Lossless Congestion Control

Dave Taht
Director, Bufferbloat Project
davet@teklibre.net

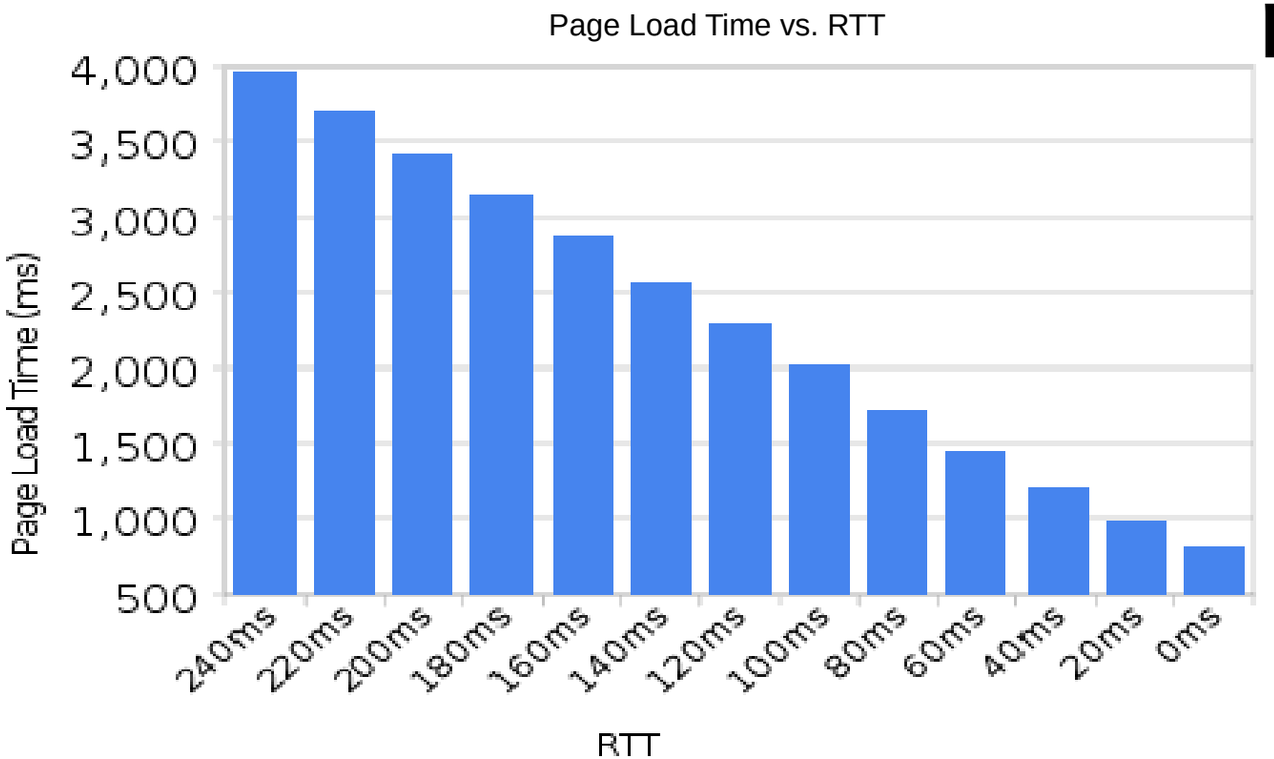
Plumbing the MetaVerse

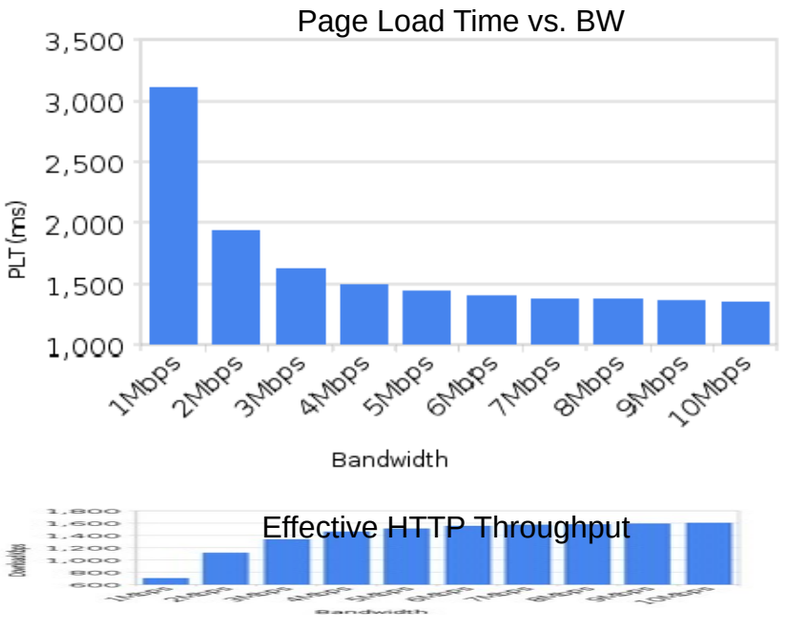
- I’m trying to describe some rapidly deploying improvements to the internet’s underlying infrastructure, a “Best Effort” network, where “Best” often had seconds of queuing delay.
- Solving old problems the Net’s always had, transparently, like:
  - TCP Global Synchronization
  - RTT independent Bandwidth Fairness
  - Deterministic inter-flow latencies
  - Excessive queuing latency and jitter from “load”
  - Packet loss used for congestion control
  - Reliable data delivery in adverse conditions
  - The most current data in adverse conditions
- Using FQ and AQM techniques that are backward compatible and increasingly widely deployed across all network stacks, computers, routers, and isps.
- Adding in one new feature requiring e2e co-operation, Explicit Congestion Notification (ECN), for lossless congestion control.
- All this stuff makes for a better MetaVerse. But it’s just plumbing.

Tackling fallacies first

- Bandwidth is not speed
- Network transfers are bound by physical round trip time, queuing delay, and retransmits
- Today’s internet does congestion control (responses to overload) via packet loss.
- Every potential bottleneck link can use a fq + aqm queuing algorithm to reduce queuing delay, absorb transient shocks, and reduce or eliminate packet loss. See: rfc7567

Web PLT depends on **RTT not bandwidth**





- Page Load Time is sensitive to *round-trip* latency
  - Google data shows 14x multiplier
    - +200ms RTT = +2.8 seconds PLT

- Diminishing returns from increased data rate
  - Page Load Time at 10 Mbps almost indistinguishable from 6 Mbps

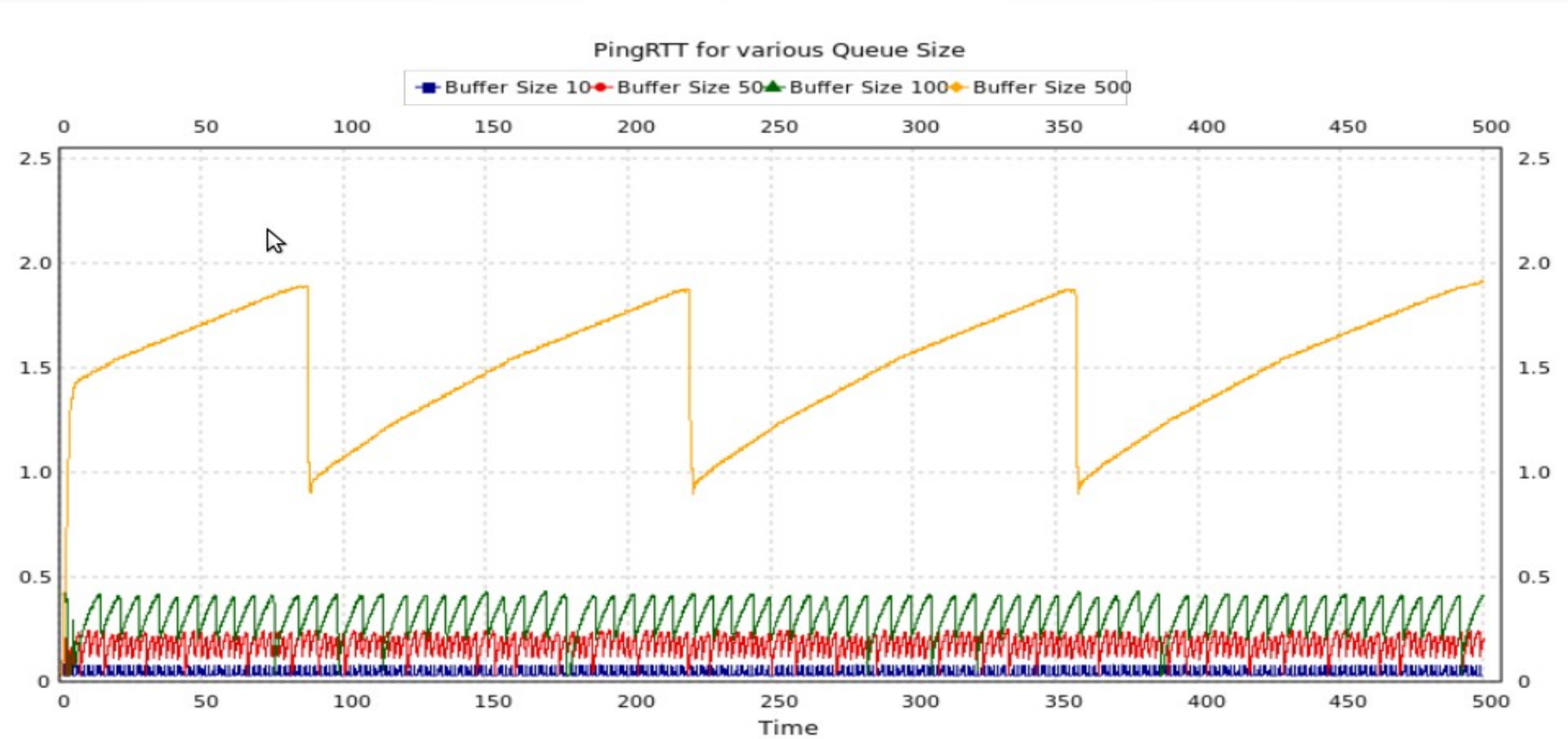
Source: SPDYEssentials, Roberto Peon & William Chan, Google Tech Talk, 12/8/11

Gaming, DNS, VOIP traffic are even more sensitive to physical and queuing RTT. Videoconferencing, and VR, especially, requires a smooth stream of packets, and constant updates.

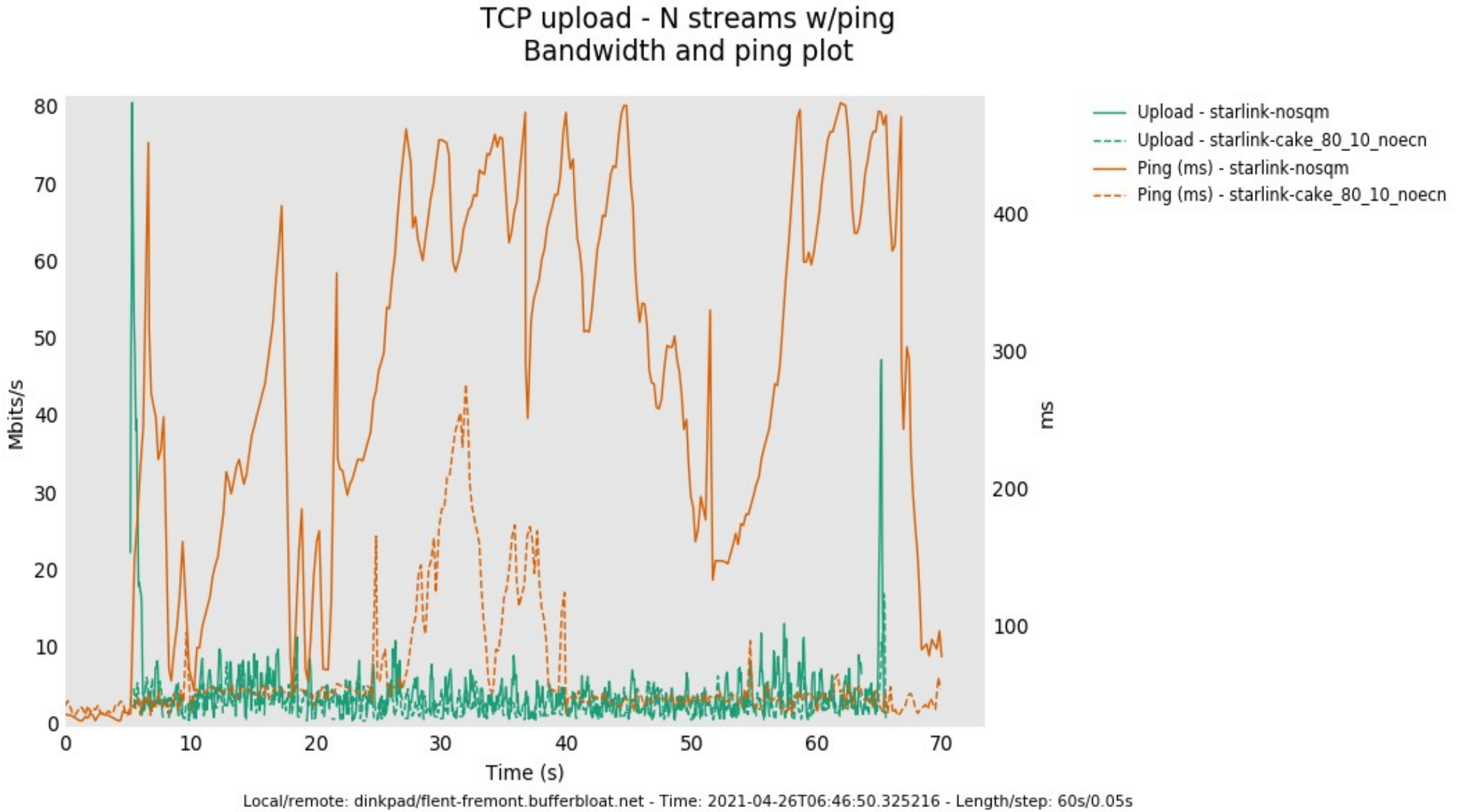
Internet Congestion control

- Queues across the internet are largely “tail drop FIFOs”.
- Packet transmissions are lossy and must be acknowledged and retransmitted for **reliable transports.**
- To probe for available bandwidth, protocols send ever more packets per round trip until the buffers overflow
- When a packet is dropped or marked
  - The sender slows down the rate by ~½
  - It costs at least one round trip to fill in the missing data.
  - Loss can happen for other reasons than congestion
- ***Some* queuing is required**
- Too much queuing leads to excess latency (aka “Bufferbloat”)
- Tail drop is the worst (but simplest) form of queuing

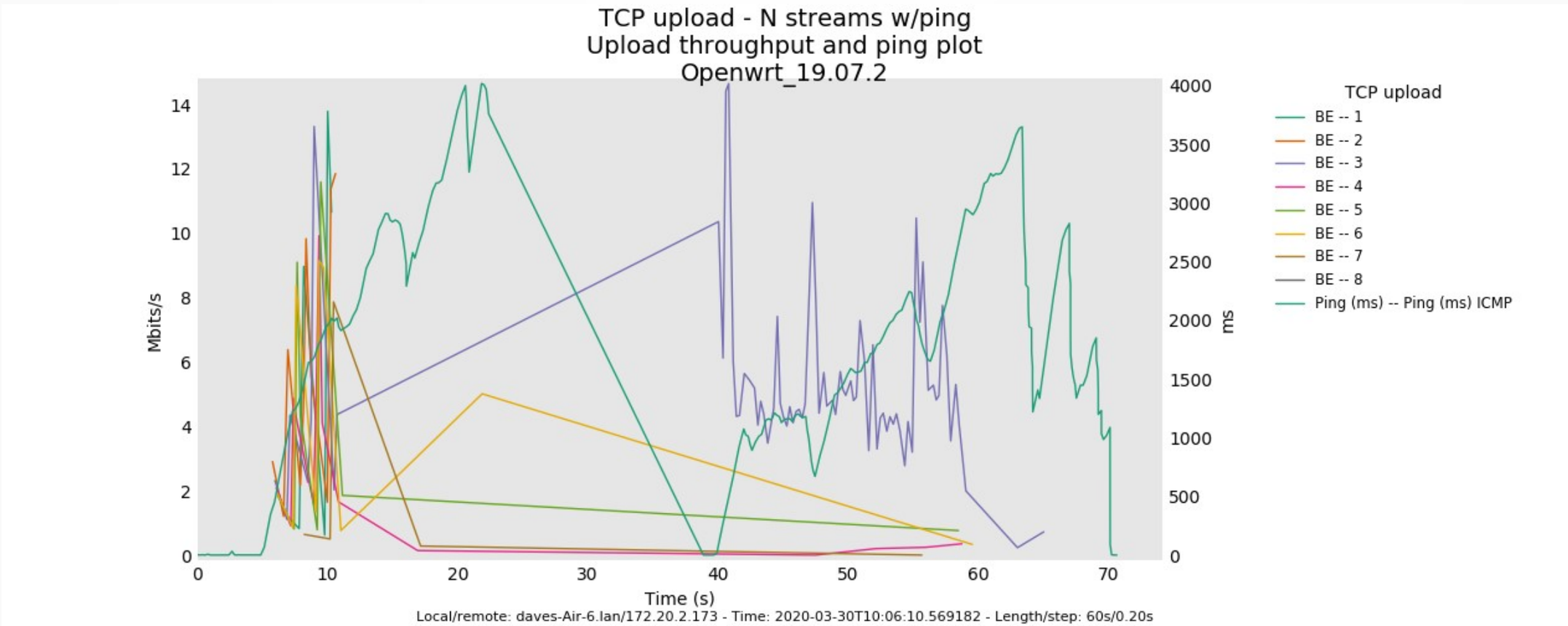
The TCP "Sawtooth"



Starlink's Working Latency



Wifi and LTE for many today



FQ and/or AQM technologies

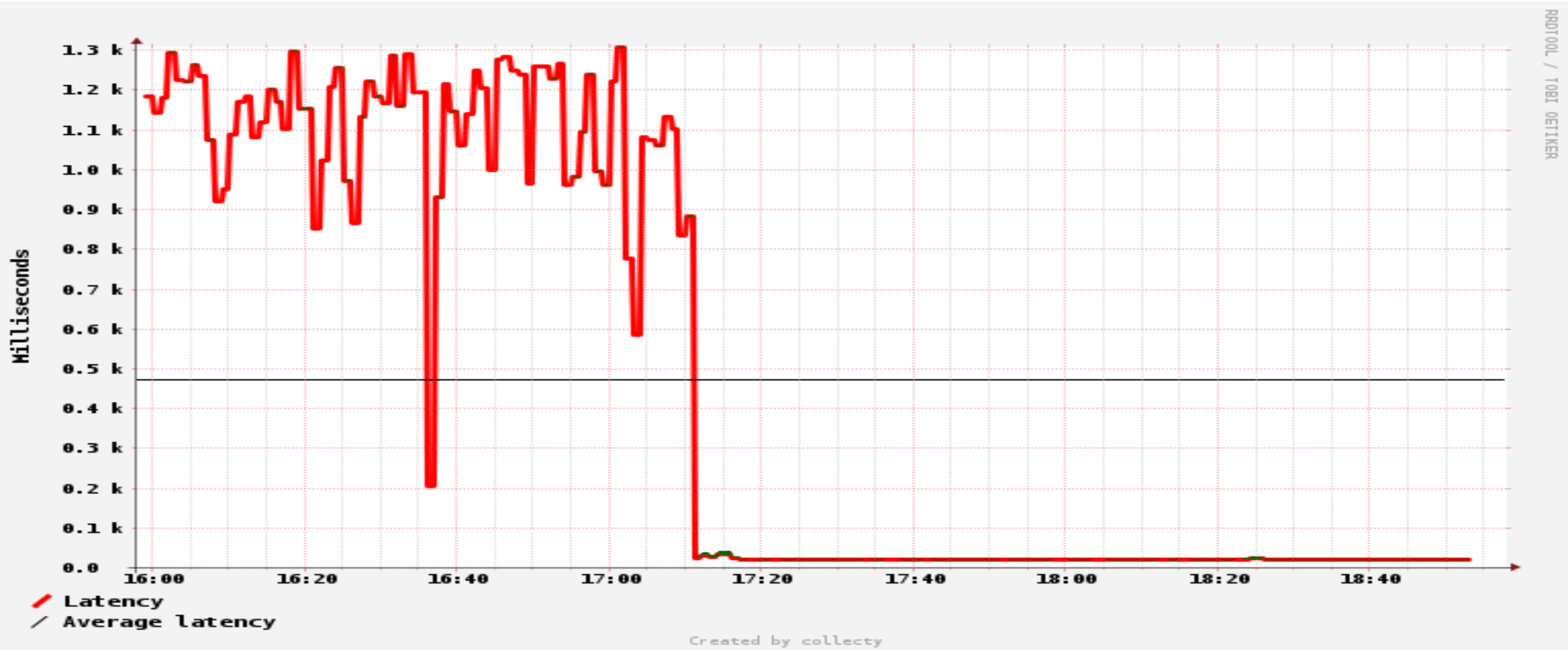
- If you enable “smart queues” (ubnt, mikrotik, vyatta, evenroute, many others), or “optimize for gaming and videoconferencing” (eero, google, many others)…

  What’s underneath is an advanced network queuing algorithm algorithm called fq_codel (rfc8290,rfc8289). Or a later variant (sch_cake).
- To take control of your queuing away from your ISP, it requires configuration of up/down slightly lower than what the ISP provides. Some ISPs are now turning it on automatically.
- fq_codel is the default queueing algorithm for Linux, OSX, iOS
  - Where it needs no configuration. It just works!

Flow Queuing (rfc8290)

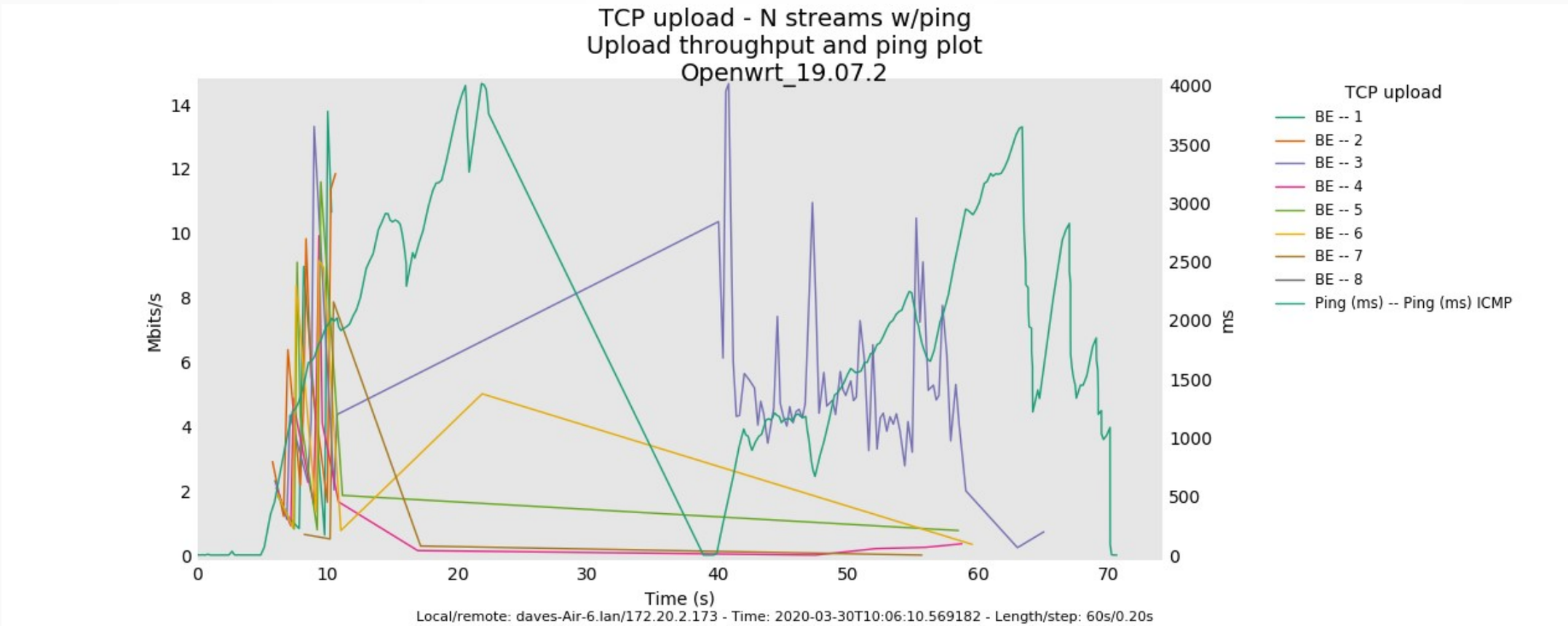
- 5 IP tuple fair queuing derived from “DRR++”
  - A sparse flow optimization – zero queuing delay for request/response protocols, C&C packets, voip
  - Multiplexes all forms of traffic together
- Codel or Cobalt AQMs
  - Head drop (or marking) rather than tail drop
  - Based on “Time in queue” so works with HW flow control
  - Deterministic, predictable in the face of indeterminate changes in load
  - Aims for 5ms of queuing latency while absorbing bursts
  - Replacement for the RED AQM algorithm
- Near zero queuing delays for sparse packets
- Target of 5ms for big flows.
- RFC3168 lossless congestion control enabled by default (linux/bsd only)

ADSL modem w/FQ_Codel with ethernet pause frames

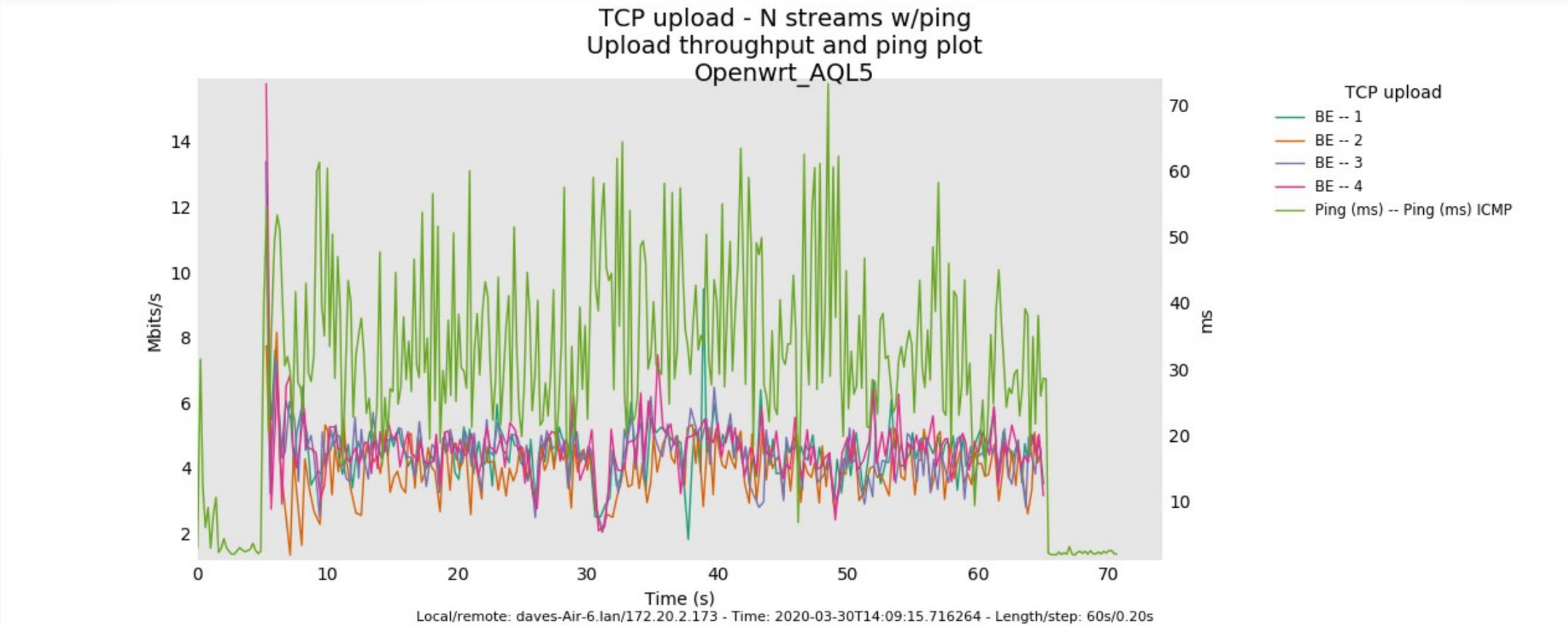


- http://planet.ipfire.org/post/ipfire-2-13-tech-preview-fighting-bufferbloat

Wifi and LTE for many today



Ath9k,10k,mwl, intel today



Explicit congestion notification

- Two bits in the IPv4 and IPv6 headers set initially by the originating program
  - 00 – My protocol promises to respond to drop
  - 10 – ECT0 My protocol promises to respond to marks
  - 01 – ECT1 Undefined, competing L4S and SCE proposals in the IETF
  - 11 – CE - Congestion Experienced – a router running out of space marked the packet asking the other end to slow down
  - RFC3168 has a defined response of a rate reduction of ½ to a loss or a CE mark.

Clients can turn ECN on

- TCP stacks
  - Linux: sysctl -w net.ipv4.tcp_ecn=1
  - OSX: sysctl -w net.inet.tcp.disable_tcp_heuristics=1
  - Windows: netsh interface tcp set global ecncapability=enabled
- Any other application – be it VR/AR videoconferencing, or other type of flow transiting an ECN enabled AQM link
  - Can just set ECT1 on their packets (setsockopt)
  - And respond (sanely) to CE marks
  - And (almost) never drop packets
  - And/or use delay based congestion control with FQ on that path.

Some stats

- Linux: $ tc -s qdisc show
- IOS/OSX: $ netstat –qq -I the_interface_name
- NetBSD/Freebsd
- NS2/NS3/P4/OFS/ and various other versions
- If you like the results, always looking to extend the tech into new wireless systems/network stacks, new applications, seeking grants, improving our primary test tools: see https://www.bufferbloat.net
- Or: https://www.patreon.com/dtaht

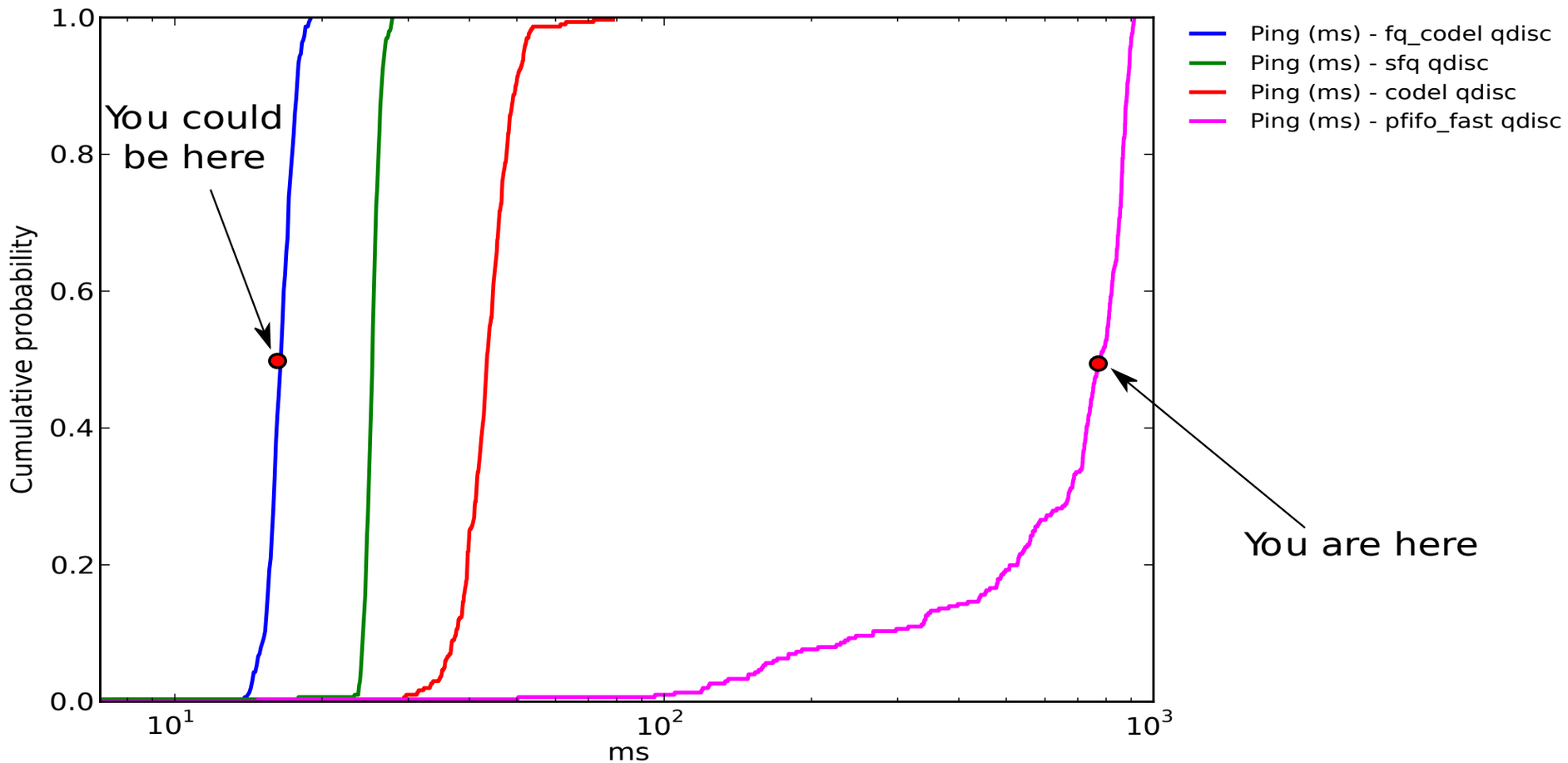
New Closed Loop Systems

- FQ + AQM Rigorous endpoint awareness of explicit congestion marking techniques leads to a positive result
  - Too old (stale) real time data can be dropped
  - Packet loss can be treated as “noise”, and not a signal to slow down. Wireless technologies benefit.
  - FQ means: Fixed rate senders below the sum of the bottleneck/flows never drop packets
  - No retransmits needed for ECN aware protocols

Pending Lossless Research

- Shallow multipacket marking (DCTCP/L4S or SCE)
  - Marks many packets at a very shallow queue depth
  - Flows are signaled to “slow down” by a fraction
  - L4S - Incompatible with RFC3168
  - SCE – extends RFC3168
  - L4S is presently “winning” in the IETF
- RFC3168 enabled videoconferencing (galene.org)

Working Latency reductions (if you turn the right stuff on)



dave.taht@gmail.com

FCC 05/29/2013

Some useful links on “working latency”

- Recent Broadband Internet TAG Report:
  https://www.bitag.org/documents/BITAG_latency_explained.pdf
- Packets as people:
  https://blog.apnic.net/2020/01/22/bufferbloat-may-be-solved-but-its-not-over-yet/
- FQ_Codel on WiFi:
- Bufferbloat and Beyond:
  https://bufferbloat-and-beyond.net/
- Recent Updates
  https://www.youtube.com/watch?reload=9&v=AjZXx4N1tmY