White Paper

Acceleration 101

Acceleration is a term used commonly today. What is it?
What does it really do for your applications?

by Peter Murray
Technical Marketing Manger

Contents

Introduction

Getting your applications to work properly over a Wide Area Network (WAN) is a complex task, and it is not likely to get easier any time soon. Trends such as data center consolidation, the advent of Web 2.0 applications, and the move to web-based application delivery have only served to increase complexity and slow user response times. Often, the result is sluggish application response and at worst, abandoned applications and shopping carts due to slow or failed web page loads.

The good news is that you can do something to improve application performance over slow or congested WAN networks. An Application Delivery Network can accelerate your applications and help make sure they're secure, fast, and available.

Throughout this paper, Wonder Widgets, Inc. will serve as an example company. Wonder Widgets, Inc. manufactures the uPozr, an MP3 player with WiFi capability and embedded internet applications and the uFony, a mobile phone-enabled version of uPozr. Customers buy direct from sales representatives internationally, and purchase online as well.

Wonder Widgets, Inc. has a global footprint, with headquarters and data center located in Paris, a major design/engineering branch office and back-up data center in San Diego, software development outsourcing in Mumbai, and device manufacturing and branch offices in multiple sites in China. Finally, they have a globally distributed sales/distribution force working from several branch offices, home offices (SOHO), and mobile workers. Wonder Widgets, Inc. has 10,000 employees worldwide.

Wonder Widgets' primary revenue sources are from supplying and delivering content for their devices via the Internet, in addition to ad revenue received from hosted applications on their devices. They provide both a service for purchasing MP3 files as well as a Web 2.0 community site, which enables users to create their own web space and web applications for use on their device.

Figure 1 shows the global distribution of Wonder Widgets, Inc. offices.

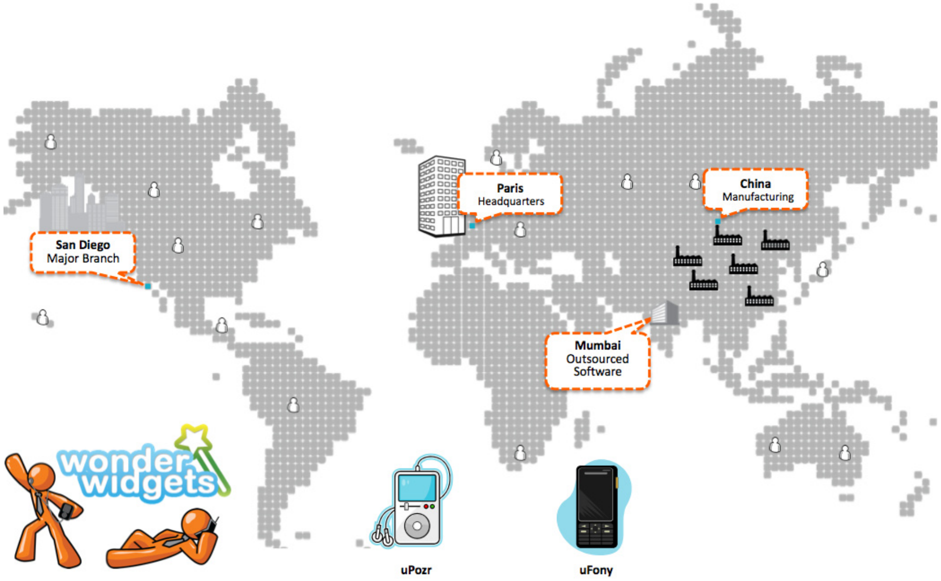

Figure 1. *Wonder Widgets' Inc. offices*

Wonder Widgets, Inc. is experiencing great success with its social networking site and content delivery applications. However, with the latest releases of applications for the uPozr and uFony performance of their download site was severely degraded due to demand spikes. Additionally, users reported the performance of the social networking site has consistently slowed over the past year. The board of Wonder Widgets, Inc. is concerned about these performance lapses and wants the issues resolved as soon as possible.

Kristina is the CIO for Wonder Widgets, Inc. During the most recent board meeting, she was asked to find a solution for their performance issues. She is already somewhat familiar with the industry but asks her team to investigate acceleration technology, figure out where it will fit in their current setup, and create a short list of potential vendors. Her first task is to make sure she and her team understand the components of acceleration and what makes the most sense for Wonder Widgets, Inc. She calls a meeting with her team to discuss the project.

Bruno, the network architect, is somewhat skeptical about acceleration but willing to participate. His previous company deployed a compression solution that was supposed to solve their slow application performance, but it only helped solve a couple of problems, had no effect on others, and was management-intensive. He's continued to read about acceleration technology in the trade press, but sometimes finds vendor claims to be conflicting and confusing.

On the other hand, Anne, the application architect, has heard from a friend at another company about seemingly incredible performance gains they've achieved with their web applications. This is important to her because she has been struggling to improve user response time for both the social networking site and the content delivery site. After a team discussion, they set out to investigate acceleration technology.

What Is Acceleration?

Acceleration can be confusing if you're new to the technology. No, no one has figured out how to exceed the speed of light. Yet propagation delay, caused by the effect of the speed of light over distance, accounts for most network latency. No one has figured out how to bend time or remotely over-clock your laptop's processor to increase its processing. And a 1 Gbps link is still just that—you can't send bits in excess of the link's clock rate. Any data on its way to you that is sent at a rate greater than the slowest link between you and the server must be buffered if that link becomes saturated. So, what is acceleration?

We can do much to overcome the limitations inherent in today's networks and applications. The result is improved transaction response time, and that's what we call acceleration. Acceleration is really a series of techniques that modify, or optimize, the way TCP, other protocols, applications, and data flows behave over the network.

These optimizations improve performance and help complete application requests and file transfers in the least possible time over the WAN. Optimizations improve TCP and HTTP performance, mediate application issues that degrade performance, and minimize both the frequency of communication (called "chattiness") and the amount of data sent between clients and servers. The result is applications that operate at near-LAN speeds, using the minimum amount of available network bandwidth, especially over slow and congested networks.

Products designed to accelerate application performance have existed for years. Acceleration began with early load balancing products, which divided load across multiple servers and services to speed response times. Newer products offload SSL, optimize TCP transactions, compress data for transmission across a network, and cache data at remote locations. The latest web acceleration products can even change browser HTTP requests and server responses to minimize data transfer and maximize response times. Even better, the latest web acceleration features can now run as add-in software on some advanced Application Delivery Controllers.

Acceleration Components

Acceleration is comprised of many individual components. Kristina's team makes a list based on their investigation of the market. The examples they find include:

- Server load balancing
- Global server load balancing
- Compression
- Data deduplication
- Caching
- Optimizing TCP
- Optimizing HTTP and web applications
- SSL processing offload

The team then begins to look into these features in detail.

Server Load Balancing

Bruno, the network architect, begins by looking into server load balancing. Server load balancing is one of the best-understood parts of application acceleration. A virtual IP (VIP) address is configured on a load balancing device, which then acts as a proxy for that destination IP address. The load balancer directs client requests made to that VIP to one server among a pool of servers, all of which are configured to respond to requests made to that address. All servers in the pool contain the requested information. Using a server pool improves response time by reducing the load on each server and therefore the time required to process a request.

Wonder Widgets already uses load balancing products in both data centers. While discussing load balancing with Bruno, Kristina learns that some load balancing features are already accelerating Wonder Widgets' WAN traffic. Bruno tells her how early load balancer technology has advanced over the years and that a new term—Application Delivery Controller (ADC)—has emerged. ADCs combine features of traditional load balancers with other features like acceleration and security. ADCs in different locations work together to provide what's called an Application Delivery Network (ADN), which is a combination of technologies that all work to make applications secure, fast, and available. He tells her that some vendors now offer advanced ADC devices, which offer security and acceleration features as modules along with traditional load balancing. During their weekly staff meeting, Bruno reports on ADC technology. Kristina asks Bruno to include advanced ADC products in his plan.

Global Server Load Balancing

While searching for information on server load balancing information, Bruno looks into another technology: global server load balancing. It improves application performance by directing client requests to the nearest or best-performing data center to ensure the fastest possible response.

Global server load balancing typically uses an intelligent DNS resolver to monitor data centers and route connection requests to the best-performing site for each user at that particular moment. The global server load balancing device communicates with other devices, such as load balancers and servers, at each data center location to compare current availability and load.

The global server load balancer sends requests to a particular site based on many factors. The simplest decision is to send all requests to one site if another is down. Another option is to send the request based on the data center with the fastest response time. Yet another decision might be to send a request based on the client's source address. Because service providers often assign client IP addresses, it's possible to know the geographic area and source IP addresses served by the provider and ensure that requests from that IP source address range are sent to the closest site to minimize the effects of latency.

The Paris data center load continues to rise quickly, and Bruno knows that Wonder Widgets, Inc. will have to increase capacity soon. He sees that using the San Diego data center along with the Paris data center can speed up response times and provide higher availability for customers. He reports this information back to Kristina, who tucks away this tidbit for future investigation.

Compression

Compression is one of the oldest acceleration techniques, having been around for decades. During the weekly staff meeting Bruno explains that that Wonder Widgets, Inc. is already using compression technology for HTTP and HTTPS traffic passing through their server load balancers. He knows that some data can be compressed, or encoded to remove repetitive data and reduce the amount of data transmitted between browser and client or between sites.

Gzip, the most common compression algorithm, is implemented in virtually every web browser and server. Compression algorithms such as gzip are good at finding small, repeating patterns and reducing the characters required to send them. Besides web servers and browsers, acceleration devices implement compression. This is done for two reasons: first to offload compression overhead from web servers and second, to enable the acceleration device to perform other optimizations that improve performance for an HTTP/HTTPS stream.

Compression can be computationally expensive, especially for algorithms that provide high compression levels. These algorithms are of limited use with high-speed communication, where delays must be minimized to maintain rapid user response times. More effective compression algorithms are therefore limited to low-speed communications where more time is available to perform compression processing without degrading user throughput and hence, response times. Fortunately, compression hardware assist is now available in some acceleration devices that can achieve compression rates in excess of 1 Gbps.

Kristina learns from Bruno that compression technology varies widely among vendors. He tells Kristina about a conversation with a friend who recommends a variation on compression called data deduplication. She asks him to follow up.

Data Deduplication

Bruno then investigates data deduplication, a sophisticated form of compression useful for most IP applications. Data deduplication requires symmetric acceleration, meaning a device or software client must be at each end of the communication. The client-side device sends a request to the server-side device. The server-side device responds to the client object request by sending new data along with a dictionary entry, or pointer that references the data, to the client-side device. The client-side device stores the data and the pointer before sending it on to the requesting client. When any user requests the data a second or subsequent time from the client-side device, the server-side device checks for changes to the data, then sends one or more pointers along with any new data that has not previously been sent. A pointer can identify repeated data that spans many packets, or even an entire file.

Data deduplication takes one of two forms: block-based or stream-based. Block-based data deduplication uses a fixed block size to reference repetitive data. A typical block length is 256 bytes. Repetitive data must fill at least one block. Multiple blocks may be chained until a block cannot be filled, at which time the entry ends. For example, a repetitive pattern containing 501 bytes can only be saved to one 256-byte block. When a second or subsequent request is made for the same data, the extra 245 bytes must be retransmitted by the sending device, as they do not neatly fit a block boundary, which limits the effectiveness of this method.

By contrast, stream-based data deduplication has no arbitrary data length limit. A stream can be of any length, and has no fixed storage interval. The advantage to this approach is that streams of any length can be stored and referenced as needed.

A significant advantage to data deduplication over caching is that it is not application-dependent. This means the sending device transmits a pointer every time it encounters an identical data stream, regardless of the content being sent. For example, let's say the phrase "Wonder Widgets, Inc. rocks!" is sent in an email. The sending acceleration device sends both the phrase and a dictionary entry to the remote device. Now let's say that someone requests a word processing document that contains the same phrase. The sending device encounters the repeated phrase and sends only the pointer.

The acceleration device operates either transparently or non-transparently. A transparent-mode client-side device must remain in contact with its sending peer in order to be able to service a request. This means the sending peer ensures that access control mechanisms are followed, so the remote peer only serves valid requests. The sending device also verifies the freshness of the data, sending new data only when it encounters changes and ensuing the remote device always serves fresh information.

An acceleration device that operates non-transparently is able to continue serving content even if it loses contact with its peer. Non-transparent devices face two risks. First, the client-side acceleration device must implement a way to ensure that access mechanisms are followed faithfully to prevent unauthorized users from gaining access to an object. Second, when operating out of contact with the sending device, it's possible for the client-side device to pass on an out-of-date copy of the object.

When Bruno reports on data deduplication to Kristina, she's intrigued by the idea of using it for replicating data between Wonder Widgets, Inc.'s data centers, and for accelerating the legacy application that currently runs in both the Paris and San Diego offices. She asks Bruno to include transparent deduplication in his acceleration planning.

Caching

Caching involves storing data close to users and re-using the data during subsequent requests. Caching usually takes one of three forms. The first is the classic approach taken by web browsers and web applications. In this case, the web application code running on a server instructs a browser to cache an object marked as static for a specific time period. During that time period, the browser reads the object from cache when building a web page until the content expires. The client then reloads the content. Caching prevents the browser from having to waste time and bandwidth by always accessing data from a central site. This is the most common form of caching in use today.

The second form involves deploying an acceleration device in a data center to offload requests for web application content from web servers. This method operates asymmetrically, with the acceleration device caching objects from web servers and delivering them directly to users. Some acceleration devices cache static content only, while some additionally can process HTTP responses, include objects referenced in a response, and send the included objects as a single object to a browser. This not only offloads web server processing but also offloads web browser processing too. A side benefit to this approach is that as the acceleration device is typically in the data center and connected to higher-speed connections, the acceleration device can both assemble the objects from instructions in the HTTP response and deliver them using fewer objects and with fewer transactions. Operating in this manner, caching can dramatically reduce server TCP and application processing, improve web page loading time, and hence reduce the need to regularly expand the number of web servers required to service an application.

The third form of caching involves using symmetric acceleration devices to cache and serve content to users at the remote site. The remote acceleration device serves content locally whenever possible, which reduces both response time and network utilization. This form of caching can be deployed not only for HTTP, but also for other protocols as well.

Caching has its limitations. First, if the client-side acceleration device serves content regardless of whether it is in contact with its remote peer, the client-side device must implement access control to prevent unauthorized access to an object. Second, the client-side device may serve older, stale versions of content that change after the connection between the devices is broken. While this typically is not an issue with static web content, it can have significant impact on files that regularly change. When both issues are addressed, remote caching can greatly improve application performance, especially for web applications and static files used with other applications.

Kristina knows that caching can be a powerful way to speed up page loading and application response. She understands that the way caching is deployed is important, and is aware of the concerns regarding remote caching. She asks Bruno to include caching where appropriate in their design and evaluation.

Optimizing TCP

Although TCP is ubiquitous today, the protocol has undergone many updates to help overcome limitations that existed in earlier versions. An acceleration device can help optimize TCP by implementing features that may not be present in either a client or server's TCP implementation.

An acceleration device can also decrease the number of server-side TCP connections required to service client requests. Additionally, it can help accelerate HTTP traffic by increasing the number of simultaneous client-side TCP connections a browser can open while downloading a web page.

Bruno decides to examine the three types of TCP optimization further. He knows his existing network has some optimizations in place but wants to understand what else he can do to improve TCP performance.

General TCP Optimizations

Because it operates as a proxy, an acceleration device may be able to implement features missing from a client or server that can help speed application delivery. The acceleration device may be able to leverage optimizations natively supported by particular client or server operating systems and is likely to be able to implement optimizations that are not operating-system specific. The benefit to high speed, high latency WAN connections is that the acceleration device can perform TCP window scaling to improve performance. To overcome packet loss, the acceleration device can implement selective TCP acknowledgements (SACK) and advanced congestion control algorithms to prevent TCP from reducing throughput.

These are only two examples. Some acceleration devices implement hundreds of improvements to TCP in order to help it perform better. Bruno knows that his existing load balancing solution optimizes TCP, but wants to use the technology as much as possible to accelerate his traffic globally. He sees that although Wonder Widgets, Inc. has high-bandwidth links between San Diego and Paris, he can achieve even better performance by reducing the effects of TCP on his users with their disparate network connection speeds and latency.

Decreasing Server-side TCP Connections

Reducing server-side connection processing can dramatically improve application performance and reduce the number of servers required to host an application. TCP connection setup and teardown requires significant overhead, particularly for servers. As the number of open server connections increases, maintaining the open connections while simultaneously opening new connections can severely degrade server performance and therefore, user response time.

Although multiple transactions (for example, file transfers) can occur within a single TCP connection, a connection is generally between one client and one server. Normally, a connection closes either when a server reaches a defined transaction limit or when a client has transferred all needed files from that server. Because an acceleration device operates as a proxy, it can aggregate, or "pool," TCP server-side connections by combining many separate transactions, potentially from many users, through fewer (or one) TCP connections. The acceleration device opens new server-side connections only when necessary, and instead reuses existing connections for requests from other users whenever possible.

Increasing Client-side TCP Connections

By default, most web browsers limit the maximum number of simultaneous HTTP/HTTPS connections that the browser can open to one URL. For example, Microsoft Internet Explorer v7 and below limit the maximum number of simultaneous connections to two per domain. Earlier versions of Firefox limit the browser to eight connections per domain. Given that a web page can contain dozens of objects, this limitation can greatly slow page loading times.

For example, suppose a user running Internet Explorer v7 requests a page from a web server that returns a response containing a list of the 30 objects that make up the web page. Further assume that that all objects are accessed through the domain, www.example.com. The browser opens two connections to www.example.com, requests one object at a time per connection (the limit imposed by TCP), and then reuses the two connections until all files have been downloaded or the connection reaches the server's transaction limit. If the connection suffers high latency, round trip time is high and download speed can be greatly reduced.

If the server terminates the connection after reaching a pre-defined transaction limit, the browser opens another connection to that URL. This process continues until the page downloads completely. Operating this way needlessly increases the page load time.

Some acceleration devices can "spoof" a browser by modifying the URLs in an HTTP response to speed page downloading. The modified URLs must first be defined in DNS to point to the same IP address. When examining the server response, the modified names appear to the browser to be different servers, so the web browser opens parallel connections to these altered URLs rather than serially downloading the objects from one URL.

Kristina sees that these implementations can help her increase user satisfaction and control costs significantly. Anne, the application architect, hears Kristina and Bruno discussing these optimizations and asks them to include TCP optimization in their plans. Bruno tells Anne that they already implement some features, but will definitely include the rest in his vendor evaluation.

HTTP Protocol and Web Application Optimizations

Kristina has received a lot of grief from users and upper management dissatisfied with declining web application performance. Two recently deployed applications that worked fine during testing are experiencing degraded performance when used over the WAN. One is a homegrown wiki application that pulls content from multiple sources on the Internet; the other is a commercial application that uses a web-enabled front end to communicate with a popular database product. Both use significant bandwidth and are inefficient at enabling the caching of content.

Bruno and Anne are responsible for maximizing application performance, so Kristina directs them to investigate HTTP and web application protocol optimization.

Fortunately, some advanced acceleration devices specifically target web applications and can significantly improve performance, especially over WAN links. Because it acts as a proxy, a web acceleration device can inspect the content of web requests and responses, proxy static-content to offload servers, and manipulate requests and responses to reduce web traffic and increase performance.

As mentioned earlier, reducing server connections and increasing simultaneous browser connections can significantly improve performance. Some web acceleration devices can optimize TCP and therefore improve performance for all web applications.

HTTP protocol optimizations maintain high user performance levels by optimally tuning each HTTP session. For example, some web applications are unable to return an HTTP 304 status code (Not Modified) in response to a client request rather than returning the entire object. Because an acceleration device proxies connections and caches content, it may be able to note when there is no change to a requested object and return the 304 response instead. This enables the browser to load the content from its own cache, even in conditions where the web application is hard-coded to re-send the object.

Some acceleration devices can additionally examine and change server responses to provide better browser and server performance. For example, some off-the-shelf and custom applications add a no-cache header to some objects, which directs a browser not to cache an object, rather to download the object from the origin web server every time. The purpose of the no-cache header is to ensure a browser always downloads dynamic (changing) data. However, applications in some cases mark static data like a company logo as being non-cacheeable. Some acceleration devices can re-write the server response to mark the object as being cacheable and supply a more realistic expiration date. This feature can help remedy problems with off-the-shelf or custom-developed applications where code cannot easily be modified.

Accelerating HTTP and web applications can help accelerate other applications as well. By reducing HTTP communication, including decreasing TCP opens/closes, opening more browser connections, and caching all possible web objects, this technology can open up significant bandwidth on valuable WAN links.

Anne sees that web acceleration can immediately reduce traffic from their new wiki and commercial web applications, and help with other web applications. Anne and Bruno decide to evaluate this technology immediately.

SSL Offload

SSL technology secures communication between a client and server, and with some devices, can secure all communication between sites over public networks. SSL was created specifically for HTTP processing, primarily for commercial transactions like secure shopping over the web. It has the advantage of being built into virtually every browser and web server—after all, HTTPS is just encrypted HTTP.

SSL processing overhead is a significant burden to servers. SSL-encrypted traffic cannot be accelerated by an external device because acceleration devices cannot understand the content of encrypted packets. Once encrypted, the only thing that can be done is to implement QoS to prioritize the traffic.

However, implementing SSL processing on an acceleration device in the data center rather than a server can improve application response because the device can perform all optimization functions on the packets in clear text. Implementing SSL in an acceleration device also reduces SSL overhead and management, because SSL processing is offloaded from the servers.

Incoming packets are decrypted, then processed. The acceleration device can perform caching, compression, and web/application optimizations before passing packets to the server. Outgoing packets can undergo acceleration optimizations, then undergo encryption before being transmitted over the WAN.

Remote office users can also benefit from having an acceleration device in the remote office to perform SSL processing. Client requests for static data or data that has undergone deduplication can be served locally, greatly reducing response time. Compression can be optimized for the WAN link, ensuring the best possible perfomance.

Claudia, the security architect, tells Kristina that Wonder Widgets, Inc. is already using SSL offload in the Paris data center to accelerate consumer traffic and corporate web traffic. She suggests using SSL offload between the Paris and San Diego sites and between the Paris and Beijing sites to help improve performance. They agree to use SSL offload between sites wherever it makes sense.

Conclusion

Kristina and her team have seen for themselves that application acceleration is viable technology that can improve user productivity, ensure remote application deployments are successful, and help boost their online sales. By optimizing TCP, other protocols, applications, and data flows, Wonder Widgets, Inc. and, its internal users, customers, partners, and suppliers can benefit from acceleration. They're already using some aspects of acceleration, but see that implementing advanced Application Delivery Controllers and web acceleration can help ensure that the complex new Web 2.0 applications they've implemented and are implementing now run well and delight users.

It is important to understand that acceleration is not a "one size fits all" set of features. Some features are widely implemented, while others are limited to one or a small subset of vendors. The most important decisions should be choosing a partner you can trust and choosing the solution that gives you the best application acceleration return for your investment.

F5 Networks, Inc.
Corporate Headquarters
401 Elliott Avenue West
Seattle, WA 98119
+1-206-272-5555 Phone
(888) 88BIGIP Toll-free
+1-206-272-5556 Fax
www.f5.com
info@f5.com

F5 Networks
Asia-Pacific
+65-6533-6103 Phone
+65-6533-6106 Fax
info.asia@f5.com

F5 Networks Ltd.
Europe/Middle-East/Africa
+44 (0) 1932 582 000 Phone
+44 (0) 1932 582 001 Fax
emeainfo@f5.com

F5 Networks
Japan K.K.
+81-3-5114-3200 Phone
+81-3-5114-3201 Fax
info@f5networks.co.jp