-megabaud ESCON interconnect system.

- The developers of this code list the following advantages:
- It can be implemented with relatively simple and reliable transceivers at low cost.
- It is well balanced, with minimal deviation from the occurrence of an equal number of 1 and 0 bits across any sequence.
- It provides good transition density for easier clock recovery.
- It provides useful error-detection capability.

The 8B/10B code is an example of the more general *m*B*n*B code, in which *m* binary source bits are mapped into *n* binary bits for transmission. Redundancy is built into the code to provide the desired transmission features by making $n > m$. Figure 4 illustrates the operation of this code. The 8B/10B code actually combines two other codes, a 5B/6B code and a 3B/4B code. The use of these two codes is simply an artifact that simplifies the definition of the mapping and the implementation; the mapping could have been defined directly as an 8B/10B code. In any case, a mapping is defined that maps each of the possible 8-bit source blocks into a 10-bit code block. There is also a function called *disparity control*. In essence, this function keeps track of the excess of zeros over ones or ones over zeros. An excess in either direction is referred to as a disparity. If there is a disparity, and if the current code block would add to that disparity, then the disparity control block complements the 10-bit code block. This complement has the effect of either eliminating the disparity or at least moving it in the opposite direction of the current disparity.

Figure 4: 8B/10B Encoding



The encoding mechanism also includes a control line input, K, which indicates whether the lines A through H are data or control bits. In the latter case, a special nondata 10-bit block is generated. A total of 12 of these nondata blocks are defined as valid in the standard. These blocks are used for synchronization and other control purposes.

For 1000Base-T, the encoding scheme used is PAM-5, over four twisted-pair links. Therefore, each link must provide a data rate of 250 Mbps. PAM-5 provides better bandwidth utilization than simple binary signaling by using five different signaling levels. Each signal element can represent two bits of information (using four signaling levels). In addition, a fifth signal level is used in a forward error correction scheme.

References

A good tutorial on Gigabit Ethernet is [1]. The Gigabit Ethernet Alliance is at **`http://www.gigabit-ethernet.org`**

[1] Frazier, H., and Johnson, H. "Gigabit Ethernet: From 100 to 1,000 Mbps." *IEEE Internet Computing,* January/February 1999.

[2] *Gigabit Ethernet: Technology and Applications for High-Speed LANs* by Rich Seifert, ISBN 0-201-18553-9, Addison-Wesley, 1998. (Reviewed in *The Internet Protocol Journal,* Volume 1, Number 2, September 1998.)

WILLIAM STALLINGS is a consultant, lecturer, and author of more than a dozen books on data communications and computer networking. He has a PhD in computer science from M.I.T. His latest book is *Data and Computer Communications, Sixth Edition* (Prentice Hall, 1999). His home in cyberspace is **`http://www.shore.net/~ws`** and he can be reached at **`ws@shore.net`**

One Byte at a Time: Is Your FTP Active or Passive?

by Thomas M. Thomas, NetCerts

What many people don't know is that the *File Transfer Protocol* (FTP) has multiple modes of operation that can dramatically affect its operation and, as a result, the security of your network. These modes of operation determine whether the FTP server or FTP client initiates the TCP connections that are used to send information from the server to the client. The FTP protocol supports two modes of operation, as follows:

- The first FTP mode of operation is known as *normal,* though it is often referred to as *active*. This mode of operation is typically the default.
- The second FTP mode of operation is known as *passive*.

In active (normal) FTP, the client opens a control connection on port 21 to the server, and whenever the client requests data from the server, the server opens a TCP session on port 20. In passive FTP, the client opens the data sessions, using a port number supplied by the server.

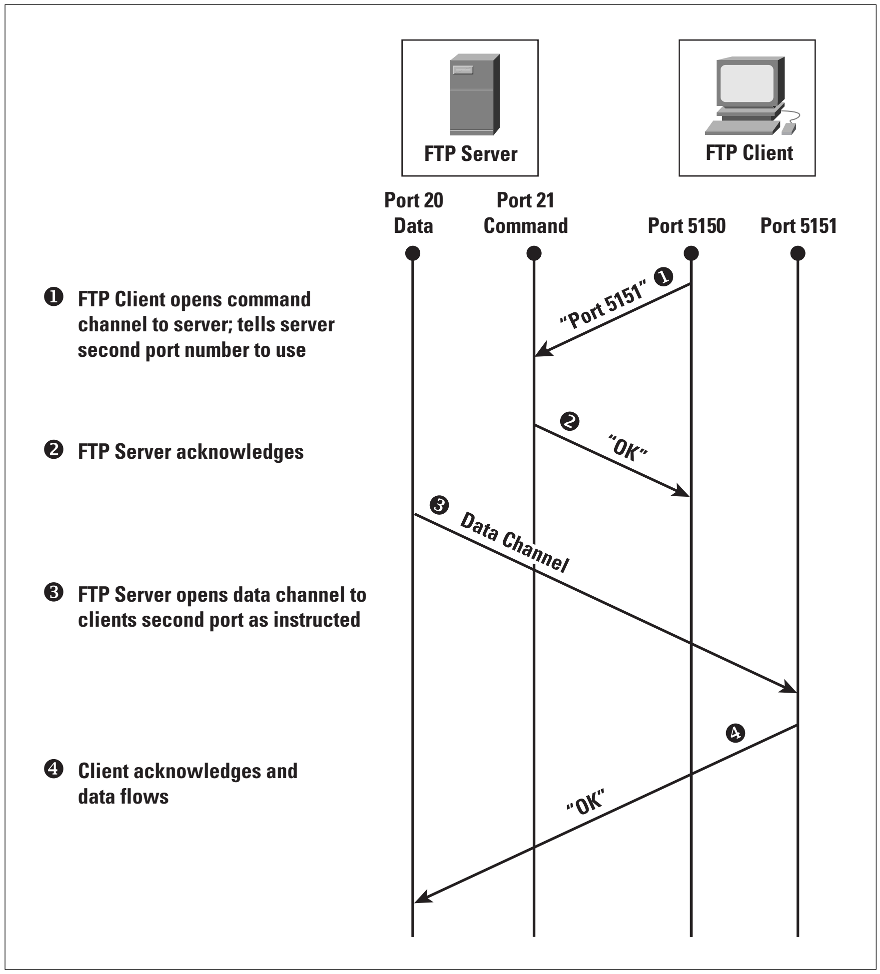
Active FTP Operation

The active mode of operation is less secure than the passive mode. This mode of operation complicates the construction of firewalls, because the firewall must anticipate the connection from the FTP server back to the client program. The steps of this mode of operation are discussed below and are shown in Figure 1.

- The client opens a control channel (port 21) to the server and tells the server the port number to respond on. This port number is a randomly determined port greater than 1023.
- The server receives this information and sends the client an acknowledgement "OK" (ack). The client and server exchange commands on this control connection.
- When the user requests a directory listing or initiates the sending or receiving of a file, the client software sends a "PORT" command that includes a port number > 1023 that the client wishes the server to use for the data connection.
- The server then opens a data connection from port 20 to the client's port number, as provided to it in the "PORT" command.

The client acknowledges and data flows.

Figure 1: Active-Mode FTP Connection



Passive FTP Operation

This mode of operation is assumed to be more secure because all the connections are being initiated from the client, so there is less chance that the connection will be compromised. The reason it is called passive is that the server performs a "*passive open*." The steps of this mode of operation are discussed below and are shown in Figure 2.

- In passive FTP, the client opens a control connection on port 21 to the server, and then requests passive mode through the use of the "PASV" command.
- The server agrees to this mode, and then selects a random port number (>1023). It supplies this port number to the client for data transfer.
- The client receives this information and opens a data channel to the server-assigned port.

The server receives the data and sends an "OK" (ack).

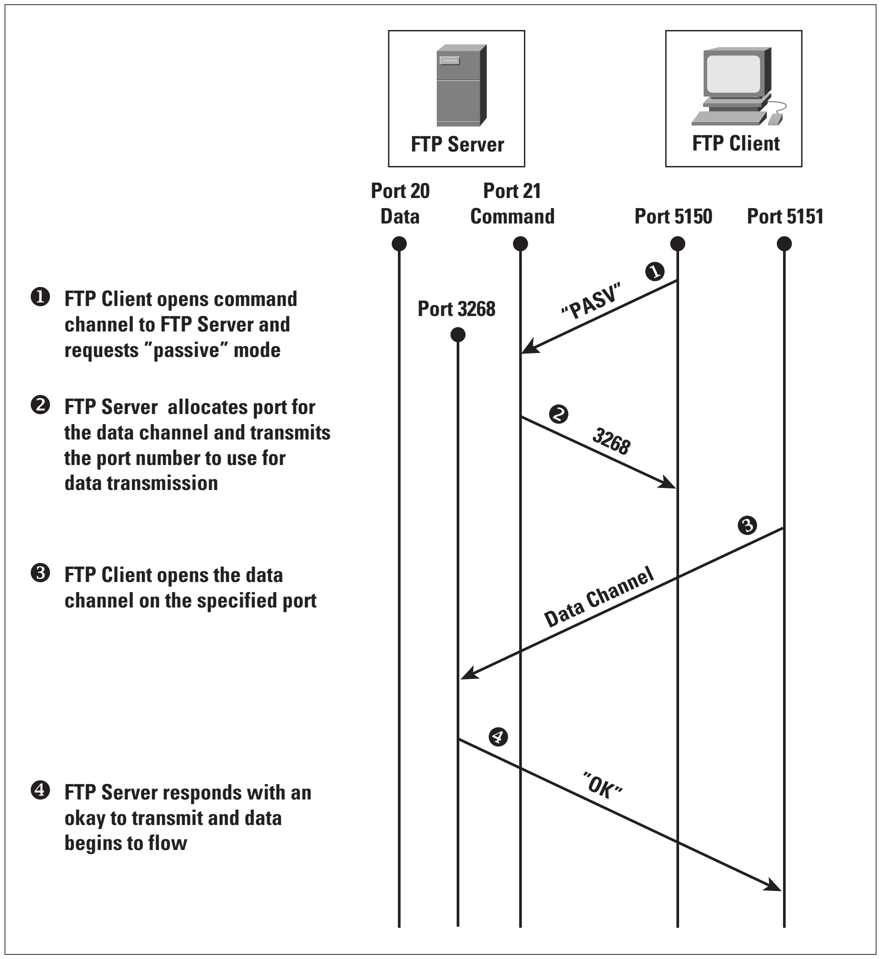


Figure 2: Passive-Mode FTP Connection

References

[1] *Internetworking With TCP/IP, Volume 1: Principles, Protocols, Architecture, Third Edition,* by Douglas E. Comer, ISBN 0-13-216987-8, Prentice Hall, 1995.

[2] R. Braden, "Requirements for Internet hosts—application and support," RFC 1123, October 1989.

[3] S. Bellovin, "Firewall-Friendly FTP," RFC 1579, February 1994.

[4] P. Deutsch, A. Emtage, A. Marine, "How to Use Anonymous FTP," RFC 1635, May 1994.

THOMAS M. THOMAS II has recently founded his own company, NetCerts (**`www.netcerts.com`**), to assist network engineers working toward their Cisco Certifications. Before starting NetCerts, Tom worked as a Course Developer at Cisco Systems for the Worldwide Training division. He worked as part of a team on a new course on Multilayer Switching. He also wrote the book *OSPF Network Design Solutions* for Cisco Press. Tom has also worked as a senior network engineer and group leader of the Advanced Systems Solutions Engineering Team for MCI's Managed Network Services. In this capacity, he developed network maintenance Standard Operating Procedures and performed various in-house training duties. Before joining MCI, Tom worked as a technical team leader at AT&T Solutions, where he provided technical support and network management for Cisco routers over ATM and Frame Relay and configured various networking protocols. E-mail: **`tothomas@netcerts.com`**

Letter to the Editor

The article "Was the Melissa Virus so Different?" (*The Internet Protocol Journal,* Volume 2, Number 2, June 1999) by Barbara Y. Fraser et al makes an interesting comparison between events in our real and virtual lives, comparing e-mail borne viruses with commercial samples delivered to our physical mail boxes. While I think the comparison is a useful exercise, the authors fail to point out one of the fundamental differences between these two worlds.

An electronic message contains a finite amount of information: a careful sender can make sure his identity cannot be revealed. In contrast, a physical "message" (i.e., mail bomb, extortion letter, etc.) contains an essentially unlimited amount of information: from finger prints and material analysis to DNA traces, a potential perpetrator can never be certain that he can deny his involvement. For cyberspace crimes the chance to be caught is (and is perceived to be) much smaller. As a result, many virus authors have but the slimmest motive for their deed.

The fact that the Melissa author was quickly identified because of a hidden signature in Microsoft Word is little comfort. For reasons of privacy, this feature has been disabled: it was a bug, not a feature.

To extend the analogy: suppose a simple device would become available that can look up a person's full ID based on a DNA trace (a few molecules) on any object touched or handled. Move the scanner over the door handle and you know who's been visiting. The ramifications would be extensive. Most likely, the as-yet hypothetical device would be illegal except for police use.

—*Ernst Lopes Cardozo, Aranea Consult BV*
e.lopes.cardozo@aranea.nl

Send us your comments!

We look forward to hearing your comments and suggestions regarding anything you read in this publication. Send us e-mail at: **ipj@cisco.com**

Changes at the IPJ Web Site

Now you can find every issue of *The Internet Protocol Journal* in both PDF and HTML format at **www.cisco.com/ipj**. We are also pleased to announce that Nikkei Business Publications in Tokyo has provided an introduction to IPJ in Japanese, as well as translation of some of the titles from previous issues at: **http://nit.nikkei.co.jp/ipj.html**. We hope to set up similar links with other publications around the world.

Book Reviews

DHCP *DHCP—A Guide to Dynamic TCP/IP Network Configuration,* by Berry Kercheval, ISBN 0-13-099721-8, Prentice Hall PTR, 1998, `http://www.prenhall.com/ptrbooks/ptr_0130997218.html`

First, I should note that this book arrived at the perfect time for me: I am involved in adding *Dynamic Host Configuration Protocol* (DHCP) support to a software product and needed a quick, thorough understanding of DHCP that went into sufficient detail to support some key design decisions. The book provided me with exactly what I wanted. However, as to whether or not this is a book you should own or even want to read, that is a much more difficult question to answer.

Organization

The author begins with a chapter of general background information. Then, in a logical progression, he goes through an overview of DHCP and on to explicit details of both the client and server aspects of the protocol. In other sections he covers server administration, DHCP and IP Version 6 (IPv6), and the future of DHCP. He then briefly reviews a few available implementations. In supporting sections he covers the relationship between DHCP and the *Domain Name System* (DNS), specifically Dynamic DNS. In one chapter he discusses the relationship between directory services and DHCP, in particular, the *Lightweight Directory Access Protocol* (LDAP). He then concludes with three appendices: one lists DHCP vendors, another covers the available DHCP options, and a final appendix provides the DHCP RFCs, RFC 2131 and RFC 2132.

Presentation

Overall, the book is well planned and easy to read. The background information is clearly written and gives sufficient material to assure that even novice readers will not get left behind. The author clearly explains the origins of DHCP in BOOTP and the continuing relationship between the two protocols. He also provides many examples that help make the more difficult aspects of DHCP easier to grasp. The chapters tend to progress in a logical order, making absorption of the fairly technical subject almost easy.

The presentation, however, is somewhat marred by minor errors and omissions. None of these mistakes would confuse an expert, but they will make it harder for the novice to be sure what he or she is to understand. In one example, a client workstation on net 10.0.1.0 is offered, and selects, an address of 10.0.2.32. This scenario is, however, clearly unroutable, and the example only confuses the reader. The author also makes a good effort at defining terms the first time they are used, and then again in an extensive glossary. However, for some reason he never defines two key terms: *broadcast* and *multicast.* Since both techniques are core to understanding DHCP, this oversight is difficult to understand.

The chapters on DHCP are fairly exhaustive in their examination of the protocol from overview to minutiae. The roles of clients, servers, and relay agents are well described and documented with sample packets. Each packet field is thoroughly explained and easy to grasp. However, the sections of LDAP and Dynamic DNS could have been presented better. The reader is left with a glimpse of possible relationships between the protocols, but without enough information to really pull it all together. Notably missing is any mention of remote access and the *Remote Authentication Dial-In User Service* (RADIUS) protocol. DHCP and RADIUS perform similar functions in different situations, and there has been much discussion in the past year or two about use of DHCP to manage RADIUS IP address assignments.

Summary

This book sets out to accomplish a limited goal: informing the reader about the basics of DHCP. A couple of detours along the way provide useful information about related technologies (such as DNS and LDAP). The author makes no assumptions about the user's technical capability and level of knowledge. This is perhaps the book's major strength and its biggest weakness. Because of his assumptions about the reader's technical ability, a lot of space is devoted to giving background and reference information assuring that the reader has the necessary foundation to understand the more complex aspects of DHCP. If the background information and appendices (all of which are available on the net and consist mostly of the RFCs anyway) are removed from the book, little is left: without the appendices there are only 144 pages. Given that the book costs $45, and that the 144 pages are essentially a guided explanation of the RFCs anyway, the technically competent reader might do just as well to download the RFCs and slog through them.

However, for the non-technical reader, or someone who just wants it all in one convenient volume, the author's approach is well worth the cost of the book and the (short) time required to read it. Explanations are clear and concise, terms are well defined, and everything the reader needs to grasp about the complexities of DHCP is right there, in a logical order.

—*Richard Perlman, Lucent Technologies*
perl@lucent.com

Information Warfare

Information Warfare and Security, Dorothy E. Denning, ISBN 0-201-43303-6, Addison-Wesley, 1999, **`http://www.awl.com/cseng/0-201-43303-6/`**

It has been said that "information is power," and they who control the information control the power. Whether the information is broadcast on the evening news, printed in a newspaper, etched on stone tablets, or published on a USENET newsgroup or Internet Web page, we rely on information in our daily lives, and trust that most of the information we receive and process is accurate.

"Information warfare." What images does it conjure up for you? Propaganda wars via pamphlets dropped from airplanes, or "cyber-terrorists" versus the FBI on the Internet—or something else entirely? Dr. Denning covers all bases in this, her latest book. The "warfare" of the title is specifically the battle between the good guys and "information terrorists."

This book is a textbook for a course by the same name at Georgetown University. No one, however, should be scared off by this knowledge. This book is incredibly approachable, intended for a broad audience. It is an introduction to information warfare, but really concentrates on computer- and network-based information. Anyone involved or interested in computer and network security would benefit from this book. Many sections are self-contained, so a reader can jump back and forth among the sections. All the sections are interesting and informative, and should be to both the highly technical reader as well as those for whom technology is peripheral to their jobs, but who require or desire deeper and broader knowledge of information warfare.

About the Author

Dorothy E. Denning is Professor of Computer Science at Georgetown University. She is a well-known expert in the areas of computer security and cryptography, and has been called as an expert witness to testify before the U.S. Congress. She is the author of over 100 papers on computer and Internet security, and has written three other books in addition to this one: *Cryptography and Data Security* (a coeditor with Peter Denning), *Rights and Responsibilities of Participants in Networked Communities*, with Herbert S. Lin, and *Internet Besieged: Countering Cyberspace Scofflaws*. She is also a frequent contributor to security-related publications.

Organization

Information Warfare and Security has three parts. Part 1 starts with a very exciting (and still timely) discussion of the role information warfare played in the Gulf War in the early 1990s. The tone and flavor of this opening chapter continues throughout the book. Randomly put your finger in the book and you will be able to start an enjoyable and interesting read (though I recommend reading beginning to end). Part 1 introduces basic concepts upon which the work is built. Chapters 2 and 3 present a taxonomy of information warfare, relating it to information security and assurance, and suggesting four arenas of activity: play, crime, individual rights, and national security. The author discusses goals, motivations, culture, and concerns. Included is the no-doubt apocryphal, but always fun, quote attributed to Secretary of State for War Henry Stimson, upon the 1929 "discovery" of the Black Chamber code-breaking operation: "Gentlemen, do not read one another's mail."

Part 2 focuses on offense. This section covers topics that, for the most part, will be new to many readers. The chapters cover open source material and privacy (and piracy of information), "social engineering," and its kin. The threat from insiders—legitimate and those who have broken in, gets a thorough treatment. Eavesdropping also is examined, from cellular and pager intercepts, to the mysterious-to-most-people area of traffic analysis, to surveillance, packet-sniffing, and other electronic eavesdropping attacks.

Chapter 8 looks in detail at well-known computer hacking techniques and the tools that implement the attacks. Chapter 9 discusses identity theft, including forged e-mail and stolen accounts, IP-spoofing (stealing the identity of a computer), and Trojan Horse attacks. Finally, Part 2 ends with a chapter dedicated to computer viruses, both real and hoaxes.

Topics discussed in Part 3, "Defensive Information Warfare," will be familiar to most readers who understand computer and network security. Chapter 11 not only describes cryptographic techniques for protecting information, but also covers *steganography,* or "the practice of hiding a message in such a manner that its very existence is concealed"—and anonymity. Chapter 12, "How to Tell a Fake," deals with methods for determining identity or trustworthiness of entities or information. Chapter 13 talks about access control mechanisms, including firewalls, and intrusion detection. Covering vulnerability monitoring and analysis, risk analysis, risk management, and incident response, Chapter 14 possibly should have started Part 3. Devices, mechanisms, and methods should be deployed after an understanding of what is contained in this chapter. Part 3, and the book, end with a chapter dedicated to discussing the role of government in defensive information warfare. Also included are descriptions of recent (1990s) actions, laws, and initiatives of the U.S. Government in this area.

Throughout, the book is seasoned with stories—infowar stories, if you will—and background information, allowing the novice not only to understand, but also to enjoy learning what is contained within.

A Book for the Lecture Hall or Armchair

It is not surprising that *Information Warfare and Security* so thoroughly covers the space of information warfare theory, measures, and countermeasures, not because it weighs in at over 500 pages, but because it was written as a text for a course that had to cover all of this material. What may be surprising to readers unfamiliar with Dr. Denning is that such complete coverage could be done in such an easy-to-read way. I have no doubt that this book is and will continue to be useful and effective in the classroom. In addition, the reader studying for accreditation in a field requiring this knowledge, or the professional wanting to "brush up," "fill in," or just "kick back," will find much here to commend itself.

—*Frederick M. Avolio, Avolio Consulting*
fred@avolio.com

Cryptonomicon

Cryptonomicon, Neal Stephenson, ISBN 0-380-97346-4, Avon Books, 1999. **http://www.cryptonomicon.com/main.html**

It isn't often that you find reviews of works of fiction in these pages, but *Cryptonomicon* deserves special treatment. Neal Stephenson's latest work is a 918-page science fiction World War II thriller that I couldn't put down. You have to love a novel that has plot points that depend on the technical details of prime number theory, Pretty Good Privacy (PGP), public key infrastructure (PKI), Secure Shell (SSH), Global Positioning System (GPS), secure e-mail, and other Internet applications. Truly this is an epic novel of techno-epic proportions.

The story takes places during both World War II and modern times. The contemporary action revolves around an offshore data haven created by a Silicon Valley startup with the usual coterie of managers, venture capitalists, lawyers with class-action suits, marketeers, and nerds that you'll easily recognize. These entrepreneurs think nothing of flying across the Pacific to attend a meeting and then flying home to get in some quality family time.

The war setting revolves around a small group of code crackers who travel around the globe planting misinformation behind German and Japanese lines. The two groups are literally related: the modern generation is the progeny of the wartime crackers. Both groups are going after hidden caches of gold, among other things, buried near the Philippines.

Technology

There is much technology here for any self-respecting computer geek to digest. Think of Tom Clancy playing with the latest laptops and the Internet rather than with the latest guns. There is even an appendix describing the technical details of one of the crypto algorithms using synchronized decks of playing cards (a key plot point in the book). Stephenson blends in descriptions of undersea cable laying and salvage operations with the cracking of the *Enigma*[1] codes and hunting down German submarines. At one point, the code-cracking wartime division has to change its numerical designation because it can be factored into two prime numbers—too obvious.

One of my favorite scenes happens early in the book, when the modern-day principals of the crypto firm are meeting some of their backers and potential clients for the first time. The firm's engineer (using the built-in pinhole camera of the laptop) programs his UNIX laptop to surreptitiously capture a photo of whoever is using the keyboard during a demo of the firm's crypto technology, but hides his program in a way that any UNIX hacker would appreciate. He then e-mails the collected digital photos to a friend to try to confirm their identity.

Balance

Unlike Clancy, this book has characters with some depth to them and doesn't overdo the technology. The relationship of the war and modern-day periods is nicely tied together in the end, and the familiarity of the modern-day business relationships is sometimes almost too painful to read.

—David Strom, publisher of Web Informant
david@strom.com

References

[1] See **http://www.nsa.gov:8080/museum/enigma.html**

Would You Like to Review a Book for IPJ?

We receive numerous books on computer networking from all the major publishers. If you've got a specific book you are interested in reviewing, please contact us and we will make sure a copy is mailed to you. The book is yours to keep if you send us a review. We accept reviews of new titles, as well as some of the "networking classics." Contact us at **ipj@cisco.com** for more information.

Call for Papers

The Internet Protocol Journal (IPJ) is published quarterly by Cisco Systems. The journal is not intended to promote any specific products or services, but rather is intended to serve as an informational and educational resource for engineering professionals involved in the design, development, and operation of public and private internets and intranets. The journal carries tutorial articles ("What is…?"), as well as implementation/operation articles ("How to…"). It provides readers with technology and standardization updates for all levels of the protocol stack and serves as a forum for discussion of all aspects of internetworking.

Topics include, but are not limited to:

- Access and infrastructure technologies such as: ISDN, Gigabit Ethernet, SONET, ATM, xDSL, cable fiber optics, satellite, wireless, and dial systems
- Transport and interconnection functions such as: switching, routing, tunneling, protocol transition, multicast, and performance
- Network management, administration, and security issues, including: authentication, privacy, encryption, monitoring, firewalls, trouble-shooting, and mapping
- Value-added systems and services such as: Virtual Private Networks, resource location, caching, client/server systems, distributed systems, network computing, and Quality of Service
- Application and end-user issues such as: e-mail, Web authoring, server technologies and systems, electronic commerce, and appli-cation management
- Legal, policy, and regulatory topics such as: copyright, content control, content liability, settlement charges, "modem tax," and trademark disputes in the context of internetworking

In addition to feature-length articles, IPJ will contain standardization updates, overviews of leading and bleeding-edge technologies, book reviews, announcements, opinion columns, and letters to the Editor.

Cisco will pay a stipend of US$1000 for published, feature-length articles. Author guidelines are available from Ole Jacobsen, the Editor and Publisher of IPJ, reachable via e-mail at **`ole@cisco.com`**

Fragments

More ICANN News

The *Internet Corporation for Assigned Names and Numbers* (ICANN) recently announced that seven additional applicant companies have met its registrar accreditation criteria.

As accredited registrars, these seven companies will compete in the market for domain name registration services in the **`.com`**, **`.net`**, and **`.org`** domains. In addition, they will be able to participate the ongoing testbed program for the *Shared Registry System,* which allows multiple ICANN-accredited registrars to provide domain name registration services in these domains. Under an agreement announced August 6 by the U.S. Department of Commerce and Network Solutions, Inc. (NSI—the developer of the Shared Registry System), new registrars that have signed an accreditation agreement with ICANN will be eligible to join the initial five testbed registrars as participants in the testbed operation. The testbed phase is currently scheduled to conclude on September 10, 1999.

The seven new companies join the 57 companies that have already been accredited by ICANN starting in April, 1999. Until the initial introduction of competition in June, registration services in the **`.com`**, **`.net`**, and **`.org`** domains were provided solely by NSI under a 1992 Cooperative Agreement with the U.S. Government.

The additional seven companies named are: CommuniTech.Net, Inc. (United States), GANDI (France), iDirections, Inc. (United States), InterNeXt (France), ProBoard Technologies (United States), PSI-USA (United States), and Signature Domains, Inc. (United States). Further information about these companies will be made available on the ICANN Web site:
`http://www.icann.org/registrars/accreditation.html`

Under an October 6, 1998 amendment to the Cooperative Agreement between NSI and the U.S. Government, the process of opening the Internet Domain Name System's three largest domains to competition was launched with a testbed phase that began on April 26. Five companies were initially accredited to use the NSI Shared Registry System in a test operation designed to ensure that the introduction of competition occurs in a smooth, coordinated manner.

By qualifying to be accredited as registrars, the seven new registrars join the five original testbed registrars, as well as the 52 other companies that have already qualified for ICANN accreditation. The Shared Registry System testbed program has been expanded to extend to all accredited registrars that sign the standard testbed registrar agreements with NSI and meet technical certification requirements.

ICANN is a non-profit, international corporation formed in September 1998 to oversee a select set of Internet technical management functions currently managed by the U.S. Government, or by its contractors and volunteers. Specifically, ICANN is assuming responsibility for coordinating the management of the *Domain Name System* (DNS), the allocation of IP address space, the assignment of protocol parameters, and the management of the root server system. For more information, see **`http://www.icann.org`**. Here you will also find information about ICANN's upcoming public meetings.

INET 2000

INET 2000: The Internet Global Summit, is a special INET. Hosted by the Internet Society, the Summit will be held 18–21 July 2000, in Yokohama, Japan. The place, the date, and the fact that it is the 10th anniversary of this important event all mark it as an exceptional year.

To be considered as a speaker, panelist, tutorial instructor, or poster presenter, please see **`http://www.isoc.org/inet2000/callforabstracts.shtml`** for submission instructions and to read about this year's theme, "Global Distributed Knowledge for Everyone."

INET is the premier international event for Internet and internetworking professionals. Nowhere can such a broad cross-section of important movers of the Internet be found in one single location.

We look forward to receiving your abstract and seeing you in Japan!

—Jean-Claude Guedon and Jun Murai
Co-Chairs, INET 2000 Program Committee

Y2K and The Internet

As the countdown to the Year 2000 continues, a number of efforts are underway to ensure that the Internet continues to operate normally on January 1, 2000. Here we include some pointers to recent activities.

On July 30, 1999, the *President's Council on Year 2000 Conversion,* convened a roundtable meeting to examine the readiness of the Internet for the Year 2000 date change, and to coordinate efforts to maintain Internet performance and reliability during the transition to the new millennium. The roundtable brought together roughly 100 prominent organizations and individuals from different parts of the Internet community to discuss the Internet's Y2K readiness. Meeting participants included small and large ISPs, equipment vendors, root name server and domain registries, exchange points, network time servers, industry associations, and government officials. For more information see:
`http://www.y2k.gov/` and **`http://www.mids.org/y2k/`**

For small- and medium-sized businesses in the U.S. and in key trading partner countries, the U.S. Department of Commerce (DoC) is providing a strategic management tool to help battle the millennium bug. The *Y2K Self-Help Tool/CD-ROM* contains a software program that enables users to complete an inventory of assets that may be susceptible to Y2K problems, gauge the criticality of business processes, develop contingency plans and conduct remediation activities.

This CD-ROM contains a 10-minute discussion video, the software program for managing your Y2K process, a self-assessment checklist, contingency planning template, user guide and hotlinks to many helpful Y2K sites. It has been produced in several languages including English, Spanish, Mandarin Chinese, Japanese, French, Portuguese, Arabic and Russian. The software was developed by the DoC's National Institute of Standards and Technology Manufacturing Extension Partnership (MEP) in cooperation with the U.S. Department of Agriculture and the U.S. Small Business Administration.

To receive just the software, visit: **`www.nist.gov/y2k/software.htm`** and download *Conversion 2000: Y2K Jumpstart Kit*. To receive the complete CD-ROM with video and hotlinks, you can call 1-800-Y2K-7557 and ask for the Self-Help Tool in any of the languages listed above. If you are an association or organization interested in multiple copies of the CD-ROM for your members and staff, click on order form, print the form, complete the requested information, and fax it to 202-482-0077. Please note that there is a minimal charge for orders over 100 copies for duplication and shipping.

The *Internet Engineering Task Force* (IETF) has examined all of the protocol standards and related documents to identify any potential inherent Y2K problems in the Internet Protocol Suite. The resulting report, RFC 2626, "The Internet and the Millennium Problem (Year 2000)" can be found at **`http://www.ietf.org/rfc/rfc2626.txt`**

See also:

`http://www.apia.org`
`http://www.nety2k.org/`
`http://www.cert.org/y2k/indmessage.html`
`http://www.icann.org/committees/dns-root/y2k-statement.htm`

The Internet Protocol Journal is published quarterly by the Cisco News Publications Group, Cisco Systems, Inc.
www.cisco.com
Tel: +1 408 526-4000
E-mail: ipj@cisco.com

The Internet Protocol Journal, Cisco Systems
170 West Tasman Drive, M/S SJ-10/5
San Jose, CA 95134-1706
USA

ADDRESS SERVICE REQUESTED

Bulk Rate Mail
U.S. Postage
PAID
Cisco Systems, Inc.