NCSA's World Wide Web Server: Design and Performance

Thomas T. Kwan and
Robert E. McGrath
National Center for Supercomputing Applications

Daniel A. Reed
University of Illinois

Explosive Web traffic growth has placed burdens on Web servers that lie far outside today's normal operating regime. This article examines extant Web access patterns with the aim of developing more efficient file-caching and prefetching strategies.

The recent explosion of interest in the World Wide Web (WWW)[1] can be traced to the distribution of the CERN (European Laboratory for Particle Physics in Geneva, Switzerland) and NCSA (National Center for Supercomputing Applications) servers and WWW client browsers. In particular, NCSA Mosaic, the graphical user interface for WWW browsing, based on distributed, multimedia hypertext, has spawned several commercial variants and has made the Internet readily accessible to a much larger population than in the past.

Network statistics from Merit, the NSFNet backbone management group, show that WWW traffic is the largest and by far the fastest growing segment of the Internet, and growing numbers of government and commercial groups are making hundreds of gigabytes of data available via WWW servers. At the same time, the WWW servers at NCSA have experienced explosive growth in traffic, from 1 million requests per week in February 1994, to 2 million per week in June 1994, 3 million per week in September 1994, nearly 4 million per week in December 1994, and even larger numbers in 1995.[2]

To support continued growth, WWW servers must manage a multigigabyte (in some instances a multiterabyte) database of multimedia information while concurrently serving multiple request streams. This places demands on the servers' underlying operating systems and file systems that lie far outside today's normal operating regime. Simply put, WWW servers must become more adaptive and intelligent. The first step on this path is understanding extant access patterns and responses. On the basis of this understanding, one can then develop more efficient and intelligent server and system file-caching and prefetching strategies.

In this article, we describe extant access patterns and responses at NCSA's WWW server and the implications of that data. But first, we describe the context in which the data was collected—the NCSA WWW server architecture.

NCSA WWW SERVER ARCHITECTURE

Shortly after NCSA's WWW server was established, it became clear that the volume of WWW traffic would stress operating systems and network implementations in ways not originally envisioned by their designers. At peak times, the NCSA server receives 30-40 new WWW requests per second, and because the Hypertext Transfer Protocol (HTTP) is connectionless, each such request appears to the server as a separate network connection.

Not only were most implementations of the TCP/IP network protocol not designed to accept connections at this sustained rate, even conservative projections of request rate growth showed that no single processor system could serve all requests. To support the growing request rate, NCSA has developed a scalable WWW architecture that consists of a group of loosely coupled WWW servers. Though the servers operate independently, collectively they provide the illusion of a single server.

Development of the NCSA architecture required resolution of three key problems:

0018-9162/95/$4.00 © 1995 IEEE

1. *Information addressing*. Externally, the NCSA server has a single domain name (www.ncsa.uiuc.edu). Incoming requests addressed to this domain name must be mapped to multiple servers, each with a separate, user-invisible domain name. This mapping allows NCSA to invisibly add servers to accommodate the growing number of incoming requests.
2. *Information distribution*. Each server must be capable of responding to requests for any portion of the NCSA WWW server database. Otherwise, the servers must be more tightly coupled, an arbiter must distribute requests to servers on the basis of request type, and it is likely that the arbiter will become the bottleneck.
3. *Load balancing*. The requests must be equally apportioned among the servers. Thus, newly added servers will always share the load and contribute to the scalability of the implementation.

The server architecture is based on three components: a collection of independent servers, a WWW document tree shared among the servers and stored by the Andrew distributed file system (AFS),[3] and a round-robin domain name system (DNS) that multiplexes the domain name www.ncsa.uiuc.edu among the constituent servers.

With this architecture, the NCSA WWW *service* is always the same, although the number and identity of the particular *servers* may change from day to day. Beginning with one server in February 1994, the architecture grew to four servers in May, eight servers in November, and nine in early 1995. To meet increasing demand, NCSA will continue to add servers as needed.

Below, we briefly describe each of the three server components. For additional details on the server architecture, see Katz et al.[4] and Kwan et al.[5]

The servers and the network

The NCSA WWW server architecture is flexible enough to accommodate most Unix systems as component servers. The only requirements are that the systems function as AFS clients and support TCP/IP. The servers need not be homogeneous; the particular systems in use vary from time to time and may be a heterogeneous collection of systems.

To date, the backbone of the NCSA WWW service has been a group of dedicated Hewlett-Packard HP 735 workstations. Though these systems are not generally considered "servers," their efficient TCP/IP implementation has made them an effective choice to process WWW requests. In the NCSA configuration, each HP 735 has 96 megabytes of memory and uses its local disk as a moderate-size (130 megabytes) AFS cache. In addition, the local disk stores HTTP server log files and is the backing store for the virtual memory system.

The WWW servers are connected to the AFS file servers via a 100-megabit/second Fiber Distributed Data Interface ring (see Figure 1). The FDDI ring connects to the rest of NCSA and to the Internet via a T3 line.

NCSA AFS configuration

All documents provided by the NCSA WWW service are served from the NCSA center-wide Andrew File System environment.[6] This distributed file system is shared by many hundreds of client workstations and supercomputers, as well as the WWW servers.

AFS provides a single, consistent view of the file system to each WWW server, allowing each server to access the entire WWW document tree. Because AFS clients (that is, the WWW servers) cache recently used files on their local disks, the most frequently accessed documents are generally available locally, without remote disk access. In effect, AFS caching replicates the document tree on each WWW server.

Because AFS manages the shared document tree, the individual WWW servers need not and do not know either the number or identity of the other servers. It is difficult to overemphasize the importance of this point. This allows rapid, "plug-and-play" addition (and removal) of component servers and the use of heterogeneous systems. In practice, we have found that servers can be added or removed from the ensemble in under an hour.

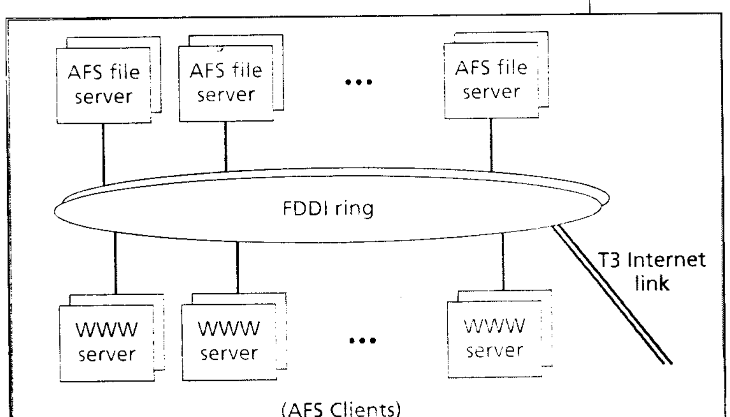

Figure 1. The NCSA scalable WWW server.

Glossary

Browser—Software that allows users to view documents retrieved from the Internet.
Client—Software responsible for communicating with servers to retrieve necessary documents and files.
Firewall—A computer system and network interface that maintains security for an organization by filtering incoming networking requests.
HTTP—Hypertext Transfer Protocol (HTTP) is a stateless data-transfer protocol used by the WWW.
NCSA Mosaic—A freely available browser developed at NCSA. (NCSA Mosaic is a trademark of the Board of Trustees of the University of Illinois.)
Server—Software responsible for making local documents or files available to other software systems.
World Wide Web (WWW)—A global information system providing hypertext-linked access to resources on the Internet. The WWW also incorporates existing network services, such as FTP and Gopher.

Our experience to date has been that AFS's local caching is critical to the success of the NCSA WWW server architecture. Stateless distributed file systems (for example, NFS) cannot exploit the locality inherent in the HTTP request stream by locally caching frequently requested items. Instead, they must repeatedly retrieve those items from a shared file server. Not only does this increase the load on the file server, it is inherently unscalable.

Despite the advantages of local caching, much research remains before we learn how distributed file systems in general, and AFS in particular, support large, less frequently accessed files (for example, 24-bit color images and digital video clips). With standard caching algorithms, access to these files will displace smaller, more frequently accessed files from the local cache. If these large, nontext files are not cached, their access latencies will be large. Data-type-specific caching algorithms are one potential solution.

Round-robin domain name system

The third and final component of the NCSA scalable WWW server is a modified network name resolver based on the Berkeley Internet Name Domain (BIND) code.[7] The existing BIND 4.9.2 code has a round-robin option that can associate a single domain name with several IP addresses. In response to requests, these addresses are distributed using a simple rotation algorithm. Because this rotation conflicted with extant software at NCSA, the BIND software was modified to rotate only specific addresses, namely those of the WWW servers (see Katz et al.[4] for details).

The modified domain name system (DNS) allows a domain name with more than one associated IP address to be specified as "round-robin." Each incoming request for the address of a round-robin domain name is satisfied by the next IP address on the list in a simple rotation. Thus, 1/*N*th

WWW server performance visualization

To gain insights into the large volume of access and performance data in the WWW logs, we relied on a variety of standard statistical data analysis tools. However, to understand the dynamics of server behavior and the interactions of request patterns with round-robin DNS system, we exploited the local availability of the CAVE, an immersive, unencumbered virtual environment, and our Avatar visualization software,[1] to create dynamic displays of server behavior. Figures A and B show snapshots of this visualization from a "day in the life" of the NCSA WWW server.

In Figure A, the trajectories of four different servers in the performance metric space are denoted by the four colored ribbons. This snapshot, from near noon on September 7, 1994, shows that the round robin DNS system effectively balances the server load—the trajectories of all the servers cluster in the same region of the performance metric space; the small variations are due to differing request patterns.

Figure B shows a global view of the origin of the requests. The height of the bar at each geographical location represents the number of bytes requested by that location; the different color segments represent the different data types. Figure B shows the activity at 6 p.m., local time, of a typical workday. Because of the time zone difference, most of the requests at this time are originating from the west coast of the United States. (The bar at the north pole represents sites that cannot be mapped to a specific geographical location.)

For details about the infrastructure of this visualization environment, see Reed et al.[1] in this issue of *Computer*.

Reference

1. D.A. Reed et al., "Virtual Reality and Parallel Systems Performance Analysis," *Computer*, Vol. 28, No. 11, Nov. 1995, p. 57-67.

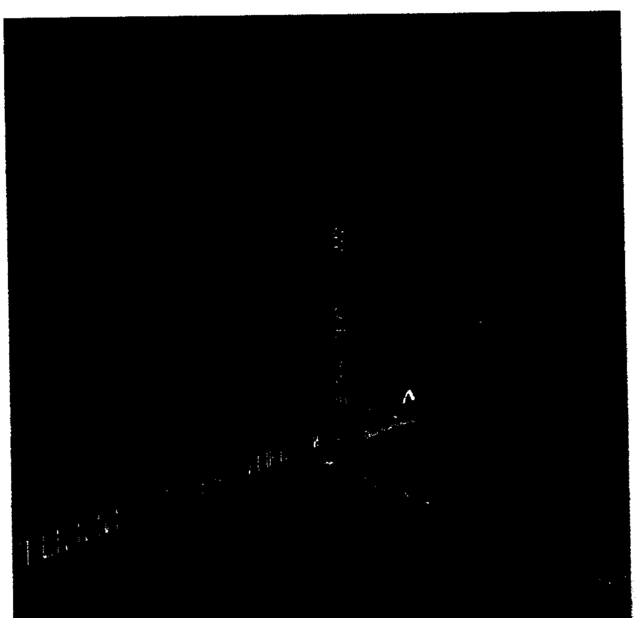

Figure A. WWW server visualization.

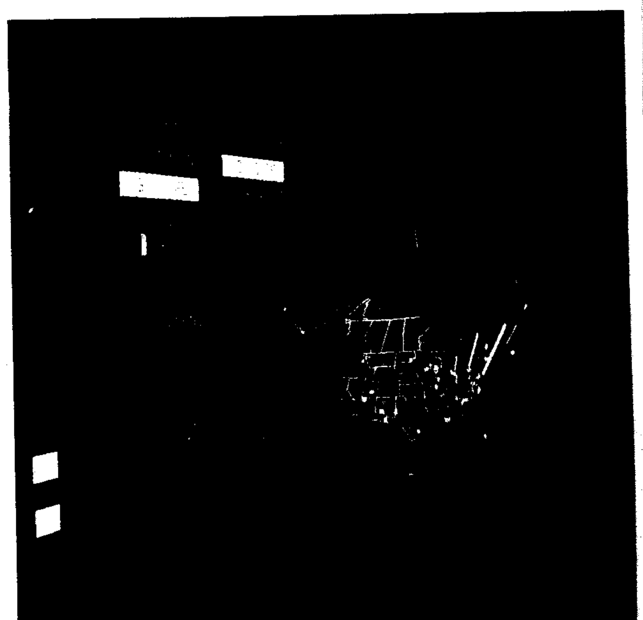

Figure B. Origin of WWW requests at 6 p.m. local time.

of the DNS requests get each of the *N* different IP addresses. This allows NCSA to maintain a group of WWW servers aliased by the single domain name www.ncsa.uiuc.edu. Adding a new server to the group is as simple as adding its IP address to the DNS entry for www.ncsa.uiuc.edu.

DATA COLLECTION

If the NCSA WWW service has accomplished nothing else, it has produced copious amounts of performance- and access-pattern data. This data is collected continuously on each server and is permanently archived each day to be available for researchers. Collectively, the files constitute more than 150 megabytes of data each weekday.[8]

On each of the component WWW servers, the data collected includes

- the standard access logs from the NCSA HTTP daemons (httpd),
- the standard error logs from the httpd daemons,
- a custom log of the client browser type (the "user agent") that initiated each request,
- a trace of virtual memory statistics, obtained by recording Unix *vmstat* data once each minute,
- a trace of packet counts, obtained by recording Unix *netstat* data once each minute, and
- a count of active processes, sampled with *ps* once every 5 minutes.

REQUEST PATTERN ANALYSIS

To understand the access pattern and characteristics of NCSA's WWW service, we analyzed the data described above for selected weeks during five different months of 1994. Below, we present the qualitative results with respect to the general access trends, the domain characteristics, and the file type distribution (see Kwan et al.[5] for details).

General trends

Qualitatively, WWW traffic growth on the Internet is well known. However, the specific characteristics of this growth and the sources of requests are much less well understood. Hence, the initial goal of our analysis was a simple characterization of WWW traffic in terms of request count, request data volume, and request sources (by hardware platform type).

TRAFFIC GROWTH. The number of requests received by the NCSA WWW servers during the period of our analysis grew from about 300,000 per day in May 1994 to about 500,000 per day in September. Thus, the compounded growth rate over the five-month period is roughly 14 percent per month. A scan of NCSA's January 1995 WWW server logs shows that the number of requests has increased to about 690,000 per day. As a result, the compounded growth rate is about 11 percent per month from May 1994 to January 1995. For the rest of 1995, however, the number of requests to the NCSA server have slowly decreased. (See File[2] for the latest 1995 statistics.)

CLIENT PLATFORMS. Knowing the platform from which a request originated has great potential value. Information providers can customize documents for different platforms, and servers can exploit this knowledge by tailoring their response to the hardware and software capabilities of the requesting platform.

The user agent logs from the first 20 days of December 1994 show that 31 percent of all connections were from X Windows clients, 38 percent from Microsoft Windows clients, 20 percent from Macintoshes, and 21 percent from all other types of clients. This data shows that at least 58 percent of the requests originate from personal computers. As vendors continue to ship new and improved versions of WWW browsers for personal computers, we expect requests from personal computers to grow at a very rapid rate. However, because of the relatively low bandwidth (modem) connections from most personal computers to the Internet, it is becoming increasingly important for WWW servers to adapt to client needs (for example, by sending lower resolution images) and for clients to prefetch selected data to hide the long latency for data retrieval.

Table 1. Server request origins by domain.

Internet domain	Percentage of requests
Education (edu)	26
Commercial (com)	18
Government (gov)	5
Others	51

Domain characteristics

Much discussion has centered on the commercial potential of the World Wide Web and the increasing accessibility of commercial information. To assess the number and distribution of commercial and other requesting sites, we aggregated domain names into a small number of broad categories: educational, commercial, government, and other. Table 1 summarizes the fraction of requests from the major Internet domains.

Although Table 1 shows that the edu domain generates more requests than any other single domain, Figure 2 shows that the number of requests from commercial domains is growing rapidly. (For each month, the figure shows seven data points, corresponding to Sunday through Saturday of the week we analyzed during that month.) This reflects the increasing presence of commer-

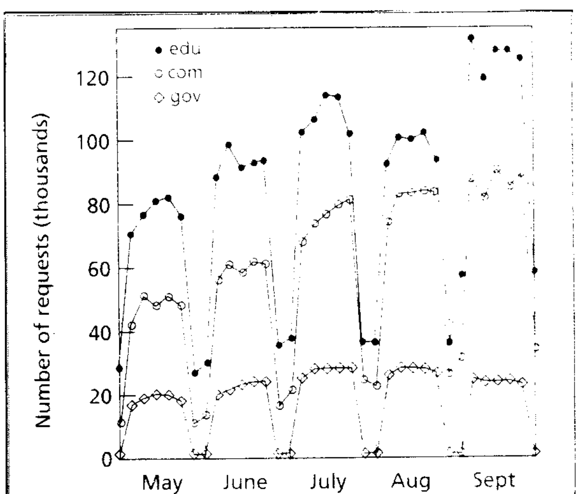

Figure 2. Weekly domain request statistics from May to September 1994. Each data point represents a Sunday-Saturday cycle analyzed during the month.

cial Internet service providers and the growing use of the Internet by the staff of commercial organizations.

Although the top 10 educational and government domains (which generate the largest number of requests to NCSA's server) change almost daily, the top 10 commercial domain names change little. Indeed, most of the top 10 commercial domain names on any given day were also among the top 10 domain names throughout the five months of data we analyzed.

The domain names in the com domain are mainly network firewalls for large organizations; they have long connection times and make an unusually large number of requests. Because a firewall acts as a central location for accessing data outside a given organization, it is the ideal location for implementing network caching and proxy servers, a topic to which we will return.

Media distributions

As we noted above, the request rate to the NCSA WWW server is growing at a compounded rate of between 11 and 14 percent per month. In addition to the rate, the characteristics of the growth have important implications for WWW server implementation. For example, satisfying large numbers of requests for small, text-based documents is much easier than responding to large numbers of requests for color images, video clips, or large data files.

Because the HTTPD server logs contain the name of the document being requested, and the file extension can be used to identify the document category, it is possible to determine the relative request frequency for text, images, audio, video, and data. The text category includes Hypertext Markup Language (HTML) documents, plain text, and postscript files; the image category includes GIF, X bitmap (xbm), JPEG, and RGB files; the audio category includes au, aiff, and aifc files; and the video category includes MPEG and QuickTime files.

Figure 3 shows that text and images account for the majority of the requests. Although audio and video account for only 1 percent of the requests, they represent 28 percent of the bytes transferred. The requests for large audio and video files also lead to more bursty data transfer rates. Interestingly, the temporal distribution of the requests for audio and video is skewed toward later in the day than the distribution of those for text and images. We conjecture that users seek off-peak times to retrieve large items from the server.

One should be chary about projecting access characteristics from this data. The NCSA WWW document tree is dominated by a large number of small objects. As WWW document repositories mature, we expect them to contain a much larger number of large scientific and technical data sets, scientific visualizations and video clips, and audio segments. This shift will accentuate the behavior found in this study: Many of the requests will be for small data items, but an increasing fraction of the data volume will be associated with requests for large, nontext items.

SERVER CACHING

To this point, our focus has been on the characteristics of the request stream. We turn now to an examination of the servers' "response" to the incoming request stream.

Effective, distributed file caching was one of the key design principles in NCSA's WWW server architecture. Local caching at the WWW servers reduces the load on the shared AFS file servers, minimizes file traffic on the FDDI ring, and allows the WWW servers to respond quickly to requests for frequently accessed documents. To measure the effectiveness of the current AFS caching protocols, we analyzed the WWW server logs to identify the characteristics of the most frequently requested documents.

As mentioned above, NCSA serves documents from the AFS distributed file system, which automatically caches the most recently used files in local AFS client caches. The left portion of Figure 4 shows the number of distinct files requested per day during the five months of our analysis; the right portion shows the total size of these same files.

Comparing the two figures shows that although the number of distinct files requested has increased, the total size of all the requested files has remained under 450 megabytes per day. Most of the newly added files have been small text and image files. To date, the AFS client cache hit ratios for the WWW servers have been near 90

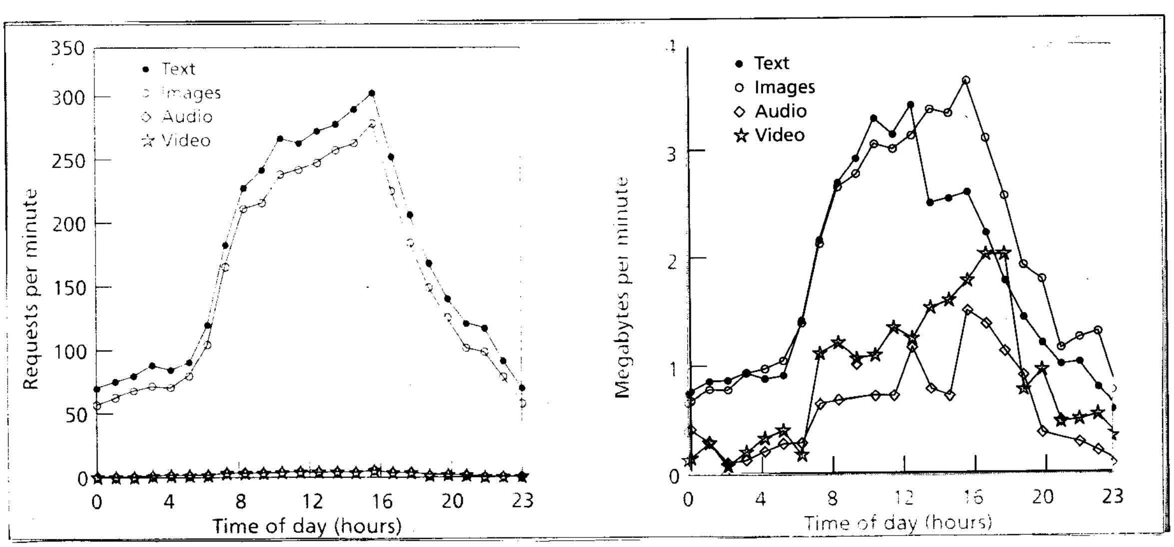

Figure 3. File type statistics by rate (left) and volume (right).

percent, suggesting that AFS caching has worked quite well for the past access patterns.

Note that not only does the AFS file system cache frequently accessed files on the local disk of the WWW servers, but also the most frequently accessed of those files are cached in the primary memory of the WWW servers. With the observed access patterns to NCSA's WWW servers, less than 60 megabytes of primary memory cache space is needed to satisfy 95 percent of all incoming requests, which corresponds to roughly 800 distinct files. Though most requests are small, a small number of requests retrieve large items. For this reason, satisfying 95 percent of the requests represents only 80 percent of the total data volume.

IMPLICATIONS

As the number of requests to NCSA's and other WWW servers continues to grow, the continued scalability of the server architecture, the efficiency of the HTTP protocol, and the effectiveness of caching strategies become increasingly critical research and implementation issues. Let's examine salient aspects of each issue.

Scalability and persistent state

Although round-robin DNS has allowed NCSA to add WWW servers without piercing the illusion that www.ncsa.uiuc.edu is a single host, the use of round-robin DNS is not an ideal solution to either the decoupling of logical WWW server names from the physical server identity or to request load balancing. With this approach, the distribution of WWW server addresses is divorced from the characteristics and load of the constituent servers.

While the round-robin mechanism equally distributes the IP addresses of the constituent servers, there is no mechanism to limit the number of times an address is used after it is distributed, or to guarantee that the client system will honor the advertized time to live (TTL). For instance, a local DNS service might distribute a single IP address to any number of clients in its domain.

Moreover, envisioned extensions to HTTP include long-lasting state (for example, the results of previous database searches) that must be retained by a WWW server. Supporting such extensions may be difficult for a multi-server architecture that relies on round-robin DNS. A second request may be sent to a different server than the one holding the result of the previous request. Unless the data is shared (for example, via AFS), obtaining the requisite information will require closer server cooperation, with associated overhead.

HTTP protocol extensions

The overriding trend from our data analysis is the continued growth in request rate. Currently, each request from the client uses a separate TCP connection, and the large number of short-lived TCP connections limits the performance of the server. This problem is exacerbated by the fact that a document may be composed of several pieces, each of which is fetched separately, with each fetch requiring a separate TCP/IP connection. Padmanabhan and Mogul[9] have proposed opening a single TCP connection per HTML document to avoid unnecessary TCP overhead; preliminary experiments show that this reduces document retrieval latency. Spero[10] has proposed a new protocol, HTTP-NG, which dramatically alters HTTP to reduce overhead, allow more parallelism, and efficiently support features such as authentication.

These and related protocol changes will reduce the latency to deliver data and transmit more data over each TCP/IP connection. It will make HTTP servers much more like FTP and other session-oriented services. This may well make much better use of the available network bandwidth and other server resources.

Distributed caching and prefetching

Beyond reducing the network protocol overhead, one can also aggressively cache and prefetch the data. At the moment, various browsers cache data on local client disks to improve performance. Pitkow and Recker[11] have shown that caching based on recent rates of past access is an effective technique. However, to design and implement effective prefetching, one must first study and understand the

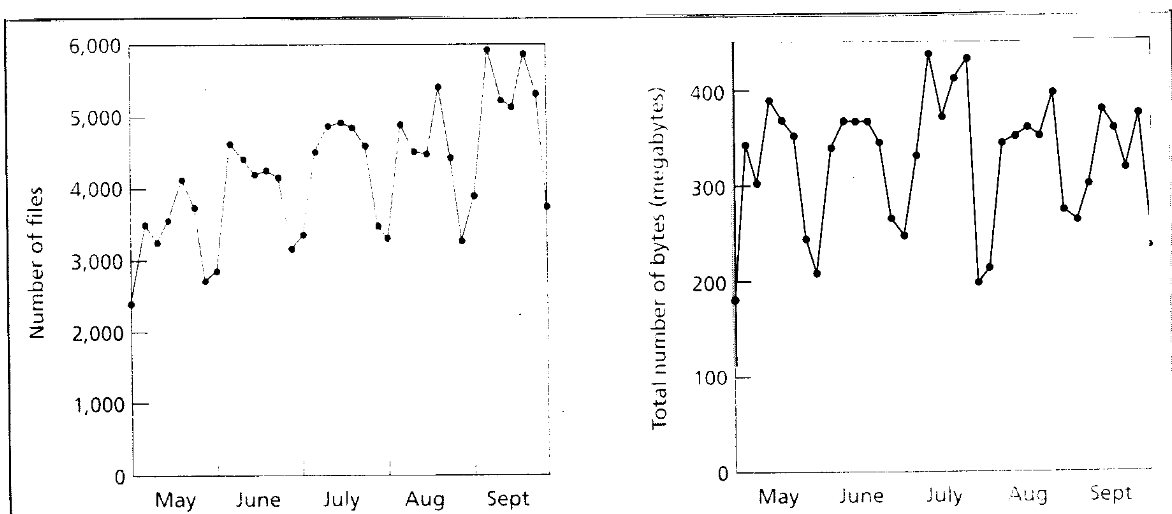

Figure 4. Request profile: number of distinct files requested (left) and total size of all files requested (right). Each data point represents a Sunday-Saturday cycle analyzed during that month.

extant access patterns. Our data suggests that partitioned caches are a promising alternative. However, prototype implementations and trace-driven simulations are needed to measure the performance benefits that might accrue from this approach.

We noted that the most prolific sites are all commercial gateways. Moreover, about 2 percent of the requests to the NCSA WWW servers are from hosts that make only one request. The most popular of these requests are to the "directory" pages, namely the NCSA Internet Starting Points, the Internet Resources Meta-Index, and the What's New pages. These pages are excellent candidates for replication and caching throughout the Internet, particularly at commercial gateways.

In the future, as audio and video clips play a larger role in conveying multimedia information, audio and video requests will significantly affect network traffic and caching strategies. As we have seen, even a small increase in the use of these data types will dramatically increase the amount of data to be read and transmitted, with a concomitant deleterious effect on the efficiency of server caching strategies.

WE HAVE DESCRIBED THE DESIGN OF NCSA'S WWW SERVER and analyzed the access patterns to the server in terms of the user request patterns and the responses of the server. The analysis shows that scalability, protocol efficiency, and effective caching strategies are the major issues for the next generation of WWW servers. In particular, we believe that to improve performance, both clients and servers must aggressively exploit caching and prefetching on the basis of knowledge of request patterns, data types, and hardware capabilities. ■

Acknowledgments

Our thanks to Eric Katz for providing us with the initial log analysis scripts, to Nancy Yeager, Michelle Butler, and Paul Zawada for providing crucial assistance in understanding the NCSA WWW server, and to Charlie Catlett, without whom this work would not have been possible. Finally, thanks to Will Scullin and Steve Lamm for developing the virtual reality software used to display dynamic server behavior.

Thomas Kwan is supported in part by the National Science Foundation and the Advanced Research Projects Agency under Cooperative Agreement NCR-8919038 with the Corporation for National Research Initiatives. Robert McGrath is supported in part by the National Science Foundation, the Advanced Research Projects Agency, corporate partners, and the state and University of Illinois. Daniel Reed is supported in part by the National Science Foundation under grants NSF IRI 92-12976 and NSF CDA94-01124, by the National Aeronautics and Space Administration under Contract NAG-1-613, and by the Advanced Research Projects Agency under Contracts DABT63-91-C-0004 and DABT63-93-C-0040.

References

1. T. Berners-Lee et al., "The World-Wide Web," *Comm. ACM*, Vol. 37, No. 8, Aug. 1994, pp. 76-82.
2. J. File, *Web Server Activity: Total Connections*, 1994 (http://www.ncsa.uiuc.edu/SDG/Presentations/Stats/WebServer.html).
3. M. Satyanarayanan, "Scalable, Secure, and Highly Available Distributed File Access," *Computer*, Vol. 23, No. 5, May 1990, pp. 9-21.
4. E.D. Katz, M. Butler, and R. McGrath, "A Scalable HTTP Server: The NCSA Prototype," *Computer Networks and ISDN Systems*, Vol. 27, 1994, pp. 155-164.
5. T.T. Kwan, R.E. McGrath, and D.A. Reed, "User Access Patterns to NCSA's World Wide Web Server," Tech. Report UIUCDCS-R-95-1934, Dept. Computer Science, Univ. of Illinois, Urbana-Champaign, Feb. 1995.
6. *NCSA AFS Users Guide*, National Center for Supercomputing Applications, Univ. of Illinois, Urbana-Champaign, 1994 (http://www.ncsa.uiuc.edu/Pubs/UserGuides/AFS-Guide/AFSv2.100.html).
7. P. Albitz and C. Liu, *DNS and BIND in a Nutshell*, O'Reilly and Associates, Sebastopol, Calif., 1992.
8. R.E. McGrath, *What We Do and Don't Know About the Load on the NCSA WWW Server*, Sept. 1994 (http://www.ncsa.uiuc.edu/InformationServers/Colloquia/28.Sep.94/Begin.html).
9. V.N. Padmanabhan and J.C. Mogul, "Improving HTTP Latency," *Proc. Second Int'l WWW Conf.*, 1994, pp. 995-1,005 (http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/mogul/HTTPLatency.html).
10. S. Spero, *Progress on HTTP-NG*, 1994 (http://www7.cern.ch/hypertext/WWW/Protocols/ HTTP-NG/http-ng-status.html).
11. J.E. Pitkow and M.M. Recker, "A Simple Yet Robust Caching Algorithm Based on Dynamic Access Patterns," *Proc. Second Int'l WWW Conf.*, 1994, pp. 1,039-1,046 (http://www.ncsa.uiuc.edu/SDG/IT94/Proceedings/DDay/pitkow/caching.html).

Thomas Kwan *is a doctoral candidate in the Department of Computer Science at the University of Illinois at Urbana-Champaign and a graduate research assistant at the National Center for Supercomputing Applications. His research interests include parallel computing, gigabit applications, and World Wide Web technology. He received a BS degree in electrical engineering from the University of Washington and an MS degree in computer science from the University of Illinois at Urbana-Champaign. He is a member of IEEE, ACM, and Tau Beta Pi.*

Robert McGrath *is a research programmer at the National Center for Supercomputing Applications. His research centers on the architecture and performance of large-scale distributed systems. He is a coauthor of the book* Web Server Technology *to be published by Morgan Kaufmann in 1996.*

Daniel A. Reed *is a professor in the Department of Computer Science at the University of Illinois at Urbana-Champaign, where he holds a joint appointment with the National Center for Supercomputing Applications. Reed received his BS degree in computer science from the University of Missouri at Rolla in 1978 and his MS and PhD degrees, also in computer science, from Purdue University in 1980 and 1983, respectively. He was a recipient of the 1987 National Science Foundation Presidential Young Investigator Award.*

Readers can contact the authors at the University of Illinois at Urbana-Champaign; e-mail {tkwan,mcgrath}@ncsa.uiuc.edu and reed@cs.uiuc.edu.