The ISP Column
A column on various things Internet

August 2025
Geoff Huston

Congestion Control at IETF 123

The Internet Engineering Task Force (IETF) meets three times a year to develop Internet Standards and related best practices. At its July 2025 meeting in Madrid, several sessions explored the evolving role of congestion control in transport protocols and sparked the observations in this post.

Early models of packet networking used a hop-by-hop paradigm of control. Each intermediate device (a "router" in Internet parlance) would use a control loop with its adjacent neighbour and retransmit any frame that was not explicitly acknowledged as received by the neighbour. Such models were used by the X.25 protocol, and by DECNet in its DDCMP protocol.

IP replaced this "reliable link layer forwarding" with its "end-to-end" model, where there was just one control loop, between the source and destination hosts in the transport protocol (TCP). When am IP router is operating in a congested state such that its internal queues are full, it will silently discard all additional received packets until the queue pressure is relieved. The packet loss is ultimately detected by the destination host, who will send a control signal to the sender (typically in the firm of a duplicate TCP ACK when using TCP), which will trigger the sender to resend the dropped data.

This end-to-end control model was able to realise benefits in terms of speed and efficiency, but it also introduced some new control questions. The overall objective of the transport protocol is to act in unison with other concurrently operating transport sessions to fill the path's transmission capacity while not generating undue levels of packet loss (maximising efficiency) and at the same time adapt to the presence of other concurrent transport sessions to equitably share the network's available capacity between the various transport sessions (maximising fairness).

The initial work on the control models quickly converged on the AIMD model of flow control, using a linear level of continual inflation of the sending rate (Additive Increase) until packet loss occurred (which was assumed to occur at the point where both the path transmission capacity and at least one internal queue on the path was full), at which point the sending rate was halved (Multiplicative Decrease). Much of the subsequent work on this end-to-end control function ("congestion control") has concentrated on changing the details of the increase and decrease of the sending rate to extract greater levels of efficiency under more extreme network scenarios while still remaining basically fair.

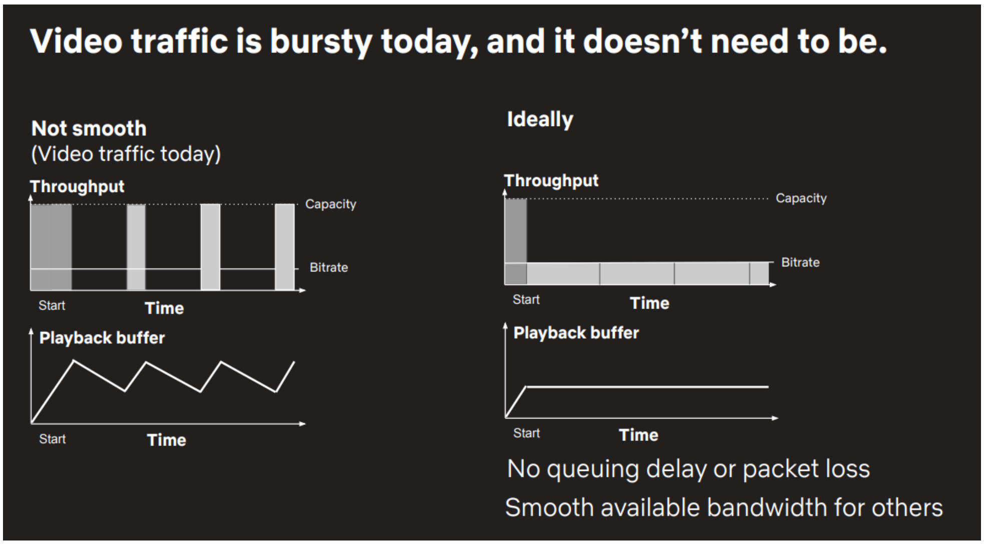
As well as this, the changing nature of the application environment has had a significant impact on the objectives of this work. These days it could be argued that the Internet is little more than a replacement for television in all its formats. And if you looked at the volume of traffic by application type you would probably find some supporting evidence for this argument. Estimates vary (as do measurement methodologies), but the various sources suggest some 65% to more than

80% of the digital content delivered to users consists of video material. There is the video component used in teleconference applications, such as Zoom, with the requirement for near real-time playback video, and streaming broadcast television which is generally more latency tolerant, and there are the various forms of pre-recorded video content, such as streamers. These different applications make different demands of the network. They all tend to be high volume, but as the application leans into near real-time (conferencing and some forms of streaming television) there is the requirement for low latency connections between source and destination.

Addressing the "Netflix Spike" Issue

For the video streamers, which are not necessarily constrained by attempting to operate with real-time playback, there appear to be two basic levels on controlling the demands on the network. One is by altering the bit rate of the video signal (adaptive bit rate, or ABR) and the other is selecting the congestion control algorithm being used. The worst case is that these two mechanisms can interfere with each other, leading to a spiral of decreasing ABR bit rates and lower quality streaming. TCP congestion control algorithms generally attempt to fair-share the common network resource with other concurrent TCP sessions. When the TCP session is source-limited (such as occurs with ABR driving down the bit rate) the TCP session may reduce its congestion pressure on the other simultaneous TCP sessions, ceding more of the common network resource to these other TCP sessions which in turn may have a negative impact on the video session which will then further downgrade the selected ABR rate. Better results can be achieved if congestion control (CC) works together with ABR to maintain a consistent flow pressure on the network.

The other aspect of current congestion control algorithms is that they achieve fairness with other TCP sessions by being relatively aggressive in their efforts to secure network resources. If all TCP sessions are equally aggressive, then the resultant situation is one of equal sharing of the network's resources across all sessions. If you combine this behaviour with the approach of treating a video session as a sequence of separate video "chunks" then each chuck will be sent across the network as fast as the network and the client device can sustain the transfer rate, and the session will then fall back to a quiescent mode while it waits for the client to complete the playback of previously loaded chunk to free off memory to allow the next chunk to be downloaded. This spiked traffic profile extracts a cost on both the session and the network. The spikes can drive the network into congestion loss, and the playback session will suffer an efficiency drop due to the retransmission events. The current thinking is that such spikes re best avoided, and the application should drive a steady pace of data traffic, by selecting an adaptive bit rate that matches the congestion control rate (Figure 1).



Video traffic is bursty today, and it doesn’t need to be.

No queuing delay or packet loss

Smooth available bandwidth for others

Figure 1 – Smoothed Video Streaming Traffic – From "Adaptive Bitrate & Congestion Control", Te-Yuan Huang, Netflix, CCRG, IETF123

This can improve session efficiency without any negative impact on session quality. Of course, this assumes that the server and client-side application can exchange service quality information and cooperate to set the server's smoothed sending rate at a level that is sustainable by the network and the bit rate used for the video content. It also should be noted that there are two control loops at play here - the network loop that is based on the round-trip time between server and client and the "chunk transmit" time which operates at a far lower frequency. The objective here is to smooth the sending rate across the entire time to transmit the chunk.

Transport "Pacing"

This is work is intended to document guidance to implementers working on incorporating pacing in transport protocols. Sender-side pacing of data refers to the sender attempting to evenly spread the set of to-be-transmitted packets across the current RTT interval. It has a number of distinct benefits. Pacing is not susceptible to various form of ACK compression or ACK thinning. The reduced reliance on network queues to smooth out sender-side burstiness reduces overall queue requirements which in turn can lead to lower network latencies. An approach, useful for video, is to pace at roughly the intended video rate, again removing the behaviour of "pulsing " the network with sharp bursts of traffic.

Given these benefits, you might wonder why sender-side pacing wasn't adopted as a default transport behaviour from the outset? I guess that the rationale was in the desire for transport protocols in end systems to be as simple as possible, and make the network perform the role of rater adaptation and pacing traffic flows. If the sender is passing data into the network path at a rate that is greater than the path capacity, then a queue will form at the bottleneck point, and that queue will impose a pacing discipline on the traffic at the bottleneck link that is exactly matched to the available capacity of that link. Subsequent experience has shown that reliance on network buffers to perform rate adaptation and pacing leads to the addition of noise to the implicit timing signals contained the ack-pacing behaviour of the transport protocol, which tends to impair the efficiency of the transport protocol. Current thinking on this topic can be summarized by the term "buffer bloat." Over-provisioning of buffers in networks leads to the formation of standing queues which add to latency and reduce the effectiveness of the feedback signals that are contained in the ACK stream. Under-provisioning of these buffers leads to premature packet loss and over-reaction by the transport protocol, leaving unused network capacity within the network. Given that optimal buffer dimensioning depends not only on the delay bandwidth product of the driven link but the number of simultaneous transport connections using that link, there is no statically defined optimal buffer size. Sender pacing relieves the pressure on the network buffers and mitigates their potential impact in terms of impairing transport efficiency.

Service Quality Measurement and Speed Tests

The Comcast network in the United States has deployed Dual Queue L4S and NQB (non-queue building) mechanisms in both the downstream and upstream parts of their access network. The existing test and measurement framework uses iPerf3 located into the CPE.

The test framework looks at the post-startup steady state TCP throughput using multiple speed tests in parallel. The assertion made in the presentation is that downstream Active Queue Management (AQM) using the DOCSIS PIE algorithm has improved average latency from ~150ms to ~25ms, which is clearly a significant outcome.

The Changing Role of Congestion Control

The bigger question raised here is: "What are we trying to control and what are we trying to measure?" In modern high speed access ISP networks there are few, if any, high speed applications that are attempting to shift large volumes of data over the network. So why are we still running symmetric TCP-based volumetric throughput tests across multi-Gbps access networks when there are few applications and host devices that can sustain such a traffic profile?

Are there other measurements and other tests that can provide a better match to the quality of the user experience in this environment? What about metrics of "responsiveness?" and how does such a responsiveness metric change under varying load? Perhaps our earlier fixation with achievable bulk data transfer speeds as the major comparative metric of access network is becoming somewhat obsolete in the face of deployment of high-speed access networks, as such metrics are not reflective of the usage profile we have today.

It appears that we are putting a different interpretation on what we mean by "congestion control" these days. Earlier efforts were directed at maximising the ability of a transport session to utilise available network capacity without overloading the network, and it achieved this by increasing its pressure on the network just to the point of overload (congestion-based packet loss). The constraints associated this approach are not to sustain this overloaded state, as this would lead to congestion collapse, and to be sensitive to the demands of other concurrent sessions so that one session does not usurp all available path capacity.

With the general rise in network capacity, maximising session throughput to the point of saturating available network path capacity is no longer the paramount objective of congestion control, and instead the objective appears to be to keep the network operating without being driven an overloaded state, if possible. The congestion control techniques that are being explored include sender pacing, application level sending constraints, the L4S framework with ECN, dual queues and active queue management, and delay-based control, such as BBR. The approach is to avoid the formation of standing queues, and to the extent possible react very quickly to the onset of queuing to avoid queue saturation and packet loss. That way latency is minimised, retransmits are also minimised and carriage efficiency and responsiveness is improved.

Disclaimer

The above views do not necessarily represent the views or positions of the Asia Pacific Network Information Centre.

Author

Geoff Huston AM, M.Sc., is the Chief Scientist at APNIC, the Regional Internet Registry serving the Asia Pacific region.

www.potaroo.net