TCP Vegas Review

by Brakmo, O'Malley, and Peterson
– Kevin Roundy –

Feb 21, 2007

TCP Vegas's main objective is to improve on the TCP implementation that was packaged with Reno distribution of BSD Unix, specifically by improving throughput by detecting and avoiding congestion problems. Here are the main contributions of the paper.

- Better round trip time (RTT) estimates (and therefore tighter variances) resulting in better timeout estimates
- Quicker detection of lost segments by inspecting the segment's timestamp rather than waiting for a course-grained timer to go off to check for timeouts. Checks are made when:
  1. duplicate ACKs are received
  2. the 1st or 2nd ACK after a retransmission is received
- More careful resizing of the congestion window after a loss, so that the window size is not reduced multiple times for losses occurring during a single RTT interval.
- Spike suppression to alleviate oversending of data due to ACKs not being evenly spaced.
- Congestion avoidance rather than congestion detection. This is accomplished by calculating an expected throughput rate equal to the size of the congestion window divided by the best round trip time seen so far. The expected throughput rate is compared to frequent measurements of the actual throughput (once per RTT) to adjust the amount of data being set over the network.

In the rest of the review we'll focus on TCP-Vegas's congestion avoidance mechanism. If the difference between expected and actual throughputs (DIFF) becomes too small, there will be excessive congestion on the network and segments will start to be dropped, whereas if DIFF is too large, the available bandwidth of the connection is perhaps not being fully utilized. This suggests the use of an upper and lower threshold on DIFF, which the authors call alpha and beta.

The authors state that the main weakness in their approach is that alpha and beta depend on a good expected throughput estimate, which as we have seen, depends on a good minimum RTT estimate. In order to make the system more robust, either alpha and beta must be made responsive to network conditions (they are currently chosen statically), or the formula for expected throughput has to change. The formula for expected throughput assumes that the minimum achievable RTT does not increase, but it might change considerably if the path traversed by the connections is unstable. It might also be an overestimate if the network was experiencing high loads when the TCP connection is starting up (which is typically when the smallest RTT is found). This suggests that dynamic adjustments to the alpha and beta parameters may be required. Because of these

and other potential issues with TCP Vegas, the lack of an extensive real-world performance study is troubling.

However, while this paper leaves some questions unanswered, TCP-Vegas clearly showed significant improvements with respect to TCP-Reno, as highlighted by their effective use of graphs and performance measurements. The paper's graphs especially highlight the importance of congestion avoidance in any clean slate design. Assuming that the end-to-end principle is applied to a future internet design, we will not have the network telling its end-users how much data they should send, so we will still need to include congestion avoidance techniques at the end-hosts using similar techniques to those we find in TCP-Vegas.