The ISP Column
A monthly column on things Internet

March 2021
Geoff Huston

TCP Congestion Control at IETF 110

IETF 110 was held virtually in March 2020. These are some notes I took on the topic of current research activities in the area of transport protocol flow control at the meeting of the Internet Congestion Control Research Group at that meeting.

HPCC+: High Precision Congestion Control

In the early days of congestion control signaling, we originally came up with the rather crude "Source Quench" control message, an ICMP control message sent from an interior gateway to a packet source that directed the source to reduce its packet sending rate. It was unrelated to the transport protocol that the end host might be using, and intentionally operated at the level of the IP layer within the context of the protocol stack. It was a decent first effort at a rather complex problem, but it was always going to be replaced by successive refinements as we gained experience in this area. The IETF got around to formally deprecating the Source Quench mechanism in 2012 in RFC 6633, though it had lapsed into disuse around two decades earlier. Then there was Explicit Congestion Notification (ECN) for TCP (RFC 3168) where a gateway marked one or more packets in a TCP data flow when it was experiencing congestion and the ECN signal was echoed in the reverse TCP ACK packets so that the signal was passed back to the sender. It's an improvement without doubt, as it allows the sender to respond to the onset of congestion prior to the more disruptive signal of packet drop, but at the same time it's a single bit signal, so the signal can signal the presence of a congestion condition or not, but not much more.

It is always fascinating how we continue to refine silicon in ways previously considered impossible or completely impractical. New commodity ASICs have in-band telemetry ability. This can be used to formulate precise feedback for congestion control. The idea is to replace the coarse single-bit feedback in ECN with more information, including queue length, link capacity, timestamp and similar. Packets in the data forwarding path have switch telemetry attached to the packet by every switch, and this telemetry is copied to the corresponding ACK packet being passed in the reverse direction (Figure 1). The objective here is to eliminate the extended process of searching for a stable flow profile with a process of rapid adjustment to a desired profile, while at the same time making minimal queue demands as a latency reduction mechanism.

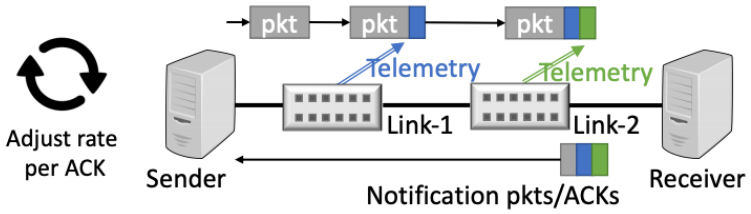

Figure 1 – HPCC+, Rui Mao et al, Presentation to ICCRG, IETF 110
https://datatracker.ietf.org/meeting/110/materials/slides-110-iccrg-hpcc-enhanced-high-precision-congestion-control-00

The environment assumed in this work is a datacenter where the switching elements in the datacenter are all operated by the same party and the packet markings can be trusted, and the end-to-end hop count

is limited. In a more heterogenous environment, such as the public Internet, this approach has many potential issues, but in a uniform environment where all switching elements can mark packets with current switch characteristics and where the end systems can trust this meta-data, then it is certainly possible for systems to quickly and accurately adjust their sending profile in accordance with their knowledge of the path characteristics.

QLOG

There are many tools for analysis of TCP performance. These tools typically rely on the observation that the TCP flow control parameters (stream sequence number values, window parameters and such) are placed into the TCP packet header in the clear, and an on-path observer can use this information to derive TCP performance.

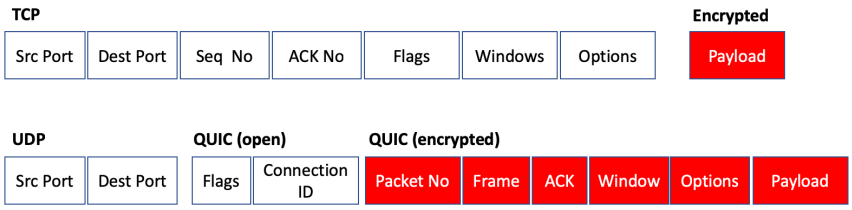

Figure 2 – QUIC encryption of Transport Control parameters

QUIC is almost entirely encrypted (Figure 2), and the only way you can perform a similar analysis of transport control performance is to gather the data from the end points and infer flow control state from the data on the endpoints. This would then allow network congestion events to be inferred from this data, and the response from the protocol can be analyzed.

This is the intent of the Qlog tool. The tool gathers parameters from the qlog endpoint logging facility in QUIC stacks and assembles the data to create a state inspection capability that is comparable to the TCP packet capture analysis tools that have been used in "open" TCP for decades (Figure 3)

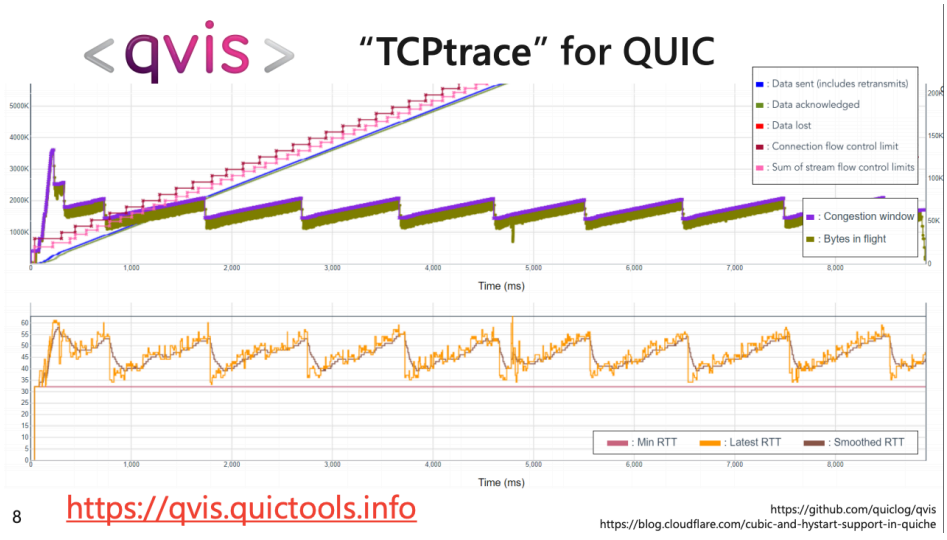

Figure 3 – TCPtrace for QUIC using QLOG tools
https://datatracker.ietf.org/meeting/110/materials/slides-110-iccrg-qlog-structured-endpoint-logging-for-encrypted-protocols-00

I suspect that there is a deeper observation behind this presentation, namely that as we lift up functionality in networking away from the common platform and load up applications with this function, then the only way to obtain visibility on what these applications are doing is to have the ability ot interrogate them directly.

MultiPath Transport and Congestion Control

Multipath has been a fascinating area of study. For example, mobile devices often operate with functioning WiFi and cellular radio connections, and the promise of having the transport protocol make use of the paths across both transmission systems has attracted interest from time to time. Sometimes its seen as a fast-failover approach, where the data flow uses a single path at any time, but other paths are help in a ready state to be used in the event of a failure of the primary path. In other cases, they are seen as a load balancing approach, where the data flow is spread across both paths. In TCP this load balancing and implicit congestion control coms from the independent TCP congestion control algorithms being used on the paths. However, UDP flows are more challenging. The IETF standardized the Datagram Congestion Control Protocol (DCCP) (RFC4340), allowing a congestion control algorithm to control the UDP packet flow rate for one of more UDP packet flows between the same two endpoints. This work reports on efforts to extend this DCCP approach into the realm of multiple paths

DCCP was intended to allow a collection of UDP flows between the same endpoints to behave "fairly" with respect to other simultaneous flows. At its crudest this would impose a rate limit of UDP flows when the network is experiencing congestion, and the rate management would be largely the same as a TCP-controlled flow of a similar flow volume. At the same time the flow should not be penalised unfairly. Multipath introduces a further consideration, namely that when a path experiences congestion the multipath control should be able to use alternate paths to maintain the flow where possible. This study suggests that there is a complex set of interaction across a number of factors, including the scheduling and reordering mechanism that is used, the choice of the control congestion protocol used, the placement of functionality of the control proxy, and of course the path characteristics. So far they have observed that BBR is a better congestion control proxy for MP-DCCP than loss-based control systems such as Reno.

BBR

Work on refinement of BBR continues. BBR is a flow control algorithm that is based on variations in end-to-end delay, as compared to the more "traditional" drop-based flow control algorithms. The initial release of BBR showed the promise of this approach. BBR flow control was able to achieve high throughput with minimal demands on network buffers. However, it did not play well with others, and it was possible for a BBR flow to starve concurrent loss-based flows of any capacity, or to be locked out by these flows. Subsequent effort has been directed to tune BBR to be a little more sensitive in its adjustment to other flows.

BBRv2 has been deployed within Google for all internal TCP traffic. BBRv2 uses the parameter of the estimated path bandwidth and minimum RTT to determine the point of the onset of queuing, and now incorporates ECN and packet loss as further control signals. They have seen latency reductions at the tail for RPC traffic. As well as internal traffic Google are looking at deploying this for YouTube and google.com externally directed traffic, with further measurement and tests underway.

There is an "alpha" release of BBRv2 Linux platforms available at https://github.com/google/bbr/blob/v2alpha/README.md, rebased to a more recent version of Linux with a few bug fixes in addition to the rebasing.

The Google group is working on BBR.Swift that leverages the Swift control protocol https://dl.acm.org/doi/abs/10.1145/3387514.3406591). This is a protocol that uses the network round trip time as the primary congestion signal, which makes the protocol react quickly to the formation of large queues within the network, while being more tolerant of short queues. As part of this work there is a proposal to include extensible timestamp options for TCP (draft-yang-tcpm-ets). It's not yet HPCC++, but the use of timestamps with microsecond granularity, and the inclusion of the delayed ACK processing delay and packet echoing delay would allow a flow controller to make a more accurate measurement of network delay as compared to protocol and host processing delay. Like HPCC++ the underlying notion is that more information about the end-to-end behaviours and the interaction between packets and networks result in more efficient flow control procedures.

QUIC 0-RTT Transport Parameters

There has been much work in the area of so-called 0-RTT TLS connections, where the security association of previous sessions can be used by the same pair of endpoints for a new session. This avoids the repeated set of handshakes for each new secure session.

Now the concept of storing flow behaviours between a sequential set of flows is nothing new for platforms. Many platforms implementation s of TCP were able to cache the flow control parameters, and when a new session was started then the platform would kick in with the flow parameters from the previous session. The talk at this session of the CCRG proposed to use a similar mechanism for QUIC sessions. I'm not sure that there is much that is novel here in this work.

This is not the entirety of the IETF work in congestion control, and the TCP Maintenance Working Group (TCPM) session had a number of presentations that are also of interest to congestion control.

Enhancements to ECN

ECN is a marvellously simple yet elegant way for the network to signal to the endpoints that the flow is passing through a congestion point (or points). The mechanism (RFC368) uses 2 bits of the IP header, where one bit is used to signal ECN capability and the other used to indicated congestion. These bits are echoed back in the TCP ACK packets using two further bits of the TCP header.

Of course, there is a difference between elegant simplicity and crude inadequacy. ECN as described in RFC3168 is certainly simple, but is it enough? It can only perform one signal each round trip time and can only signal the presence of a congestion point, without being able to indicate the number of such points that may exist on a path. A proposal to augment the ECN mechanism. "Accurate ECN" has been slowly progressing through the TCPM Working Group for the past six years and the working document is now in its 14th revision. The proposal uses a new TCP option to count the number of congestion events in a path and the byte count of traffic that have been impacted.

There is a massive level of inertial mass in TCP these days, and proposals to alter TCP by adding new options or changing the interpretation of existing signals will always encounter resistance. It is unsurprising that BBR stayed away from changing the TCP headers and why QUIC pulled the end-to-end transport protocol into the application. Both approaches avoided dealing with this inertial stasis of TCP in today's Internet. It may be a depressing observation, but it's entirely possible that this proposal to change ECN in TCP may take a further 5 or more years and a further 14 or more revisions before the IETF is ready to publish this specification as a Standards Track RFC. As to the time to get this widely deployed in the public Internet, well that's anyone's guess!

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