TCP Optimization for Service Providers

Service providers want to deliver the best quality of experience possible to their subscribers while maximizing network efficiency. Data optimization technologies such as video compression and caching have proven beneficial in the past. Unfortunately, the benefits of these technologies have diminished due to the increased use of encrypted connections and the emergence of protocols like SPDY 2.0. Service providers need to look at TCP optimization as a method to improve subscriber QoE.

White Paper

Introduction

We are living in a connected world. The Internet is in nearly everybody's life and in the palms of their hands. In the first quarter of 2014, smartphone sales hit 281.5 million units, rising 28.6 percent Q/Q.[1] The number of mobile devices connected to the Internet exceeded the global population in 2014, and will continue to grow from there. The amount of data consumers use is growing as well. Some sources claim that data traffic will increase eleven fold between 2013 and 2018.[2]

The Internet is also infiltrating other aspects of our lives. Smart cars, smart glasses, and even smart TVs are available today. And although wireless technology is nothing new, it highlights how we have adapted from a life with the Internet as a luxury into a life with it constantly running in the background. This concept that many parts of our lives can now be controlled wirelessly is called the "Internet of Things."

How exactly will the Internet handle all of the data transportation required? Through Transmission Control Protocol (TCP), the key transport protocol of the Internet infrastructure. TCP is the essential glue, which together with Internet Protocol (IP), ensures that all applications connect smoothly to our devices. It allows us to share resources with billions of people, all over the world, at the same time. It also establishes and manages traffic connections and congestion while taking care of transmission errors. TCP has many moving parts, with new ones being added every day. Without the proper tuning and combination of these parts, TCP can hurt more than it helps in optimizing network use. Now F5 has created a framework to tune and adjust the parameters of TCP to enhance the connections and subscriber experience.

Historical Context

Initially, TCP had very few configurable parameters. When it was designed in 1973, during the infancy of the Internet, it was made for a wired infrastructure—the Advanced Research Projects Agency (ARPA) Net. The ARPA Net was a low-capacity network of 213 computers for the purpose of sharing knowledge among some of the world's leading research institutions at the time; thus, the design of the network and protocols was very different from what we use today.

Beginning in 1986 after 1G technology was released, the Internet began to experience "congestion collapses" where the transmission rates of the networks dropped by a thousand fold from 32 Kbps to 40 bps. This drastic drop in rates led to some investigation and analysis by leading computer scientists including Van Jacobson, who helped create what we now know as congestion control algorithms. These algorithms are methods that allow a TCP stack to alter how it treats data based on network conditions.[3]

The Internet has followed the trend of most technologies still alive from the early '70s—advancing at a rate nobody could imagine. Now, with the rise of smartphones, we are using mobile networks such as 3G and 4G, and high-capacity, fixed-line networks. Needless to say, these networks have very different characteristics than their ancestral networks.

Network Characteristics

As the Internet has progressed, user experience has always been the most important factor. The new breadth of access technologies leads to a wide spread of network characteristics. Recently, network access has shifted from wired networks to 3G and 4G cellular networks.

Network	Base Latency	Base Download Speeds	Buffer Sizes	Characteristics
3G (released early 2000s)	100–500 ms	21–42 Mbps	Small	High packet loss, even without congestion.
4G (late 2000s)	50 ms	Up to 300 Mbps	Larger than 3G	Lower packet loss due to error correction. Increased latency due to buffer sizes and not necessarily congestion.

Figure 1: Network characteristics for different wireless technologies

Modern network traffic is harder to control than it was in the 1980s because packet loss does not necessarily mean congestion in the networks, and congestion does not necessarily mean packet loss. As shown in figure 1, 3G and 4G networks both exhibit different types of behavior based on their characteristics, but a server may view the different aspects as congestion. This means that an algorithm cannot only focus on packet loss or latency for determining congestion. Other modern access technologies, such as fiber to the home (FttH) and WiFi, expand upon the characteristics represented above by 3G and 4G, making congestion control even more difficult. With different access technologies having such different characteristics, a variety of congestion control algorithms has been developed in an attempt to accommodate the various networks.

Algorithm Evolution

The changing network characteristics have led to a simultaneous evolution of congestion control algorithms.

Packet-Loss Algorithms

Initial algorithms, such as TCP Reno, use packet loss to determine when to reduce the congestion window, which influences the send rate. TCP Reno increases the send rate and congestion window by 1 MSS (maximum segment size) until it perceives packet loss. Once this occurs, TCP Reno slows down and cuts the window in half. However, as established in the previous section, modern networks may have packet loss with no congestion, so this algorithm is not as applicable.

Bandwidth-Estimation Algorithms

The next generation of algorithms is based on bandwidth estimation. These algorithms change the transmission rate depending on the estimated bandwidth at the time of packet loss. TCP Westwood and its successor, TCP Westwood+, are both bandwidth-estimating algorithms, and have higher throughput and better fairness over wireless links when compared to TCP Reno. However, these algorithms do not perform well with smaller buffers or quality of service (QoS) policies.

Latency-Based Algorithms

The latest congestion control algorithms are latency-based, which means that they determine how to change the send rate by analyzing changes in round-trip time (RTT). These algorithms attempt to prevent congestion before it begins, thus minimizing queuing delay at the cost of goodput (the amount of useful information transferred per second). An example of latency-based algorithms is TCP Vegas. TCP Vegas is heavily dependent upon an accurate calculation of a base RTT value, which is how it determines the transmission delay of the network when buffers are empty. Using the base RTT, TCP Vegas then estimates the amount of buffering in the network by comparing the base RTT to the current RTT. If the base RTT estimation is too low, the network will not be optimally used; if it is too high, TCP Vegas may overload the network. Also, as mentioned earlier, large latency values do not necessarily mean congestion in some networks, such as 4G.

By knowing the traffic characteristics and keeping the current inadequate algorithms in mind, service providers can implement an ideal TCP stack.

The Ideal TCP Stack

The ideal TCP stack should achieve one goal: optimizing a subscriber's QoE. To accomplish this, it must do three things: establish high goodput, minimize buffer bloat, and provide fairness between the flows.

High Goodput

High goodput is important for determining if the stack is optimized because it is a measure of how much of the data going through the network is relevant to the client. Goodput is different from throughput, which includes overhead such as unnecessary retransmission and protocol headers. Goodput also addresses the difference between content that was stalled or failed to complete versus content that the consumer was able to utilize. To help with maximizing goodput, TCP needs to address packet loss from interference as well as handle both small and large router buffers. Delay-based algorithms fail when competing with other flows for bandwidth; bandwidth-based algorithms fail when the buffers are too small or when quality of service policies are present in the network; loss-based algorithms fail by incorrectly slowing down for interference-based loss.

Buffer Bloat

Buffer bloat occurs when too many packets are buffered, increasing queuing delay and jitter in the network. Buffer bloat leads to performance issues by impacting interactive and real-time applications. It also interferes with the RTT calculation and negatively impacts retransmission behaviors. Thus, minimizing buffer bloat is ideal for an optimized TCP stack. Loss-based algorithms fail to minimize buffer bloat because they react after packets have been lost, which only happens once a buffer has been filled. These algorithms fail to lower the send rate and allow the buffer to drain. Instead, the algorithms choose rates that maintain the filled buffer.

Flow Fairness

Fairness between flows ensures that no one user's traffic dominates the network to the detriment of other users. Delay-based algorithms fail to fulfill this criteria because loss-based flows will fill all of the buffers. This leads to the delay-based flows backing off and ultimately slowing down to a trickle.

The F5 Solution

The F5 solution accomplishes the goal of the ideal TCP stack. It improves QoE for customers—resulting in less subscriber churn and increased revenue for service providers.

Achieving High Goodput

High goodput is achieved by maximizing the amount of data sent within a single packet and optimizing how quickly data is sent. The proprietary hybrid loss and latency-based algorithm, named TCP Woodside, is designed to maximize goodput while minimizing buffer bloat. It controls buffer size by constantly monitoring network buffering, and will slow down preemptively when needed—leading to a reduction in packet loss and minimal buffer bloat. However, when the queuing delay is minimal, TCP Woodside will rapidly accelerate to maximize the use of the available bandwidth, even when interference-based packet loss is present.

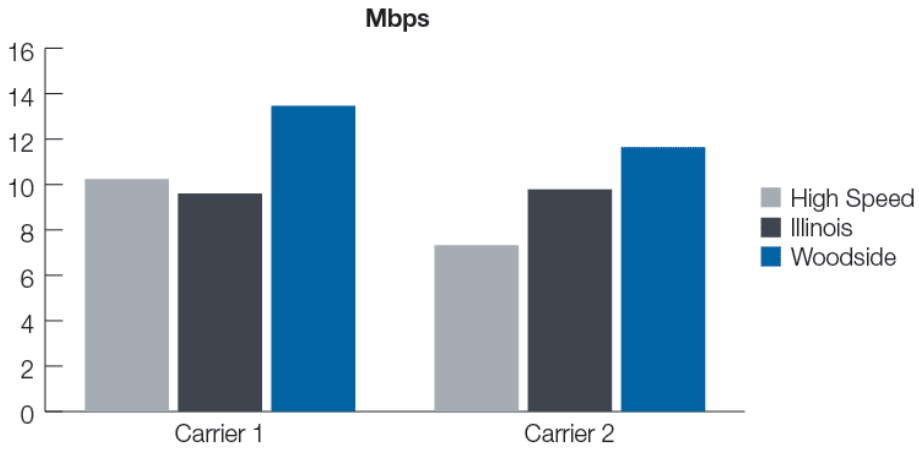

Figure 2: Comparison of real network tests between three carriers of TCP High Speed, TCP Illinois, and TCP Woodside algorithms. TCP Woodside performs particularly well.

Avoiding Buffer Bloat

Buffer bloat can be avoided by pacing the flow of data transmitted across the network. By knowing the speed at which different flows are being sent, the stack can control how quickly to send the packets through to the end device. This allows the buffers to adjust up without being overfilled. As a result, inconsistent traffic behaviors and packet loss due to network congestion are prevented.

In the figure 3 graphs below, a non-optimized stack's latency is compared to that of an F5 optimized stack. Both stacks have throughputs of 11 Mbps. In the left graph, the non-optimized stack has an increasing RTT—up to as much as 2.5 seconds—as more packets are sent through the network and the buffer starts to become bloated. However, in the right graph, the optimized stack's RTT stays around 200 milliseconds even as more packets are sent. This steady RTT time leads to an improved end-user QoE due to less "bursty" traffic, and reduces buffer bloat as well.

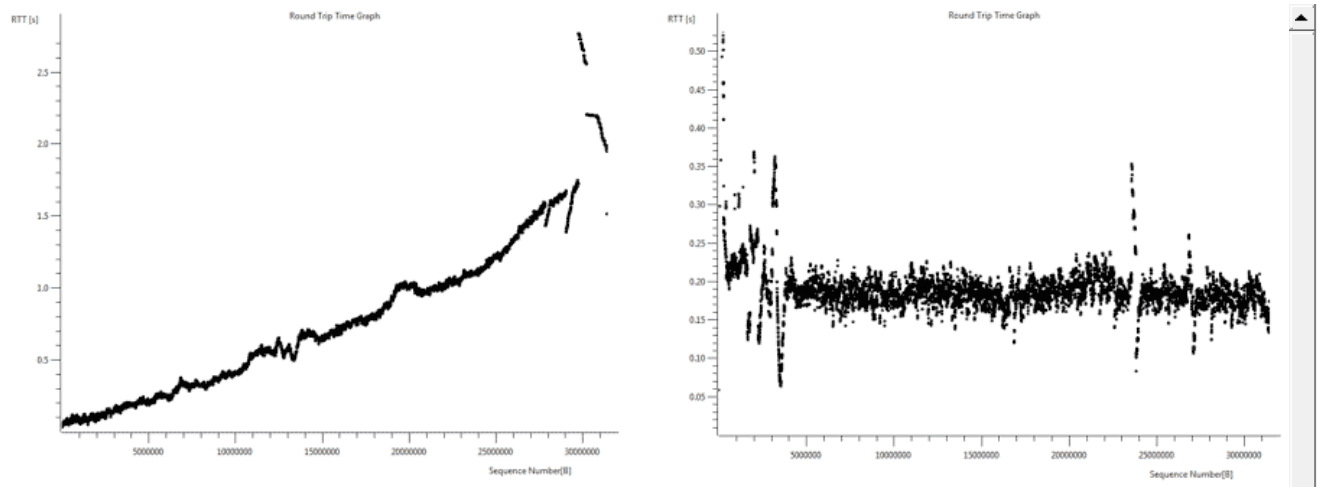

Figure 3: On the left is a standard, non-optimized transmission. Note how the average RTT increases as more packets are sent. On the right is an F5 optimized transmission. Despite more packets, the average RTT stabilizes and does not increase. Both graphs are from real 4G network tests.

Improving Fairness of Flows

Not only does rate pacing help with buffer bloat, but it also improves the fairness across flows. Without rate pacing, packets are sent immediately and consecutively. Having two flows at the same time means one flow will see different network conditions than the other flow, usually with respect to congestion. These conditions will affect the behavior of each flow.

As shown in the figure 4 left graph below, the flows have different behaviors at different times. Sometimes one flow has more bandwidth and sends more information. However, the next second, another flow may gain that bandwidth and stop the flow of others.

Controlling the speed at which packets are sent on a connection allows gaps to occur between packets on any individual flow. Instead of both flows attempting to send consecutive packets that become intermixed, one flow will send a packet, and the second flow can then send another packet within the time gap of the first flow. This behavior changes how the two flows see the network as well. Rather than one flow seeing an open network and the other seeing a congested network, both flows will likely recognize similar congestion conditions and be able to share the bandwidth more efficiently (as shown in the right graph).

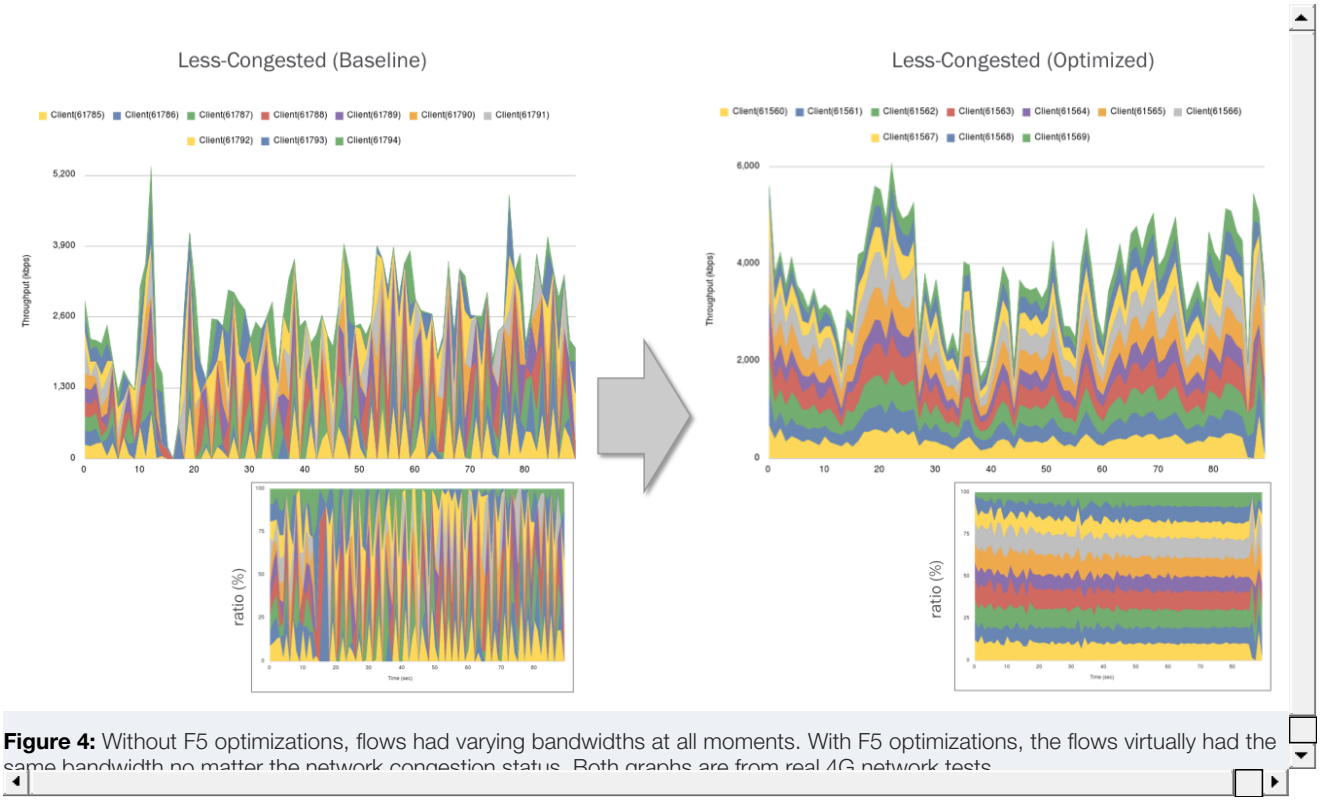

Figure 4: Without F5 optimizations, flows had varying bandwidths at all moments. With F5 optimizations, the flows virtually had the same bandwidth no matter the network congestion status. Both graphs are from real 4G network tests.

Maximizing Performance

With TCP Woodside and rate-pacing features working together, live test data shows that performance improves enough to bump subscribers from one category of congestion or signal strength to one category better. In figure 5, an optimized subscriber on heavy congestion receives better performance than the baseline medium congestion, and the optimized medium congestion signal performance is better than the baseline uncongested.

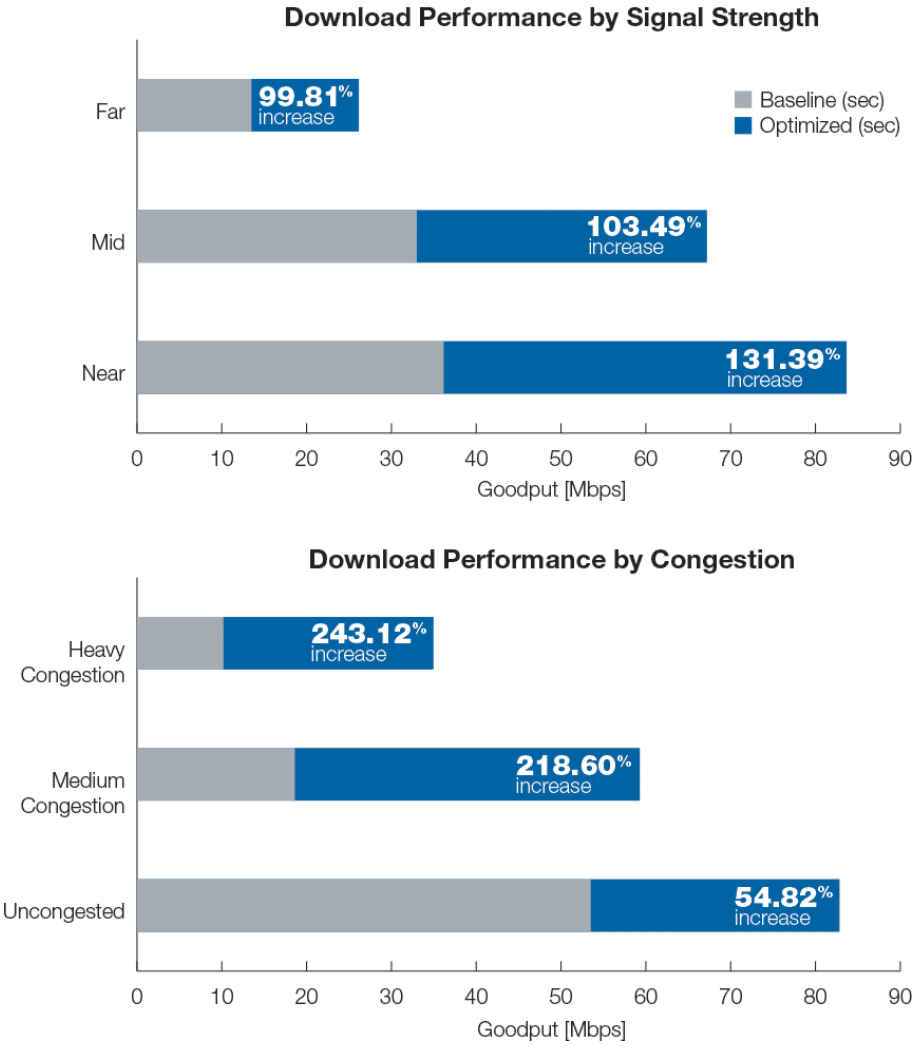

Figure 5: TCP optimization improvement under different conditions.

The F5 stack, which implements both standardized and proprietary optimizations, accomplishes its goals through two main features: proprietary hybrid loss and latency-based algorithm (TCP Woodside), and rate-pacing capabilities. These features are able to constantly monitor network buffers—sending packets at rates that prevent buffer bloat and improving fairness across flows—while also preemptively slowing down to prevent congestion during heavy traffic. Once traffic lightens up, the algorithm speeds up to maximize the use of available bandwidth.

Conclusion

The Internet has gone through many changes since it was initially implemented on 213 fixed-line hosts in the late 1970s. With the number of Internet-connected devices now exceeding the global population, people speculate about the future speed of the Internet. As the world moves toward becoming completely mobile, new technology is being developed to handle the traffic across wireless networks.

Though many types of TCP stacks are available, only F5's properly provides for all three characteristics of an ideal TCP stack: having high goodput, minimizing buffer bloat, and allowing for fairness between flows. In addition to these unique functions, F5's TCP stack integrates with other F5 solutions. This allows multiple functionalities—including deep packet inspection, traffic steering, and load balancing—to be consolidated onto one platform.

[1] http://www.idc.com/getdoc.jsp?containerId=prUS24823414

[2] http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white_paper_c11-520862.html

[3] http://ee.lbl.gov/papers/congavoid.pdf

F5 Networks, Inc.
401 Elliott Avenue West, Seattle, WA 98119
888-882-4447 www.f5.com

Americas
info@f5.com

Asia-Pacific
apacinfo@f5.com

Europe/Middle-East/Africa
emeainfo@f5.com

Japan
f5j-info@f5.com