



Global Knowledge®

Expert Reference Series of White Papers

Three Easy Ways To
Achieve and
Understand WAN
Acceleration with TCP
Applications

Three Easy Ways To Achieve and Understand WAN Acceleration with TCP Applications

Alex Marcotte, CCIE #16673, CCSI, & Cisco WAN Specialist

Introduction

Long gone are the days where work ends at 5:00PM, or that, in order to access a client file while on the road, you had to pull out a physical file folder from your briefcase.

The name of the game these days is mobility and accessibility. CIOs and other decision makers want to achieve worker productivity by giving employees the tools they need anywhere at any time. Unfortunately, those who make these promises sometimes don't ask for your opinion first, and you are now tasked to make it happen.

Software developers write code to power applications that drive today's businesses. Whether you are looking at ubiquitous applications such as Microsoft Word or your own in-house software, chances are that they are not very WAN-friendly. There may be different reasons for this, but most programmers do not take networks into consideration when writing code, which ends up creating very "chatty" applications.

That being said, the focus of this paper is to highlight the TCP protocol's shortcomings for WAN transmission purposes. Let's look at your options and what you can do to address those challenges.

Understanding Latency

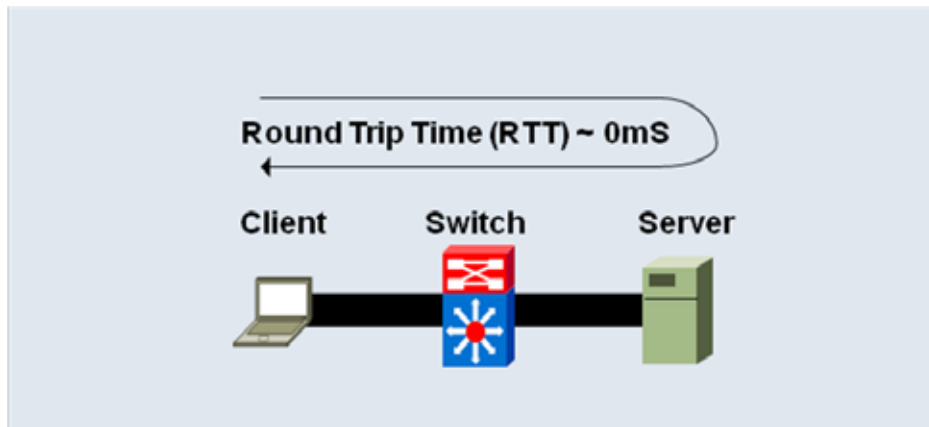
Chatter of applications on a LAN seldom matters due to the low amount of latency. Normally, a typical LAN will not create more than 1MS (milliseconds) of latency on your roundtrips. This means that as a host sends a packet to another host on the LAN, it should not take more than 1MS to get a response.

Latency comes from various sources, the first one being the very hardware (switches, routers) that you are dealing with as well as the media that you are using. Latency is increased by the distance that a packet needs to travel from point A to point B, and it is limited by the speed of light. Here is a summary of the different sources of latency on a network:

- **Propagation:** This is simply the time it takes for a packet to travel between one place and another at the speed of light.
- **Transmission:** The medium itself (whether optical fiber, wireless, or some other) introduces some delay. The size of the packet introduces delay in a round trip since a larger packet will take longer to receive and return than a short one.

- **Routers and other processing:** Each gateway node takes time to examine and possibly change the header in a packet (for example, changing the hop count in the time to live field).
- **Other computer and storage delays:** Within networks at each end of the journey, a packet may be subject to storage and hard disk access delays at intermediate devices such as switches and bridges. (In backbone statistics, however, this kind of latency is probably not considered.)

Look at a packet movement between hosts A and B on the same segment of a LAN.



Source: Cisco WAAS 2.0

This being very fast, a normal user would never notice the impact of TCP sessions and all the background messaging taking place. For example, a Wireshark packet trace revealed that when a user opens a simple Word document residing on a Common Internet File Sharing protocol (CIFS) share on the network, over 1000 messages are generated between the two hosts. This is due to the semantics of the application (Word) on top of the CIFS. Once again, it's not an issue over the LAN, but try this over a WAN connection, and you may have some problems.

For a corporate link between sites, latency is a reality and is unavoidable. Regardless of the service level agreement that you may have with the Telco (even if you are the Telco), the laws of physics still apply.

Latency can be roughly calculated by the distance covered, multiplied by the speed of light. This will give you the absolute lowest number for the latency of your connection. You then have to factor in all the devices introducing latency on the path such as routers, switches, modems, etc.

Here Is an Example

ACME Company has an office in New York City and another in San Francisco. Both sites may have some servers and ACME is leasing a T-3 (44.736 Mbits/s). This is roughly half the speed of a 100MB connection that a desktop user has in the office. However, users are reporting that opening files that reside in New York from San Francisco is really slow. Why? Latency.

Let's assume that the RTT (Round Trip Time) of a packet on the LAN in New York is 1MS. A New York user opening a New York file will get his file quickly considering this:

1000 messages are exchanged between the host and
the server at $\sim 1\text{MS RTT} = \sim 1$ second total time

In this case, NY to SF is approximately 2900 miles, which when multiplied by the speed of light (299,792,458 Meters per second) puts the raw latency at 16MS. Factor in the gear in between point A and point B and you get closer to 24MS for a good dedicated link. If you share the bandwidth, you are subject to traffic jams so you could be more realistically closer to 40MS.

Now let's see what happens with a user in San Francisco who wants to open a New York file. The user complains that it takes over 6 seconds to get the file, even with all this bandwidth available. The problem lies with the latency. One thousand messages will be exchanged sequentially with CFS (TCP connection) and the round trips will add up. You have then 3 to 4 seconds of setup time even before the file is even transferred.

Sadly, this is not the worst example. Most of us will deal with home and SOHO users who are using DSL, cable modems, or EVDO cards. These connections average 40MS of RTT for the wired solutions and 120MS for the wireless ones. The file in the previous example would take at least 4 seconds of setup time over the wire and 12 seconds of setup time before the file is even transferred. Some applications would time out before even giving it a chance to the transfer. But this is not the only issue.

The Solution

Latency is not something that you can correct by itself. The simple theory is to avoid latency, or to avoid transmitting as many packets over the WAN as you would over a LAN. In the Cisco WAAS world, we use the Data Redundancy Eliminator (DRE) engine to do this. The DRE engine allows the endpoints to understand the protocol at hand (CIFS, HTTP, MAPI, etc.) and send fewer commands over the WAN as compared to the original protocol.

DRE also uses the concept of caching. Data being cached at the remote endpoint eliminates redundancy since the same data being requested by multiple users is served from cache locally and, therefore, is not transmitted over the WAN over and over.

Cisco's DRE is a bit level cache and is application agnostic. Cisco caches on patterns of ones and zeroes (binary) so the same content contained within multiple documents can be considered to be already cached. This is done via a "chunking" process that is proprietary to Cisco and is discussed in the Cisco WAAS class.

Packet Loss

The greater the distance, the greater the chance that you will suffer packet loss. Packet loss can be caused by a number of factors, including signal degradation over the network medium, oversaturated network links, corrupted packets rejected in-transit, faulty networking hardware, faulty network drivers, or normal routing routines.

TCP connections have a way to deal with packet loss by using retransmission. TCP connections negotiate a Maximum Window Size (MWS) that indicates how much data can be transmitted at one time while unacknowledged, also referred to as data in flight.

The main issue here is that if packet loss is detected, it may not mean that the entire contents of the window have been lost. TCP doesn't know that and the default behavior is to retransmit the entire contents of the window. On WAN connections, you have to factor in the speed and the time it took to transmit the original window. This adds to the overall transaction time which, once again, can cause the application to just time out.

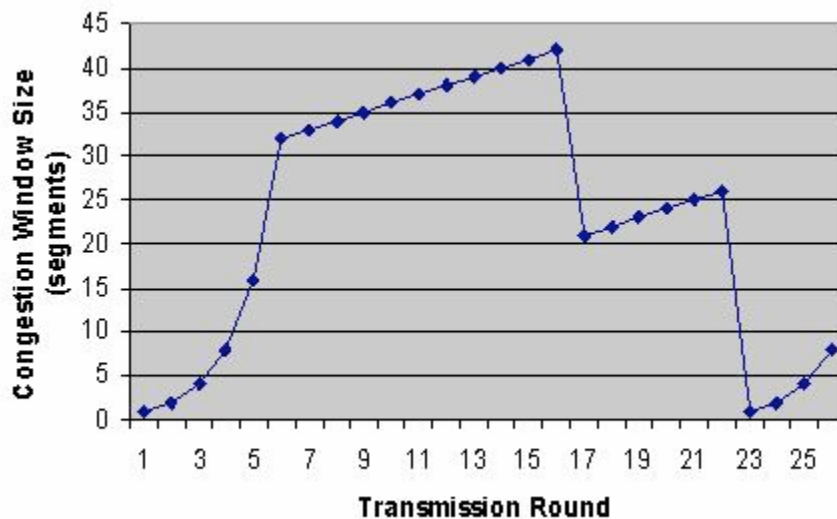
Solution

The TCP Selective Window Acknowledgement (SACK) options as defined in RFC 2018 allows a TCP receiver to precisely inform the TCP server which segments have been lost. This increases performance on high-RTT links, when multiple losses per window are possible.

This is not a native TCP behavior, but you can make use of SACK within Windows and UNIX operating systems as long as both ends can understand SACK. When using a Cisco WAAS implementation, SACK is part of the standard operating procedure and can be performed between two WAN Acceleration Engines (WAE) by simply using the Transport Flow Optimization (TFO) method.

Bandwidth Starvation and Congestion

If you are using mostly HTTP applications, you may suffer from this problem. This problem is caused by the original TCP slow start, which will timidly negotiate small window sizes to begin with, then grow the window size proportionally until congestion (packet loss) is encountered.



TCP slow start formula with initial window (IW):

$$IW = \min(4 * SMSS, \max(2 * SMSS, 4380 \text{ bytes}))$$

Read more: www.faqs.org/rfcs/rfc2581.html#ixzz0Zj8GI3r7

Now, this may be great for applications that are using long-lived sessions, but what about applications that send a small amount of data per session, such as HTTP? These sessions may suffer from bandwidth starvation because the total amount of the available bandwidth may not be fully discovered by TCP due to the low amount of round trips required to transmit the data. Large initial windows are defined in RFC 1323 and could allow you to multiply the original MSS (discovered in the 3-way handshake) by 4 or 5 times the value and use it initially instead of the “twice the amount” used in regular TCP.

Congestion is dealt with much more efficiently with Cisco WAAS than regular TCP. (Refer to the diagram above.) When congestion is encountered in TCP, the MSS is dropped in half. When using a WAAS implementation, MSS is dropped by 1/8 of the value, allowing for much faster recovery.

Dealing with bandwidth starvation means that you would like TCP to start sessions with a Large Initial Window (LIW) measured by the Bandwidth Delay Product (BDP). The formula to calculate large initial window sizes based on BDP is:

$$\begin{aligned} (\text{Bandwidth-in-bits-per-second}) \times (\text{Round-trip-latency-in-seconds}) &= \text{TCP window size in bits} / 8 \\ &= \text{TCP window size in bytes} \end{aligned}$$

So, the large initial windows can be configured on endpoints that you own (servers) or by using a WAN accelerator, such as the Cisco WAAS solution. The TCP initial window can be configured on most UNIX, Linux, and Windows hosts. If you would like to use a BDP calculator, one is available at: www.speedguide.net/bdp.php.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge. Check out the following Global Knowledge course:

[Cisco WAAS 2.0](#)

For more information or to register, visit www.globalknowledge.com or call **1-800-COURSES** to speak with a sales representative.

Our courses and enhanced, hands-on labs and exercises offer practical skills and tips that you can immediately put to use. Our expert instructors draw upon their experiences to help you understand key concepts and how to apply them to your specific work situation. Choose from our more than 1,200 courses, delivered through Classrooms, e-Learning, and On-site sessions, to meet your IT and business training needs.

About the author

Alex Marcotte is a CCIE in security #16673, Cisco Certified Instructor (CCSI), and a Cisco WAN Specialist with 14 years of experience in the field. He manages a data center in Houston, Texas (Evertech, LLC) that hosts several business applications such as remote backups, email and virtual servers.