Bufferbloat and Beyond

Speeding up the edge of the internet
Making for better voip and videoconferencing

Dave Taht, CEO, TekLibe, LLC
Dave.taht@gmail.com

Intro Shapiro

- Dave Taht's Bufferbloat Project project members have been working on reducing network latency, and improving the internet, with new algorithms, open source code, and IETF standards, for over 12 years.
- Today he's going to try and explain a few of the their algorithms many of you already have, in your devices, running on each and every packet, that have made it more possible for quality videoconferencing, among other things.
- Without further ado, Doctor Dave Taht:

Thx

In my coming talk we are going to try and tackle some really difficult networking topics like cryptography, the importance of Round Trips (otherwise known as RTT) – congestion control - how web traffic and videoconferencing traffic have trouble co-existing due - “bufferbloat” - how many of you have hear of that? “network qeueing delay” - which if we can somehow fix in the coming years, will lead to a better internet for everyone.

Hands

Could I get a show of hands? Anyone here ever heard of “Fair Queuing”? “Active Queue Management?”

How about more basic stuff? WEBRTC?

Those of you not paying attention, please raise your hands?

Stage call

- I’d like to bring some brave volunteers up to the stage
- [arrive, get stuff setup]

I’m going to start with the simplest most basic thing all of you are familiar with. First we’re going to describe how voice traffic used to work, and how it works after it got packetized.

And then... We’re going to simulute loading a simple, basic, 1993-era web page and then hopefully get to explaining some of the big concepts I just mentioned.

Connect M-S-C-D

Up until the 1960s, when making a telephone call, the analog signal was transmitted over wires via a “circuit switched” network, that were plugged in by various operators. And then your sound impulses were translated into a “waveform”, and carried across that kind of “network”. Operator, Texas institute of Technology and Science calling MIT”

“Dallas Operator Calling Boston” “Boston Operator Calling MIT” [plug in wires]

We Wave across the room

Gradually we automated this task, but… it was a lot of wires that had to get plugged into each other to make a cross country call.

After you got the call setup… [toss wires and get in line] you could talk to someone clear across the country… [start transit wave]... the sound of your voice was translated into an analog waveform.

S: Isn't C: That D: Amazing!!

Packetization

And we learned that basically, every 20ms, we could send a tiny fragment of a voice, over what was then called “The arpanet”, and reassemble them at the end.

[Traverse…] D: Isn’t S: That C: Amazing!

Packet Loss

And we also learned that packet loss – for any reason, on such large chunks of data as large as a verbal word

The three start to traverse the stage again [Grab the S out of line] -

D: Isn’t C: Amazing!

Loss recovery

… Packet loss was a new and real problem… and we have now spent many decades trying to compensate for packet loss, and jitter, and delay, and for that matter, reordering – one packet from the stream might take a detour around the moon like

S: That!

Bandwidth

- A much better way of thinking about “Bandwidth” is not “megabits per second”, but steady kilobits per millisecond.
- That’s 1000 times finer granularity than how humans describe it, but when you slice data up into chunks that small you can compensate for lost bits of it.
- So instead of transmitting whole words, like “Isn’t that amazing!” we transmit audio bits at really fine intervals usually spaced 20 milliseconds apart, and if you lose one one you might hear a small disssssstortion of the audio but that’s it.

Videoconferencing

- [Team Hand Dave Ring]

Videoconferencing is very similar. You take a picture, then another picture, then another picture. So long as the intervals between “frames” are constant, and very very small and have minimal packet loss that can be compensated for, your brain can compensate for most of the weirdness that happens.

Bandwidth again

- I want to go back to something important – for packetized voice and videoconferencing to work well, you need:
- Consistent, smooth, nearly lossless deliverery not of megabits per second, but of kilobits per millisecond.

Web Traffic

- [Team get Clubs setup]

The requirements of web traffic are very different from this. All we care about really is that the page load completes as fast as possible - typically under 3 seconds. Movie streaming is similar also, you typically download 10 seconds of a movie, or more, as fast as you can - before starting to play it.

You **don’t** need, smooth, consistent bandwidth in kilobits per millisecond, but as much bandwidth as you can grab, quickly.

Loading a web page

Here we have the DNS server. Over here is the “SERVER” that has the content you want, and you, sir, are the client. Let’s LOAD a web page. First up, we need to translate a domain name, like www.ttivanguard.com into a number, called an IP address. So we throw a packet to a DNS server which translates that name into a number, and it sends the number back to the client. The client attempts to connect to that server’s IP address through “the cloud”.

Through a bunch of tubes, interconnected via routers (funnels).

The download

And after that, we start downloading the various objects – text, pictures, formatting in the web page. We might need to make a few more DNS queries along the way, or connect to other servers in the cloud…

And we have this nearly continuous exchange of packets, or in this case clubs, transfering more and more of the data… [me slowing down so the audience starts getting the joke], from all the sources and sinks… get to 6 clubs… until can be fully displayed.

STOP, drop clubs, dive for papers

WALLA! YOU’VE GOT YOUR WEBPAGE!

Grab papers from stage

- Rick Astley

Intro

Ladies and gentlemen, the mit juggling club!

MIT’s work on networking is unparalleled, and it’s not co-incidence that in the computer science field, jugglers can be found, everywhere.

Jaimie here is studing computer science at Hahhvard, Vasu, is studying “computer engineering” at MIT, and joshua, when not juggling professionally, moonlights as a network engineer at a major search engine provider.

W/MIT juggling club

- They are going to help explain the more difficult topics coming up
  - Cryptography
  - Internet Congestion control
  - Bufferbloat
  - Fair Queuing and Active Queue Management (AQM)

Cryptography

The web evolved. Things started to get more complicated after 1993 with the introduction of cryptography. In addition to the setup steps you just witnessed [repeat], we had to add two more “round trips” to it.

C: “nonsense” S: “nonsense”

To negotiate a secure channel for the conversation

C: “nonsense” S: “nonsense”

Crypto 2

M: “This is how cryptography really works” C: “nonsense”, toss, S “This is how cryptography really works”

M: “You have to negotiate a secure connection”: C: “nonsense”, toss, S “You have to negotiate a secure connection”

M: “Then encrypt the data for each and every packet. C: nonsense, toss, S: Then encrypt the data for each and every packet.”

M: “The importance of cryptography is that a man in the middle cannot understand what’s going on” - S: nonsense, toss, D: intercept, repeat numbers dubiously, then toss.

In unison: “Isn’t that amazing!”

Blind

But it’s even more complicated than this.

Because in the case of packets, the client and servers… are separate by an unknown distance… they might be really far apart… or closer together… and we never know at the outset what the actual, current bandwidth between them is.

The clients and servers are essentially… BLIND.

[whip out blindfolds] [D: starts to juggle 3 balls]

Round Trips

While you can measure the amount of time it takes for a reply to come back, you don’t know if that was the network, the server, the cryptotgraphy or the queuing delay that took the time. There’s actually three laws for how to manage this sort of thing – BDP, srqrt BDP times the number of flows, and one called ‘Power”. And despite 50 years of internet development we still don’t know which of these laws is correct.

So instead of using these “laws” for the purpose of analogy, we’re going to use an entirely different law. Gravity.

Height of the queue

Pay no attention to the balls themselves, ok? [Use a ruler]

Watch the hands. They are moving stuff from hand to hand at roughly the same rate, no matter if it’s 3 balls or 4, or 5.

That is the amount of needed “buffering” for this trick, but in general the same amount of data is being transferred. It’s just going higher every time. You just have more balls in the air.

The role of packet loss

- A new flow enters the link
- [Toss from one to the other] The height of the flow here represents the minimum amount of “buffering” required to ensure we defeat the law of gravity.
- [Toss from another to the one]
- So long as packet loss is used to control how much bandwidth is found and used!

Probing for bandwidth

- [Switch to pins]
- I gotta introduce a dose of reality into this. What actually happens on a web download, once it gets going, is first two GIANT data packets are sent over TCP, and then a really really tiny packet is sent back, called an acknowledgement. That tiny packet contains a list of what was sucessfully received so far.
- [Toss two pins]

QuackKnowledgements

Dave: What’s that?
CLIENT: It’s a quacknowlegement.
:audience boos:
Dave takes the duckie and walks back to SERVER
That… quackknowlegment says YES I received the data, please send more. And something kind of non-intuitive happens here, we double the number of packets we sent to probe for more bandwidth, and the distance. Sending 4 packets! [send 4]

Collapse

- And we get 2 more “Quackknowlegments” back…
- And then we send 8 more packets!
- [collapse]
- Most traffic is governed by the number of round trips required for the percieved bandwidth. And what a packet loss indicates in this instance is two things -

The quackknowlegement, in this case says two things – please retransmit the lost packets, and **slow down** so the reciever can handle the data you are sending.

Packet loss is actually essential

This idea is fundamental to the structure of the internet. We probe for more bandwith by throwing ever more packets, and when one is lost, the sender is signaled to slow down, and to fill in the missing data.

[juggle for a while]

Juggling

If you were to imagine the internet consisting of trillions of jugglers, all obeying these two simple rules – again, lose a packet, slow down, retransmit, slowly increase the speed, lose a packet, slow down, retransmit… sending these not through the air but threw a series of tubes, and funnels designed by a madman, you wouldn’t be too far off.

Bufferbloat

And the crux of the problem I’ve beent trying to explain for the last half hour… is how hard and yet how important that is, in the face of web/movie/file transfer traffic trying to coexist.

Too few packets in flight means your web/movie/file transfer traffic can take too long, and too many packets in flight, is bad for voice and videoconferencing traffic. Packet loss is needed to keep the queues short, but excessive loss, undesirable for other traffic.

Up until 2010 or so, we were making web traffic “better” by really, really overbuffering too many packets in flight, and destroying the capability to do interactive traffic simultaneously.

Bandwidth changes

And the network is constantly changing up and down, not just with flows entering, probing and leaving the link, but with wifi and wireless where if you move a fraction of an inch, the available bandwidth can change by a lot.

I entered this mad scene in 2008, when applications that I relied upon, like skype, were acting up periodically, and I didn’t know why. It wasn’t until jim gettys identified the root cause – really execessive amounts of buffering – that I understood.

Overbuffering

The amount of buffering we observe on the internet today is very often still excess of what actually needed, sometimes measured in seconds – too much even for web traffic!

We started developing, and deploying new algorithms to manage buffering back then, and many are widely deployed, but there’s still billions of devices left to upgrade.

Age

It’s been a very long 10 years.

I’m really happy that we developed better end to end congestion control algorithms such as BBR and packet pacing, and we’ve also made great strides in increasing bandwidth, and in improving the routers to be more intelligent as to how to manage the flows of different kinds of packets.

But we’re not done yet. The solutions are unevenly distributed. But maybe henceforth when you see someone video distort, or freeze you might be able to explain a little about why, and how to fix it.

FQ and AQM

In the time remaining, I’d like to try and describe how two of the algorithms we’ve developed actually work to assist the network – enforced at the router – work together to make it possible for both videoconferencing and file transfer traffic.

Close

- Setup rings, get on unicycle – jaime here is emulating a smarter home router
- On the bottleneck router...
- Fair queuing makes the low rate voice and videoconferencing flows interleave well with the fatter web traffic.
- Active queue management, attempts to identify the big fat flows, probing for bandwith, by either forcing a dropped packet
- [switch color ring]
- Or, in this case by marking a packet with an explicit congestion notification (ECN) early enough to keep the probing occilations under control.

Thanks Everyone!

- Huge thanks to our MIT jugglers -
  - Vasu!
  - Joshua!
  - Jaimie!
- For coming out today!
- And a big thanks to Len kleinrock for all the queue theory I’ve mangled today.

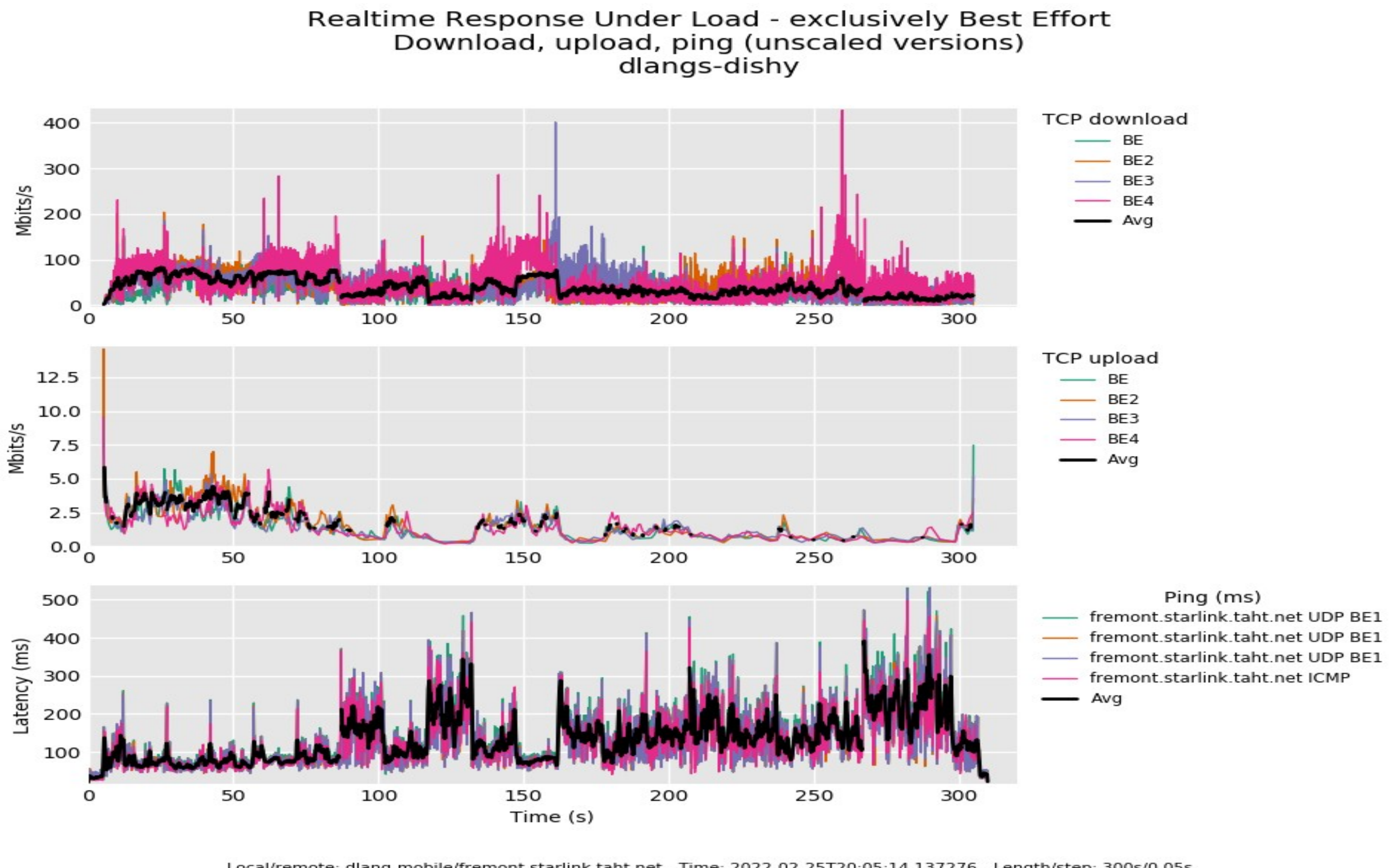
My current problems

- Trying to get NTIA/FCC to mandate new and old ISP services get better FQ+AQM algorithms enabled by default
- NTIA filing on “bufferbloat”
- Also trying to get Network Neutrality people to intuitively understand congestion control and these algorithms
- Trying to get more vendors to ship newer code with the algorithms available.
- Getting ISPs to enable them for customers automatically
- Trying to raise awareness of how the internet really works, for everyone

Bufferbloat Still exists on everything

- DSL is “slow” not just due to low bandwidth, but seconds of buffering
- Cablemodems now have DOCSIS 3.1 with the PIE AQM, but only comcast has deployed it
- We can fix these with just an “upgrade in place” on many existing routers today
- 5G folk brag about lowered latency but also have seconds of buffering
- Wifi, although fixed for 5 chipsets with default APIs in the linux kernel… in 2016
- And even new services are getting buffering wrong for a mix of videoconferencing and file transfer traffic.

Starlink 2/25/22



Bufferbloat.net Resources

- https://bufferbloat-and-beyond.net
- IETF AQM working groupe
- The “Bloat” mailing list
- RFC8290
- OpenWrt “Smart Queue Management”
- “Optimize for videoconferencing and gaming” - eero and google
- https://www.patreon.com/dtaht