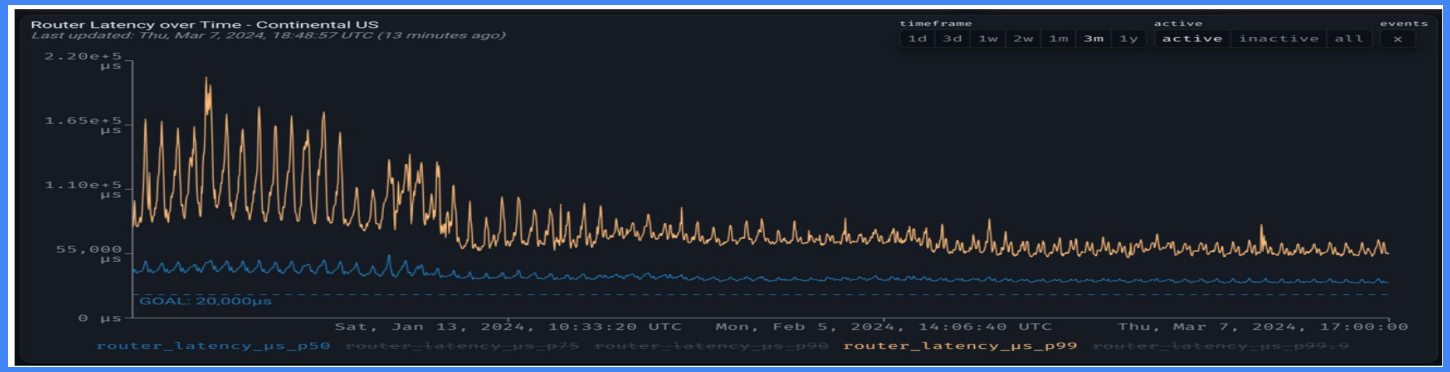
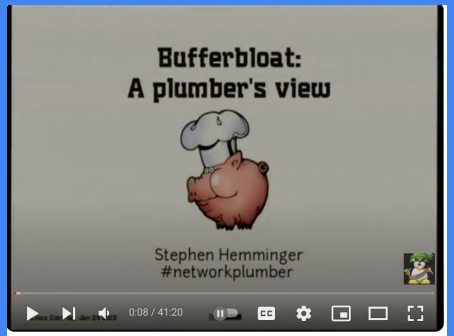


State of the (Buffer)Bloat

Dave Taht
Co-founder, Bufferbloat.net
Chief Science Officer, LibreQos.io
Linux Plumbers 9/18/2024



[Linux.conf.au 2013] - Bufferbloat from a Plumber's point of view

Executive Summary

Most ISPs still have bufferbloat

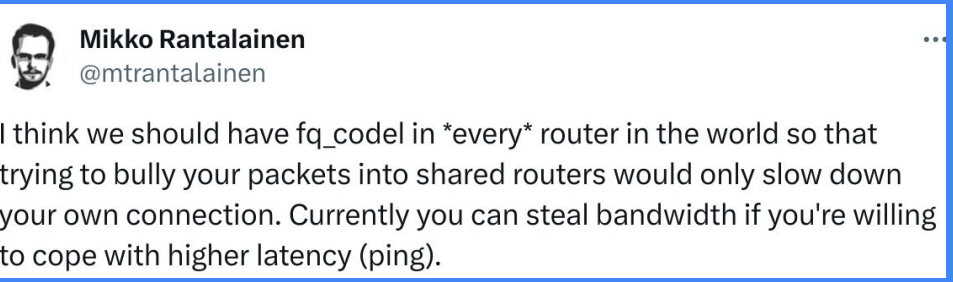
(But there are some bright spots in QoE, WiFi, and Starlink)

Smart users use SQM to better manage their network[1]

Everyone else is just used to “random” delays and jitter…

1 (htb+fq_codel, CAKE, sqm-scripts, “smart queues”, “adaptive QoS”, etc, etc)

About Bufferbloat



“Bufferbloat is the undesirable network jitter and latency that comes from overlarge and badly managed buffers across the bottleneck links of the internet. “ - Wikipedia.

Rapidly getting the most bandwidth possible across a network connection conflicts with the needs of interactive traffic - DNS, keystrokes, request/response protocols, VOIP, gaming and videoconferencing.

Solutions are in the Linux kernel today like BQL, fq_codel, TSQ, BBR - But there has been new challenges, and getting the existing solutions deployed is just barely started across the edge of the internet. Bufferbloat is still often measured in *seconds* across many edges today.

Today’s topics

The State of the Bloat

Kernel Progress & New Tools
Bloat - Cloud, Ethernet, 5G, WiFi, Starlink
New Bufferbloat.net Projects

Source: Where has the time gone? A summary of the FCC Measuring Broadband America report 2013-2023

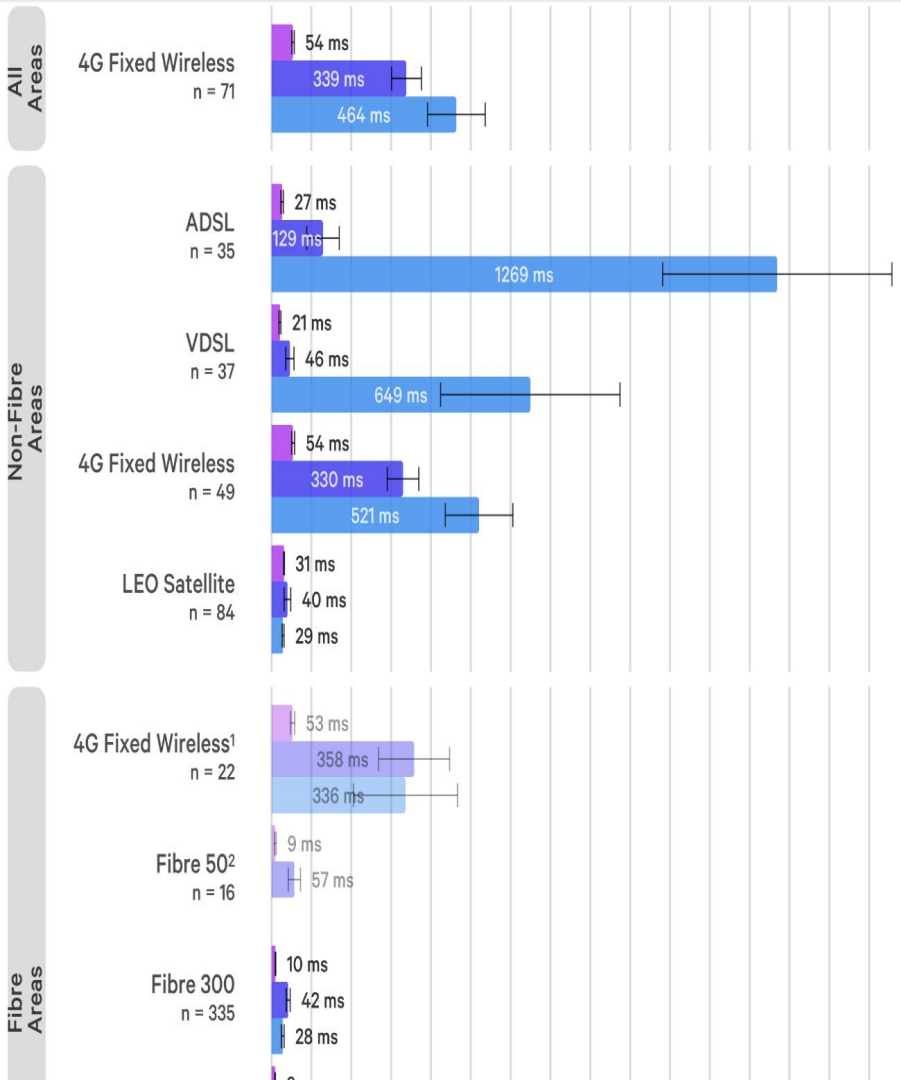
TABLE III: **Summary of Latency Measurement Results (Jul'2023).** Organized by row representing the observation presented in MBA test data (Idle, LUL Downstream (DS), LUL Upstream (US)), the minimum (min), maximum (max), and selected percentiles represent a field of latency results calculated across Cable, DSL, and Fiber access technologies. The 95th percentile column is highlighted to support the discussion in section V.

	CABLE (*ms*)						DSL (*ms*)						FIBER (*ms*)					
Test	**min**	**50th**	**90th**	**95th**	**99th**	**max**	**min**	**50th**	**90th**	**95th**	**99th**	**max**	**min**	**50th**	**90th**	**95th**	**99th**	**max**
IDLE																		
RTT_{min}	0.1	12.3	24.5	29.8	57.8	188.5	0.1	23.9	44.9	53.3	81.7	679.8	0.2	7.2	16.8	23.2	55.7	453.9
RTT_{max}	2.1	25.4	49.4	70.7	184.3	479.0	1.6	34.4	101.6	151.0	278.9	957.4	1.4	12.8	27.9	46.1	119.4	626.1
RTT_{avg}	1.9	16.8	31.1	37.1	66.5	920.7	1.3	26.8	50.8	60.4	98.3	843.3	1.3	9.7	18.7	27.7	62.5	590.7
LUL DS																		
RTT_{min}	0.1	13.2	25.7	30.7	60.4	1100.7	1.0	25.3	46.5	55.1	94.9	1558.4	0.9	8.1	17.3	23.7	51.6	970.7
RTT_{max}	2.5	69.1	248.9	387.8	806.9	2994.3	1.7	79.6	469.0	790.9	1826.2	2995.3	1.6	26.9	125.6	216.8	1040.3	2995.8
RTT_{avg}	1.9	37.2	119.6	174.6	411.9	1791.3	1.3	56.8	323.1	469.1	1113.9	2477.4	1.3	18.1	81.6	106.7	372.3	2022.0
LUL US																		
RTT_{min}	1.0	15.0	27.4	32.6	62.6	864.3	1.0	23.8	45.1	53.6	92.4	1836.6	0.8	8.3	17.3	22.3	50.3	833.8
RTT_{max}	2.0	142.8	445.4	766.7	1914.7	2990.0	3.1	256.0	1226.	1603.5	2527.3	2998.5	1.4	22.4	127.4	246.9	824.6	2982.3
RTT_{avg}	1.6	47.0	236.6	391.1	962.3	2524.8	2.0	179.0	864.8	1143.6	1857.9	2629.9	1.5	16.3	74.2	123.3	570.8	1652.7

2024 New Zealand Broadband Report

Basically shows the same buffer size across all technologies and speeds just as the original NetAlyzr results from 2012.

:(:(:(:(:(:(:(:(:(:(:(:(



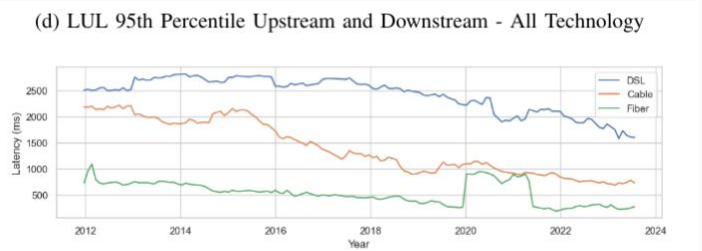
Vs the FCC on the Latency Front

“We argue that to best serve the people of the United States, the Commission should balance its near-term efforts on achieving internet resilience and **minimizing latency**, instead of only increasing “speed” or “bandwidth”.

Calls for further bandwidth increases are analogous to calling for cars to have top speeds of 100, 500, or 1000 miles per hour. Without calling also for better airbags, bumpers, brakes, or steering wheels, (or roads designed to minimize travel delay), these initiatives will fail (and are failing) to meet the needs of present and future users of the internet.” -

[Bufferbloat.net’s 2023 FCC NOI Response]

We’re going to keep doing that in 2024 - more signatures wanted! (we’ll gladly take on more regulators)



Test Tools of the trade

Flent v2.2 - landed last week!

Iperf2 - many new features!

Crusader - Rust rrul on everything

Xtcp2 - Monitor TCP_DIAG in docker

networkQuality - from Apple

WtBB?

All the speedtests have bufferbloat

Metrics now!

“It is wrong to suppose that if you can’t measure it, you can’t manage it – a costly myth.” - E.W. Deming

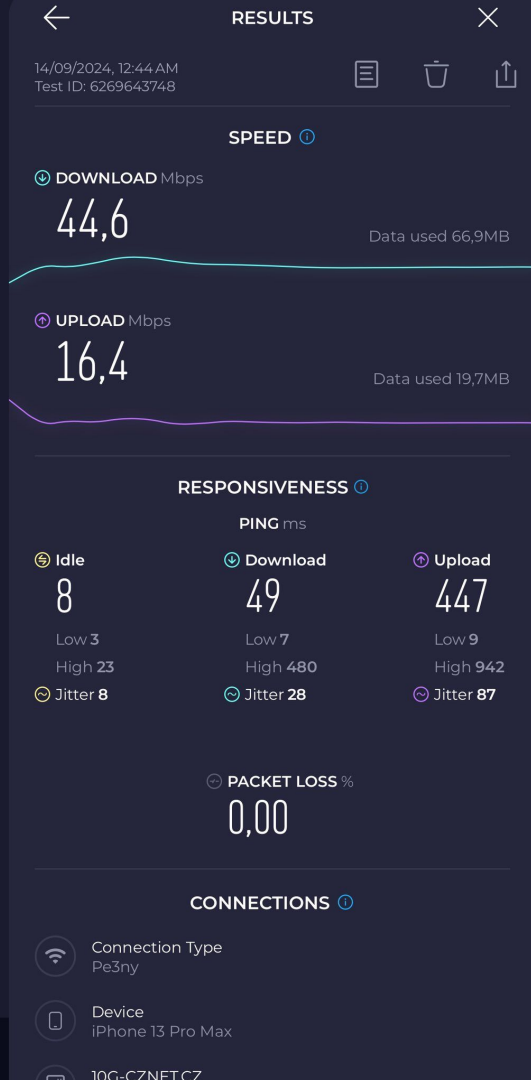
https://flent.org

https://sourceforge.net/projects/iperf2/

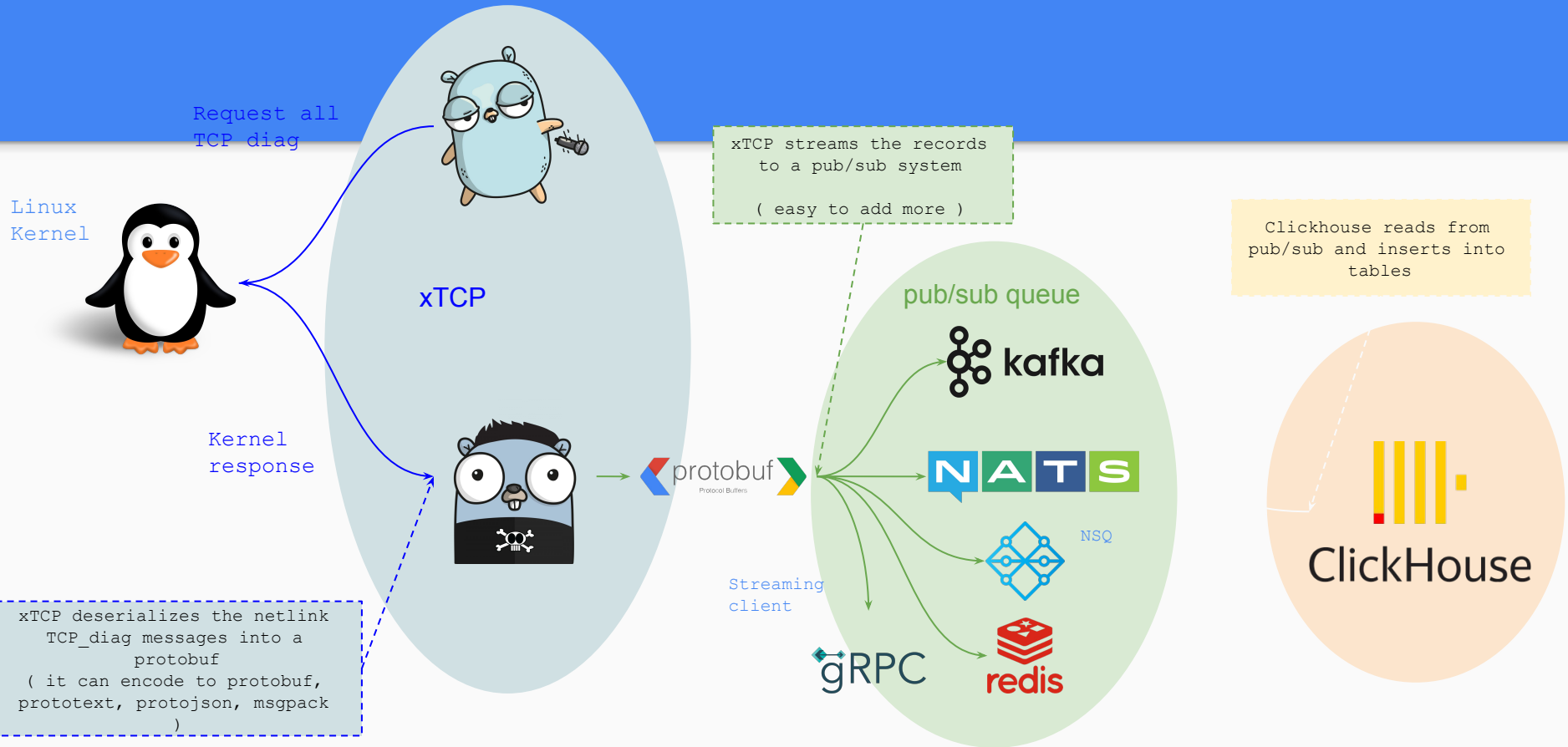
https://github.com/Zoxc/crusader

https://github.com/randomizedcoder/xtcp2

https://github.com/network-quality



Monitor Docker RTTs with xTCP2



Pesky queues (BQL)

Fq_codel tries by default to have 5ms latency across all the queues it manages. It has no control about what happens underneath.

I’m delighted to see BQL make the virtio-net driver and good results from netperf ‘s TCP_RR vs TCP_STREAM under load. Caveat:

BQL typically maintains 2 TSO’s worth of data in each queue. * 64 Cores!

Bytes = time = 8 ms of data stuck there at 10Gbit

…. And this does not count the data lying in additional queues from hypervisors, VMs, or containers, proxies, SND_BUFs… any time the cpu is about to outrun data to the wire.

Other Pesky queues (NAPI)

NAPI_WEIGHT(64) * GRO (24k) * Cores (64) =
Over a ms waiting to get serviced by the OS (at 10Gbit)

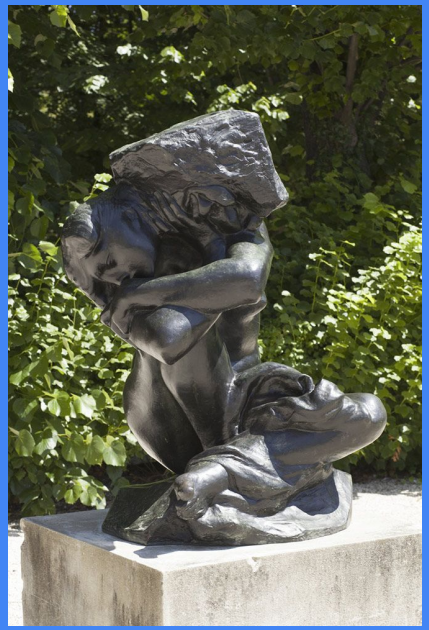
And this does not count the additional queues from hypervisors, VMs, or containers.

I would really like to break the 1ms barrier!

And please note these are 10Gbit numbers… where the edge of the internet barely runs over 100Mbit these days.

Container lesson:

Always measure throughput and latency!!!!



Source:
https://github.com/cilium/cilium/issues/29083

middaywords commented on Apr 19

For current design, can we alleviate the latency problem by setting a smaller drop horizion (smaller queue length) code? currently it's set to 2 seconds by default. with many tcp flows in one pod, it causes delay about 2 seconds for some packets. if we set it to a smaller value to make packets drop early, we may have a smaller latency I think.

dtaht commented on Jun 2

2 seconds for within containers on the same device is kind of nuts...

```
Method          | Avg Latency
-               | -
with-ECN        | 3.1ms
without-ECN     | 2247.3ms

Fixes: cilium#29083

Signed-off-by: Kangjie Xu <kanxu@ebay.com>
```

But fq_codel deployment to the edge was taking forever…

How do you get 350 lines of code running on every potential bottleneck router in the world?

We put fq_codel into open source in 2012… made it OpenWrt's default, talked about it… wrote about it… standardized it (rfc8290), got hackers to apply it… trained up reddit, slashdot, & hackernews folk about bufferbloat… made it Apple's and Linux's default… shipped CAKE… massively improved WiFi…

But as for hardware that ISPs deployed… an endless wait. Early successes like free.fr and Google Fiber and Eero went by but…

2017: The QoE middlebox market arises

Preseem - HTB: highly tweaked fq_codel

Bequant: L3-7 + fq_codel

Paraqum: dpdk + CAKE (I think)

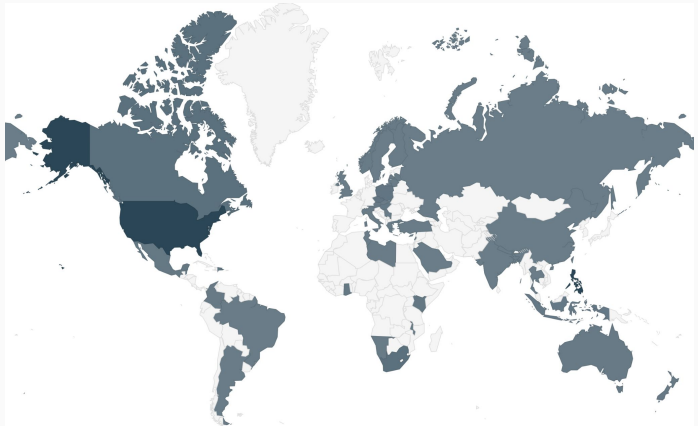
LibreQos: eBPF, HTB, CAKE (Started 2021)

Together we have less than 3% of the entire ISP market. Established players like Sandvine seem to be asleep.

About LibreQos

World beating bufferbloat solution - leveraging Rust, C, python, eBPF, and CAKE

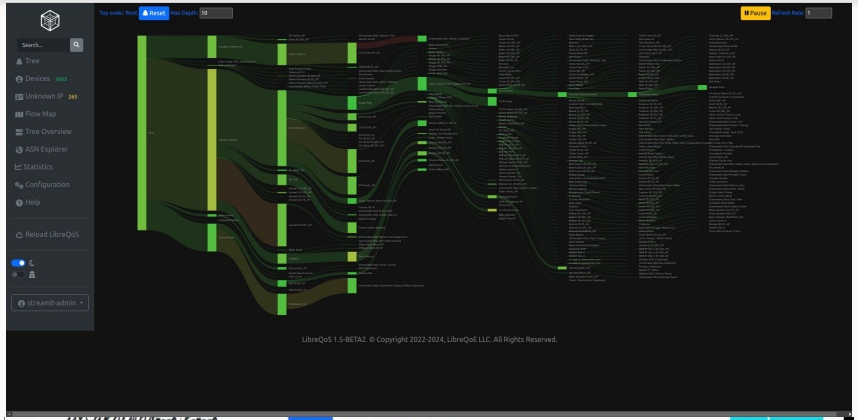
Core engine is Open source. Over 175 ISPs using it today! 30k subscribers/box.

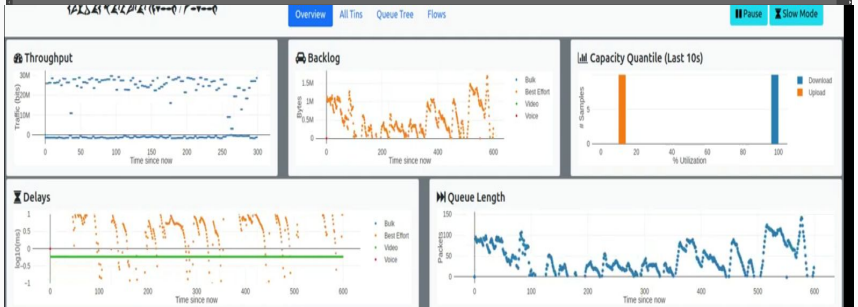


🇺🇸 (24 states) and counting!

Supported by NLNET's NGI0 Fund, and users

Really great monitoring tools too!!





LibreQos can model and manage a network 8 hops deep. It has all kinds of cool stats like this sankey, which shows latency issues across dozens of towers.

It has uses far outside the ISP world!

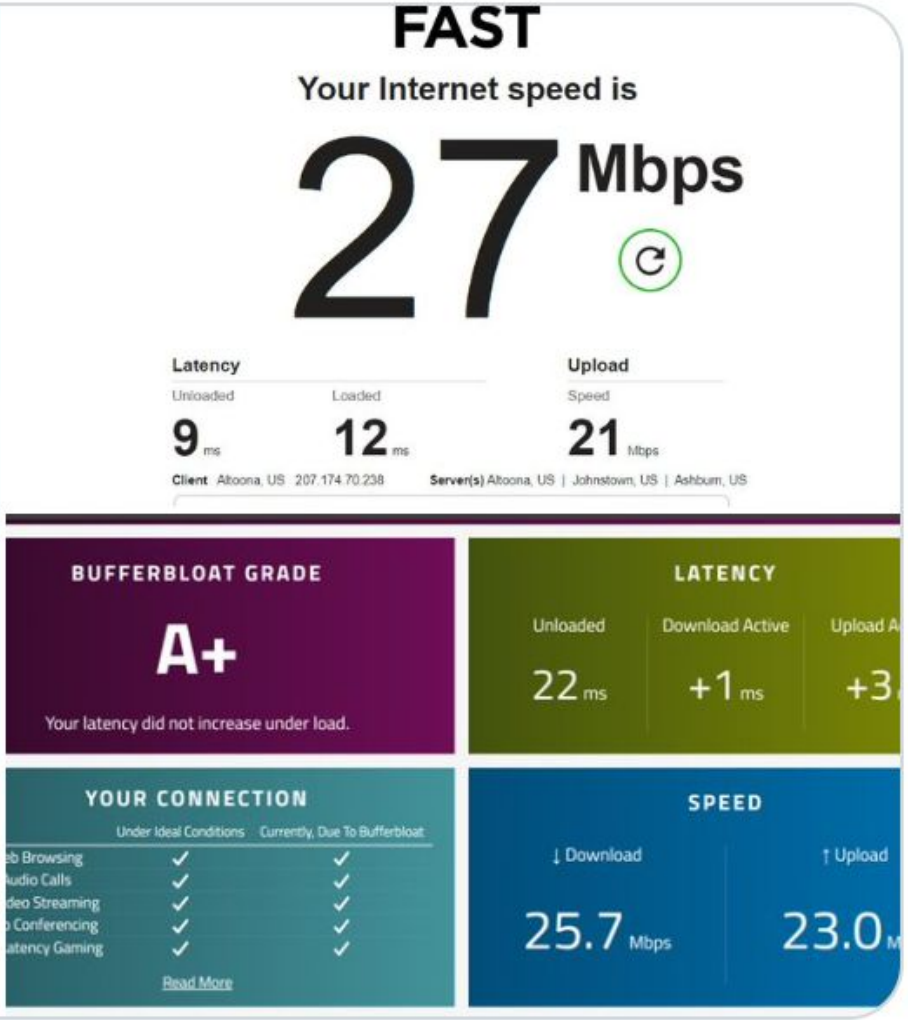
Fastest & cheapest way to debloat ISP networks we know of!

And it is Open Source!

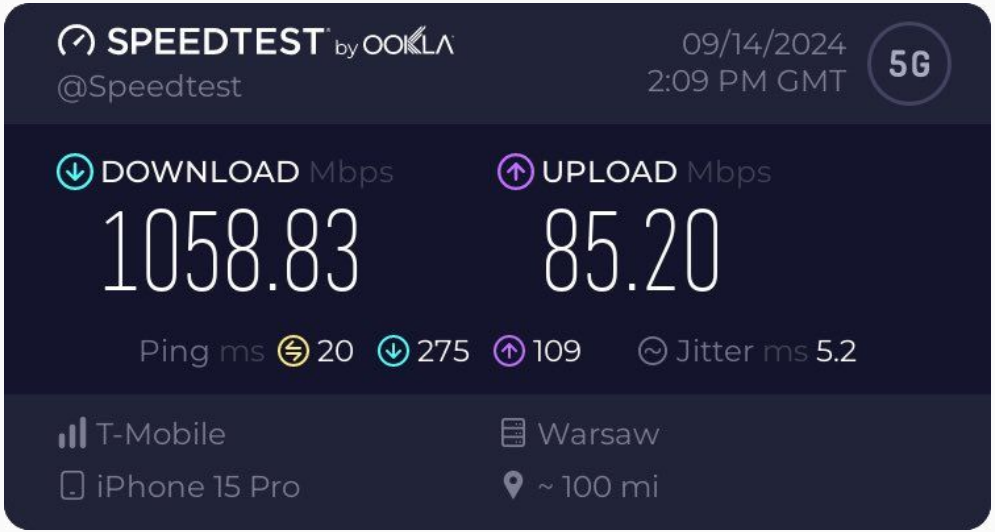
With a QoE middlebox…

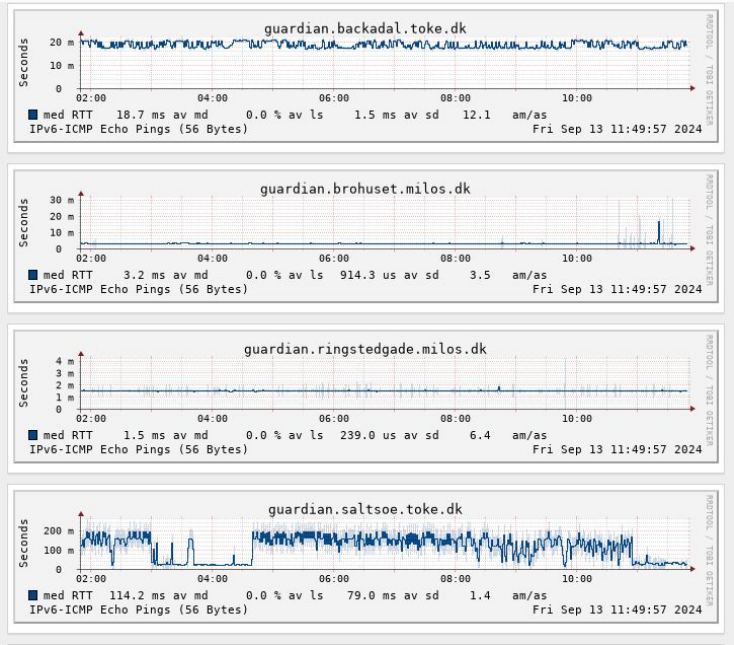
Any ISP can get a consistent low latency like this… for all their subscribers…
Without upgrading any hardware!

This is not an ad. The ISP you save might be your own.



Debloating 5G is so far, nearly impossible

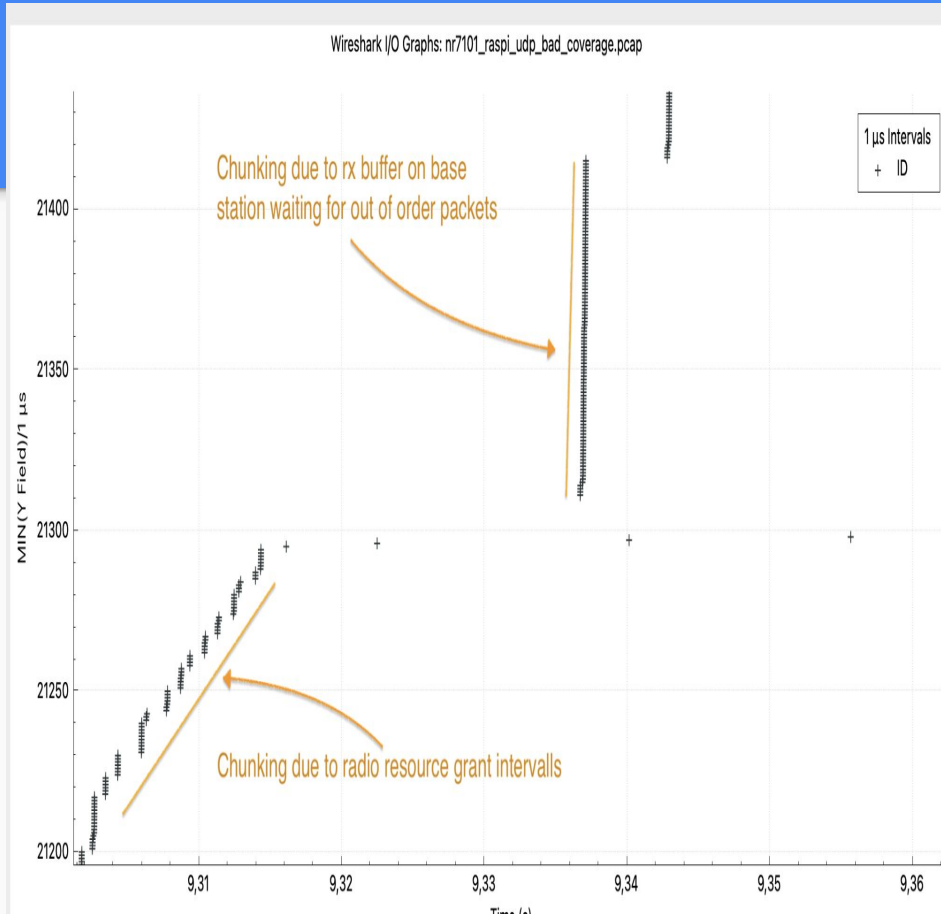




Try Cake-Autorate until that frabjous day some 5G provider pays attention!

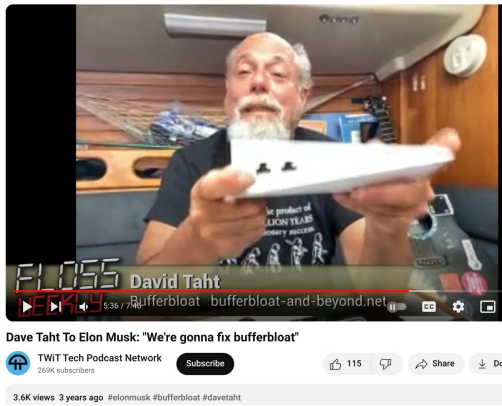
More 5G madness

On the base station receiver side (uplink from customer), there is a re-ordering buffer. Meaning if it receives a out of order packet, it will hold back the packets h has traversed the radio link. This delay set by default to 200ms, tunable up to at least 1000ms. So, in a multi link environment with one leg at 40Mbit and the other at 1Mbit (quite common since 5 has high frequency and therefor poorer reach than 4G) 5g puts in an extra 200ms of latency.



Source:
https://api.starlink.com/public-files/StarlinkLatency.pdf

Starlink is looking GOOD!



Starlink engineering teams have been focused on improving the performance of our network with the goal of delivering a service with stable 20 millisecond (ms) median latency and minimal packet loss.

Over the past month, we have meaningfully reduced median and worst-case latency for users around th world. In the United States alone, we reduced median latency by more than 30%, from 48.5ms to 33ms during hours of peak usage. Worst-case peak hour latency (p99) has dropped by over 60%, from over 150ms to less than 65ms. Outside of the United States, we have also reduced median latency by up to 25% and worst-case latencies by up to 35%.

https://www.starlink.com/map?view=latency

If only more ISPs adopted similar standards!


OpenWrt
In Space!!

New Congestion Control Research worth exploring

(still a pretty active topic)

dAQM - claims better than cobalt performance across the board

SEARCH - improvement to slow start

ETC CC - uses microbursts to detect bandwidth pre-congestion

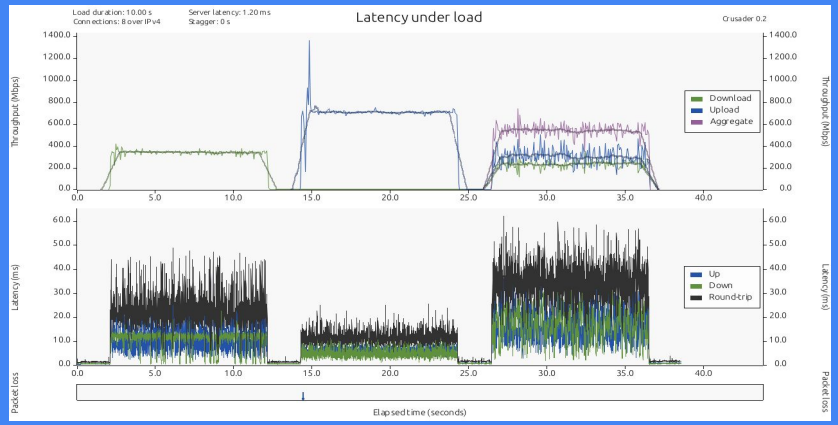
Codel for P4 - Codel AQM on tofino switches

152 bufferbloat pubs in 2024 (so far) on google scholar…

The Make-WiFi-fast-2 project

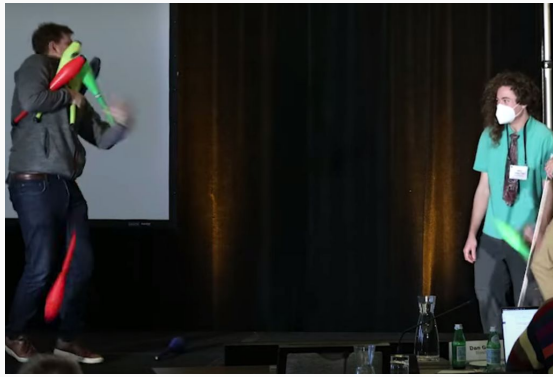
FQ_Codel now working great across many WiFi chips in OpenWrt. Notably the Mediatek MT79 is now best in class. Still, most vendors not shipping it…

Anyone going to IEEE 802.11 meetings? 802.11bn (WiFi 8) is nearly standardized…



Return(-ENOFUNDER);

New bufferbloat.net project: CAKE-MAINT



1) Fix a lot of (minor) bugs in fq_codel and CAKE
2) Re-measure, re-tune, evaluate new approaches
3) Make CAKE's shaper multicore!

"This is a KEY open source project on the internet and needs sufficient funding to continue maintenance of the code but also to incorporate improvements."

- David Reed, Author of the End to End argument, Inventor, UDP

Currently sponsored by the NLNET NGI0 Fund, and Comcast Research, with datacenter support from Equinix. Can anyone else help?

Thanks!

Contact us:

dave.taht@gmail.com
www.libreqos.io

lists.bufferbloat.net