Scalable Internet Servers: Issues and Challenges

Krishna Kant
Intel Corporation[*]

Prasant Mohapatra
Michigan state University[†]

1 Introduction

There has been an exponential growth in the Internet usage, and a lot of it is attributed to the increasing popularity of the World Wide Web (WWW). This increasing demand overburdens the bandwidth requirement in the Internet, particularly at the edges and the last-mile. Similarly, the processing demand and resource management for Internet applications have been increasing the load on Internet servers (web servers, proxy/caching servers, firewalls, etc.) rapidly. Evolving applications like continuous media need high throughput, whereas e-commerce transactions need low response time at all times, even during congested periods. Furthermore, the need for service differentiation is becoming more critical in the evolving applications. Thus, Quality of Service (QoS) support through service differentiation and load management are the two main issues that are receiving a lot of attention in the research community related to web servers.

The next generation of web services would need to support a very high rate of requests with varying needs. Thus the scalability of the current model will impose a big challenge both for the networking infrastructure and the server environment. In this paper, we have focussed only on the server-level issues.

2 Major Challenges

To satisfy the scalability of internet services, the internet research community is likely to face a number of challenges at both the network and server side. With the network bandwidth increasing much faster than the server capacity, we believe that more and more the bottlenecks will be observed on the server side. The purpose of this article is to identify the major driving forces for server-side bottlenecks and the major challenges that need to be addressed to scale up internet server capacity with networking capacity and service demands.

[*]Mail stop CO3-202, 15220 NW Greenbriar Parkway, Beaverton, OR 97006, Tel: (503)677-6939, Email: krishna.kant@.intel.com

[†]Department of Computer Science and Engineering, East Lansing, MI 48824, Tel: (517)355-8389, Email: prasant@cse.msu.edu

As a point of reference to this discussion, let us consider some numbers. As reported by several PAWS-2000 panelists, the web-site hit rate at major events is already approaching 1 million per minute. As Internet usage skyrockets throughout the world, this number could easily increase by more than an order of magnitude for major world-wide events such as the Olympics. However, the hit count doesn't tell the entire story — as the richness, perceptual quality, dynamism, and security requirements of the content increase, the size of the web-pages and the processing time required to render them could itself increase by an order of magnitude. For example, our study shows that the use of HTTPS to retrieve static web pages could increase processing requirements by an order of magnitude. Thus, even from a rather short-term perspective, we need to be able to design Internet servers that can handle sustained 10 million hits/minute with 10 times the bandwidth and processing requirements of today's servers. At the same time, the increasing criticality of Internet applications demands that the servers exhibit a very high level of robustness and availability while delivering transaction response times under a few hundred milliseconds with a high probability. In the following paragraphs, we list the major technical problems that need to be solved to enable this magnitude of scaling.

1. *Keeping up with LAN speed improvements:*

   Early experience with Gigabit ethernet showed numerous difficulties in actually supporting close to 1 Gb/sec throughput on the server because of inefficiencies in the chip-set, protocol stack (TCP/IP), and the O/S. Now, with 10 Gb ethernet around the corner, all these issues need to be revisited. And this time, simple tweaks to current practice may not be adequate; instead, it may be necessary to explore new ideas that would break the fundamental bottlenecks. In particular, a comprehensive solution demands (a) a more distributed server architecture that cleanly divides the processing layers (e.g., application, session, transport, etc.), (b) lightweight, user-level I/O (both disk and network) that avoids the O/S bottleneck, (c) peer-to-peer I/O with distributed intelligence (e.g., a direct flow of encrypted data from disk to the network interface card (NIC) without any CPU interference), (d) hardware protocol implementations, etc.

2. *Dealing with increasing dynamic content:*

   Caching servers (whether using the traditional "pull" model, or the "push" model of reverse proxies) along with the inter-cache communication protocols such as ICP have greatly enhanced user experience by reducing latency and load on the native server. However, the increasing dynamic content of web pages poses a major challenge for this technology. Some recent work has concentrated on defining structure for dynamic web pages so that changes could be tracked for individual components. Such a paradigm needs to be extended and standardized so that it allows not only the caching but more general functionality such as the insertion of dynamic content at various levels of caching hierarchy perhaps under the control of the native server. Such a structure supports more aggressive pushing of contents to edge servers and content transformation proxies needed to serve wireless and wireline appliances with a wide variety of bandwidth needs that are expected to proliferate in near future. Many of these capabilities would require enhancements to established protocols (e.g., HTTP, ICP) and markup languages (HTML, XML).

   A critical issue in dynamic content creation is the efficiency of the web application software. For example, even the fast CGI is too slow, whereas ISAPI implementations tend to be memory and memory bandwidth hogs. Novel dynamic content management techniques are needed in order to significantly cut down this overhead.

3. *Scaling up a web-server to huge content and hit rates:*

   The traditional method of scaling web-servers has been to do load-balancing across a server farm using a front-end load distributor. Such a solution is inherently non-scalable. If the front-end operates at transport layer or below, scalability suffers because of duplication of contents in the main-memory cache of each server. On the other hand, a content-aware load distributor itself becomes a bottleneck because of the amount of work expected of it. To resolve this issue we need innovative solutions that involve greater content awareness closer to the client for on-demand requests, perhaps by using functionality similar to service control points in telecommunications systems. Since the push-model automatically solves this problem, innovative pushing methods that reduce the pull frequency can be considered as part of the solution.

   Large clustered servers reduce the number of independent servers needed in the server-farm, and thus may alleviate the scaling issues. However, this brings in scalability issues for clusters. Lightweight communication protocols such as Virtual Interface Architecture (VIA) implemented over scalable system area networks (such as the emerging Infiniband architecture) provide promising scalability solutions; however, considerable challenges remain in many areas including functional partitioning, flow-management, quality of service, load management, data partitioning, I/O management, etc.

4. *Support for distributed internet services:*

   The current Internet server model is that of a centralized place (or "server") that acts as repository of valuable information identified to the client by its location. Such a model is inherently non-scalable. In a true global Internet economy, every connected computer can potentially be both a producer and a consumer of information. Centralizing this information is perhaps an important evolutionary step, but cannot be considered a viable end game in many environments. However, making a giant leap from current systems to a fully distributed environment brings in enormous technical hurdles in a multitude of areas including security, access control, information management, robustness, concurrency control, resource management, resource usage control, latency management, etc., in addition to the larger societal issues of intellectual property rights and business models.

   Viewing the Internet infrastructure as a set of servers hosting valuable data does not address the real client need of finding the desired "service" (as opposed to "data"). For example, as a client, I want to buy an airline ticket with some given attributes (e.g., lowest cost, specific airline, etc.); the data that makes this possible (or even the location of such data) is of no interest to me. This change in focus from "data" to "services" has serious implications in how data is identified, accessed and transferred. For example, data should be identified by its attributes, not by location. Similarly, the efficiency of access is measured in term of how efficiently common services can be supported, instead of how efficiently a given type of data can be accessed.

   An effective implementation of distributed services may require considerable enhancements to existing protocols in various areas of concern. For example, the support for peer to peer interaction and multiple virtual channels per TCP connection in BXXP can be exploited by the capabilities mentioned above.

5. *Management and engineering of large servers:*

   The telecommunications systems are supported by well-developed engineering (or capacity planning) procedures and comprehensive operations support systems (OSS). Such a support is virtually nonexistent for web-servers. As the complexity, volume and criticality of the traffic handled by the web-servers increases, it is important to develop good engineering practices supported by adequate traffic characterization. This issue is complicated by the highly bursty (self-similar and perhaps multifractal) nature of internet traffic, proliferation of heavy tailed distributions, traffic non-stationarity over even rather small time intervals, hard to predict growth patterns, poorly understood relationships between access growth vs. data size growth, and the complexity of estimating processing requirements for the dynamic component of web-pages. With server farms sporting thousands of servers, efficient system management becomes daunting problem in every area including fault and performance management, graceful recovery from massive failures, partial data replication for scalability and high availability, zero-downtime software and hardware upgrades, service level agreement (SLA) management, etc. The real challenge is to find management solutions that scale with number of servers, i.e., the complexity of management must increase significantly slower than linear with the number of servers.

6. *Architectural and I/O Issues:*

   Much of the literature on server performance considers only application and protocol issues and does not explicitly model the influence of O/S and hardware on application level performance. As the processing and I/O requirements for servers increase, the architectural issues could become dominant and prevent scalability. For example, a lack of proper coordination between different layers in packet processing and use of improper event recognition techniques may lead to unexpected bottlenecks in the system. On many commercial workloads, current multiprocessor systems show poor performance scaling beyond 4 or 8-way systems because of processors spinning on locks for shared data structures or excessive coherence traffic on the processor bus. The O/S supported I/O typically requires memory to memory copies which burns processor cycles, increases memory and bus bandwidth requirements, and most importantly, pollutes processor caches with one-touch accesses. In other respects also, network protocol processing is often not cache-friendly and may lead to poor cache-acess locality, high cache coherence traffic, and hence CPU stalls. A yet another issue is getting dirty data out of the cache on to the I/O bus to keep up with high speed NICs. As more demands are placed on servers, it is important to not only find solutions to these problems, but to also incorporate the impact of architectural features in application level performance models. In particular, we need to move away from queuing models where the processing subsystem is represented by a simple multiserver station with no consideration given to the impact of cache misses, coherency traffic, processor bus queuing and stalls, memory pipeline queuing, etc.

   As the usage for streaming audio and video increases, the servers must be able to support in excess of 10,000 concurrent streams. This capability will require a number of innovations including efficient caching of streams (e.g., interval caching), minimize CPU involvement in I/O, and efficient data streaming by the chipset. Efficient I/O becomes more critical in the context of HTTP access as well as embedded requests for very small files proliferate. In fact, with increasing LAN/WAN bandwidth, the amount of data sent per request (or bytes/sec) may become almost irrelevant compared with the sustained I/Os per second that must be fielded.

7. *Quality of Service Issues:*

   There has also been an increase in the traffic types in the Internet, which in turn has been necessitating a need for service quality, both at the network level as well as the server level. Examples of a few pairs of traffic types that can benefit from service differentiation include discrete media (text and images) and continuous media (audio and video), real-time and non-real-time, small and large files, static and dynamic pages, secure and insecure transactions, revenue generating and non-revenue generating requests. It is thus essential to provide support for service differentiation with different levels of QoS to different request types. Work in this direction is already in progress in the networking domain. However, very little effort has been made on the server side for providing differentiated services. The possible adoption of QoS support in the next-generation Internet further adds to the necessity of providing the support also at the server level. It may be noted here that implementing differentiated services only in the network could lead to pathological situations where a lot of high priority traffic is carried by the network only to be dropped at the server.

8. *Scaling and performance issues for back-end systems:*

   Much of research on Internet servers has concentrated on front-end (usually HTTP) servers; however, with exploding e-commerce, the attention needs to shift to the entire system, which may include middleware (search engines, shopping cart, LDAP servers, etc.) and the back-end database systems. Scaling issues for these systems are much harder to deal with than for web-servers because the business logic often involves the entire data repository (unlike web-servers that only deal with a few files at a time). Most of the issues addressed here with reference to web-servers apply to middleware and back-end systems; they only become harder to address because of inherent dependencies in data access. Thus it may be worthwhile to explore distributed (possibly partially replicated) databases or other models of data acessing environments.

9. *Internet server overload control:*

   The throughput of the server and the response time experienced by the clients are two important measures that characterize the performance of Internet servers. The throughput of a server increases with the increase in the load upto a certain level, which we call as overload point, beyond which it starts declining rapidly. At the same time the response time also becomes unbearable. By using efficient overload control mechanisms, the load on the servers can be controlled such that it never exceeds the overload point. A good overload scheme can maintain throughput close to optimum while also providing good response time to the clients. An efficient overload control implementation requires ability to classify (and drop, if necessary) packets at the lowest possible level of the protocol stack, ideally at the NIC itself. Intelligent NICs could be used for this purpose, which are begining to be cost-effective.

   Overload control is very well developed and even standardized in telecommunications systems, but its use in web servers is almost nonexistent. Some recent work has shown the promise of a direct, feedback based overload control; however, several challenging issues remain, including dealing with connection-oriented nature of HTTP, devising feedback and load-shedding mechanism that effectively exploit fractal behavior of Internet traffic, exploring guidelines for setting control parameters, accounting for middle and back-end loading in overload control, and extending the current schemes to effectively deal with focussed overload and distributed denial of service related overload scenarios.

10. *Performance issues for secure transactions:*

    The recent explosion in the use of HTTPS to provide security in e-commerce transactions has exposed the high processing overhead of SSL and consequent very slow web server response. It has been estimated that the consequent user abandonments and refrainment from e-business cost the e-commerce industry $1.9 billion annually in lost revenues. Easing the SSL bottleneck touches upon all of the areas included in the topics list above. The expected proliferation of IPSec capable NICs will also introduce some of these issues, but at a lower level in the network stack. The combined use of SSL and IPSec also brings in a host of new issues (e.g., overload control at multiple levels of stack) that need to be addressed. Also, if the same server is handling both secure and nonsecure transactions, secure transactions may need better quality of service as they are linked with the generation of revenue. On the other hand, if SSL is handled by acceleration hardware or dedicated servers, SSL related overload (perhaps due to fluctuations in the percentage of secure transactions) needs to be addressed carefully. Since security conflicts with caching, it is important to devise new mechanisms that provide a tradeoff between security and cacheability. For example, partially exposed URLs and/or caching hints could be exploited for caching secure content.

11. *Server performance data repository:*

    Much of the work on web-servers is based on HTTP logs and system activity measurements collected from the servers. Addressing architectural issues requires even more detailed information (I/O bus traces, instruction traces, processor bus traces, lock markers, etc.) The entire Internet-server community, whether working on research issues or developing capacity planning solutions, will benefit immensely from establishing a repository of such information at least for the major events. However, pulling off such an effort is a major challenge in many ways: (a) development of standardized encoding schemes for HTTP, instruction, and bus traces that keep nearly all the behavioral information intact without revealing any specific private or proprietary information, (b) developing capability to record such information from live sites at their busiest moments with no performance or robustness impact, and (c) coordinating an industry-wide effort to encourage contribution to the repository while addressing concerns of potential misuse or litigation. The recent availability of 1998 World cup HTTP server logs is a step in the right direction.

    It may be noted in this regard that even the extended HTTP log format is lacking in many respects and hinders detailed traffic characterization. For example, the current 1 second granularity for request times is highly inadequate; instead, it should be possible to provide time at scheduler granularity. Additional data such as the total size of the requested web-page (instead of just the transferred part), CPU time taken to process the request, etc., would enhance utility of HTTP logs considerably. The challenge here is to standardize additional capabilities and persudaing site operators to actually collect the more detailed data.

All of the issues mentioned above are important and need attention of researchers and developers to enable the scaling of Internet servers to meet the future demands. Almost certainly, additional issues and challenges will come up as we ride through the unprecedented growth of the Internet.