NETSEDGE RESEARCH GROUP

The Evolution of Traffic Management Systems

Peter Christy, NetsEdge Research Group, September 2001

In the last five years traffic management systems have grown from obscure and exotic technology to a standard part of server configurations. Traffic management systems are a distinct blend of network and information processing, neither a network device nor a server, but something in-between, requiring purpose built hardware and software systems.

Over the next five years, the Internet will utterly transform business by connecting an enterprise to customers, prospects, employees, suppliers, business partners, as well as the public at large. The Internet is the basis for that universal access, carrying your business presence, at electronic speed, anywhere, worldwide. With that opportunity comes a set of challenges and obligations, not the least of which is the implementation and operation of server systems capable of providing high-performance, reliable, interactive response for the large and unpredictable access loads that the Internet is capable of delivering. Building effective server systems and applications is a critical part of exploiting the potential of the Internet. Traffic management systems are an increasingly important part of those servers and applications.

Traffic management systems sit between the network and the servers and function to direct traffic, at line speeds, from the Internet to the individual systems and appliances (e.g., firewalls) that collectively constitute the "server." Traffic management systems grew out of relatively straightforward need to scale server capacity beyond the limits of the largest, standard multi-processor configurations. Subsequently, many innovative applications have been developed, and this system technology has significantly influenced the much broader topics of server system and application design.

The threads that come together in this investigation are these:

- Scaling servers
- Optimizing resources
- Making more and more sophisticated application decisions based on network traffic content
- Building more secure and trusted Web systems
- Integrating these traffic management capabilities with the application platform.

The Origins of Traffic Management

Traffic management systems grew out of necessity. The Internet provided universal, world-wide connectivity to Web sites, presenting the opportunity/challenge of system loads of an unheard scale. In the earlier era of mainframes and timesharing, attaching

users to the system was much more difficult (naturally bounding the load) and it was possible to spend more money and purchase a larger and faster computer. When use of the Web started to explode in the mid 1990's, the potential loads were essentially unbounded compared to previous systems, and it was no longer possible to easily buy a faster computer – the PC you bought your teen age son to play video games is about as powerful a CPU as you can buy for a server application. When Netscape launched NetCenter, and immediately exceeded the capacity of the high-end SGI multiprocessor servers they had purchased, they had little option but to buy dozens of PC servers and cobble together some traffic management scheme to keep the site going and growing.

The first traffic managers were done with interesting operating system level code (e.g. how the IBM Olympic Websites distributed traffic). F5 Networks and Cisco introduced BIG-IP and Local Director as two of the earliest commercial products – essentially special purpose application software running on a standard PC hardware platform. Then high-speed network switch vendors entered the fray by greatly expanding the control processing that those devices did. Over time a distinct product category has emerged blending this all together.

Why Is Traffic Management Such An Important Function?

The Internet has changed the requirements on servers quite dramatically. When the Web was created a decade ago, the vision was of many small computers integrated through a communications fabric. In somewhat startling contrast, the Internet has re-energized the use of "large" centralized computers, reminiscent of the mainframes and time-sharing systems of 20+ years ago. There are many advantages delivering an information service or function from one, centralized location, but the potential loads on these systems from world-wide, Internet connected computers, is vast. The potential demand has far outstripped improvements in the performance of processors (the infamous "Moore's Law"). Although many applications fit nicely on a single PC server, an important subset requires more than a single computer, ranging from a few up to the approximately 10,000 servers that collectively delivery Google search services. Traffic management systems have become a standard part of building these larger servers and the applications they deliver. And even at the small scale, traffic management systems provide the most effective way of dealing with the software reliability issues of complex, web information systems – hot, active backup systems.

The future traffic management story is even more interesting. The Internet requires large server systems. Reality requires that they be built as composite systems from multiple smaller computers. As more servers are implemented as composite designs, the individual computer systems can be optimized to the function they serve. We already have very highly optimized firewall systems, Web servers, and database servers, purpose-engineered to do a specific function well. We haven't seen the end of the co-evolution of system architecture and traffic management by any stretch of the imagination.

Simplistically, What A Traffic Manager Does

The way TCP/IP networking works is that one computer asks for a connection to another computer referring to the network address (the IP number) of the destination. The community of routers directs that request (in the form of a data packet) across the Internet, ultimately sending it to the target computer where software decides whether or not to accept the connection (after which the real data flow begins). If www.yahoo.com is really 100 computers, not one, the traffic manager sits in front of those 100 computers, and accepts connection requests to www.yahoo.com. Subsequently, the traffic manager sends the traffic alternatively to one of the 100 computers ("round robin load balancing") or uses much more complex traffic distribution algorithms. Once the destination server has been selected, the traffic manager opens a new connection to the server (necessary since the client connection has already been accepted) and then "splices" those two connections together, getting out of the way as much as possible.

At the simplest, a traffic manager simply balances connections to a target IP number. At the next level of sophistication, the balancing is done in part based on the category of the content. By convention, different forms of traffic are identified by making connections to different "ports" on the destination computer (e.g. a request for Web content is typically to Port 80). Or, in a Web request traffic may be further differentiated by the file type of the requested content (e.g. all JPEG images may be stored on specific servers), or by the application for which specific traffic is destined, or in the ultimate, by sophisticated pattern analysis on the contents of the data packets sent (content directed traffic routing).

What is common to all of these traffic management uses is that they are all complex software applications with very demanding network I/O requirements. A traffic manager directing traffic to 20 Windows/2000 servers handles 20 times the network traffic of any of those servers, obviously a very challenging demand. Although the first load balancers were built on standard computer platforms, they all had carefully tuned operating systems and network stacks capable of high speed network processing. As the application demands grows, more and more processing power is required (e.g. searching for patterns in messages at wire speed).

In summary, although a traffic server seems to play a simple role between the network and computer systems, implementing the functions can require both high speed network processing and enormous content processing, all in one box, at one time.

What A Traffic Manager Does – Server Scaling

As a load balancer, a traffic manager performs an apparently straightforward function of distributing incoming traffic among more than one server. The first load balancers distributed traffic in a simple, round robin fashion. There are many ways in which the operation can be embellished. Load can be distributed, for example, on the basic of open connections to that server (the fewer the better – find the free ones), or on some measure of responsiveness (the faster the better – find the lightly loaded ones).

What A Traffic Manager Does – Resource Optimization

A traffic manager can of course do much more. With only a little inspection of the message contents, it can determine the port for which the connection is destined. By convention, different protocol traffic (all riding on TCP/IP transport) is sent to different port numbers (requests for HTTP content typically go to Port 80). With a traffic manager in place, traffic destined to a single, logical server (a single IP number) can by triaged and distributed among multiple, functionally-distinct servers. Web traffic can go to one set of servers. Secure (SSL) traffic can be sent to others. File access requests (FTP) sent to yet another set of servers.

In a different dimension, traffic managers can be programmed to operate view what might be called a "Marxist" policy, sending traffic to servers based in-large on their ability to the work. In a farm, there may be a set of servers, purchased at different times, of different sizes, incorporating different technologies. An intelligent traffic manager in front can utilize each fully and optimally, even though the capabilities are quite different.

What A Traffic Manager Does – Content Based Optimization

With a little bit of content examination, we can differentiate traffic on the basis of flow, or incoming IP address, or the like. With enough computational power, we can use any part of the data packet payload. Many of the interesting applications don't require examining all the content. In an HTTP request, content is organized and referenced as a file hierarchy; often the first level of the reference is enough to allow the traffic manager to send the request to one of a set of functionally-optimized servers.

Cookies can be used as a key element in traffic management. Cookies are data items sent first from a Web site to a browser and then returned by the browser as an automatic part of a subsequent content request to the site. Cookies are a way by which sites implement automatic login or returning customer identification. They can also be used in a larger site with a traffic manager to assure that a customer is directed to a single Web server during a transaction.

In the extreme, a traffic manager can do full content-directed packet switching, sending requests to destinations based on sophisticated analysis of the content of the message (in contrast to TCP/IP networking that makes connections between specific computer systems). In the context of the Internet, directing messages by content, rather than to specific servers seems like the right big-picture direction for the future.

What A Traffic Manager Does – Security Management

The Web and the Internet are ultimately very insecure environments, not surprisingly considering they were designed 30 years ago, by researchers, for researchers. As uses evolve from the public dissemination of information to private communication or business interaction, some security overlay must be added. The most common approach is the use of SSL (the Secure Sockets Layer) which utilizes public key exchange and

robust, standard encryption to implement private Web-oriented communication over the public Internet.

Traffic management necessarily interacts with SSL security because the use of encryption blinds the Traffic Manager to all but the grossest aspects of the communication (that's the intent – make it hard to see any aspect of the traffic if you're not the intended recipient). Implementing the SSL operations as a Traffic Management capability in front of the servers has the dual benefit of offloading the processing load of encryption and authentication from the servers, and of returning the content to a "clear text" form suitable for full traffic management.

There are additional benefits to evolving authentication and encryption management away from the servers onto the traffic management system. The nature of computer security is very much that it is only as strong as the weakest link. Running an effective authentication and key management system is at best difficult and significantly more difficult across a heterogeneous complex of servers. Running SSL and authentication functionality from the Traffic Management platform is often at attractive, simpler alternative.

Traffic management devices also offer significant benefit as an element of defense against various denial-of-service attacks. Because the traffic manager is purpose engineered and built for high-speed, intelligent network processing, it is ideally suited to defend actively against these network intrusions and isolate the impact from the servers.

What A Traffic Manager Does – Application Integration

As the function of Traffic Management systems becomes more complex, there become more and more reasons to want to integrate the operation of the Traffic Management system into the management of the overall application suite:

- Conveniently take new servers into our out of the mix.
- Conveniently change the parameters of server selection, for example in the face of unusually high load.
- Coordinate traffic management activities during application or content upgrade activities.
- Direct traffic more intelligently based on specific application health or performance metrics provided directly by the application code.

Integration and coordination of traffic management devices has been possible from the beginning by using the operational interfaces provided, typically a TELNET connection to the command line interface of the traffic management system (analogous to how routers are configured). More recently some traffic management vendors are evolving those interfaces using XML data structures and transactions in place of *ad hoc* command line interaction. The potential value of using XML is huge, since it establishes standard format and processes for these command-and-control transactions, and keeps the device

coordination from breaking due to minor changes in the command line interaction details. Additionally, as is typical in XML efforts, open, multi-vendor standards are being proposed that hold the potential for establishing standards for traffic management integration that will span the offerings of multiple vendors, clearly the sign of a maturing, important part of these computer systems.

Scaling Traffic Management Systems

The earliest Traffic Management systems were carefully crafted software applications running on standard PC server hardware. These systems were quite sufficient for front-ending reasonable collections of servers (tens) and modest content inspection tasks. Other early TM systems were adapted network switches, capable of line-speed forwarding of large amounts of traffic but with limited capabilities for application programming or content inspection.

There are three separate forces that drive the need to build larger, more powerful Traffic Management systems:

- The need to build larger server systems to service more on-line users or deliver more complex application systems.
- The need to perform more complex Traffic Management applications or do more sophisticated wire-speed content analysis.
- The desire to place more server systems behind a common traffic management system in order to gain cost efficiencies in hardware and significant savings in terms of operational cost and overall cost of ownership.

Put together, these forces for scaling suggest strongly a need for more specialized hardware platform architectures capable of delivering more network processing than a standard PC server, and performing more complex applications with sophisticated content analysis than a standard network switch or PC server.

Summary

In a relatively few, short years, traffic management has evolved from the use of *ad hoc* systems to balance load across multiple computers (constituting a large Web server) into a fundamental part of the architecture of increasingly important, large, Web-technology servers and applications.

The selection of Traffic Management systems should be made with thoughtful consideration of the history of these systems, and with the anticipation that the evolution in the requirements for these systems will continue, driven both by the increased usage of Traffic Management systems and by increasingly sophisticated Traffic Management functions and application integration. Important criteria to consider include:

- Adequate raw switching capacity for increasing traffic loads due both to increasing server traffic loads and to the economic benefit of aggregating multiple servers under common management.
- Adequate content processing capacity for sophisticated content analysis at line speed.
- Thoughtful integration of security and authentication functions, starting with SSL processing.
- Clear direction for application integration, with a mature understanding of the importance of emerging interface technology (e.g., XML, SOAP) and the importance of *de facto* and formal industry standards.

NetsEdge Research Group
399 Main Street
Los Altos, CA 94022

John Katsaros
650-949-3256
john@netsedgeonline.com

Peter Christy
650-559-2103
pchristy@netsedgeonline.com

NetsEdge Research provides marketing and strategy research reports and consulting services, specializing in areas related to Internet infrastructure. NetsEdge offerings combine the formidable and unique experience and perspective of the two principals: John Katsaros and Peter Christy. At NetsEdge Research, John and Peter are continuing the work they began at the Internet Research Group (sold to Jupiter Research in 2000). At IRG, they produced definitive, early studies of the emerging markets in Internet Caching, Traffic Management, and Content Delivery and Distribution. Their clients for this work included most of the participants in the markets.