COMCAST

First Steps in Bufferbloat Mitigation

Carl Klatsky, Drew Taylor, Yana Kane-Esrig

October 2016

What is Bufferbloat?

- *"Bufferbloat is the undesirable latency that comes from a router or other network equipment buffering too much data"*
  -From https://www.bufferbloat.net/projects/

- Large buffers are the result of falling memory prices

- Large buffers provide the highest throughput value in both benchmark testing & real world usage

Why care about Bufferbloat?

- Throughput has been the primary metric pushed by the ISPs in advertising & competitive benchmarking

- But this comes at the cost of additional latency as packets are idle, awaiting transmission out of the buffer

- The additional latency impacts real-time applications such as voice & gaming

- The Bufferbloat phenomena can occur in either the upstream, downstream, or both directions

Why care about Bufferbloat?

- Back in the early days of residential Internet usage, the typical usage was one device / one non-real time, application exclusively using the Internet connection

- Maximizing the throughput available to that one device equated to a good user experience

- The proliferation of Internet enabled devices generating a mix a real time & non-real time traffic disrupts the prior usage model

- The real time applications experience latency leading to a bad user experience

What has been done about Bufferbloat?

- Active Queue Management (AQM) software algorithms operate by dynamically dropping packets from the buffer, trying to minimize latency while maximizing throughput
- AQM algorithms include CoDel, FQ_CoDel, PIE, others.
- One AQM algorithm (PIE) is now part of the DOCSIS 3.1 standard

What did we do about Bufferbloat?

- Field trial of implementing static buffer sizes on DOCSIS cable modems across our network

- AQM testing to date by Internet researchers conducted on consumer home routers, not cable modems

- CableLabs work on buffer control & PIE using cable modems was only conducted in the lab

What's DOCSIS?

- Stands for Data Over Cable System Interface Specification
- Set of specifications defined by the cable industry covering layer 2 packet encapsulation & transmission over the layer 1 physical medium (copper coaxial cable)
- Increasing versions (2.0, 3.0, 3.1) introduce higher transmission rates
- Although not dependent upon DOCSIS technology, the standards update to DOCSIS 3.1 seemed like a good point to include buffer management techniques

How did we do our field trial?

- Support was available to adjust the cable modem buffer size to fixed values
- Only the upstream buffer could be adjusted to 96 KB (default), 48 KB and 8 KB
- We enlisted participants to host modems in their homes with custom bootfiles and a Linux-based probe
- A test suite was run on these probes and the results were reported back to us

But Wait!

- ***I thought you said AQMs were developed and PIE was added to DOCSIS 3.1?***
- Yes, however DOCSIS 3.1 modems are not available yet, and AQMs have not (yet) been implemented on DOCSIS 3.0 modems

How did we do our field trial?

- Test suite consisted of Flent, a Netperf wrapper which runs a “canned” throughput test from the RRUL test suite*

- Canned test checks latency while a unidirectional throughput test is run, with latency & throughput checks generated by the test suite

- Downstream latency under load and Upstream latency under load are run as separate tests

*https://github.com/tohojo/flent

*https://tohojo.github.io/flent.1.html

How did we do our field trial?

- Tests are run three times a day (08:00, 12:00, 17:00 UTC) across all probes
- Test conducted over three week period, changing the buffer size week over week
- All tests are run to a centrally located server in West Chester, PA, regardless of probe's geographic location
- Approximately 50 trial participants (some dropped after trial started)

What metrics did we look at?

- **Throughput**, the mean of the data stream's average

- **Latency**, the mean of the UDP Ping RTT

- For a given metric, for each individual observation we computed a "**percent delta**" between that observation and the corresponding overall probe-specific mean for the given metric (the mean over all the observations from that probe over 3 weeks)

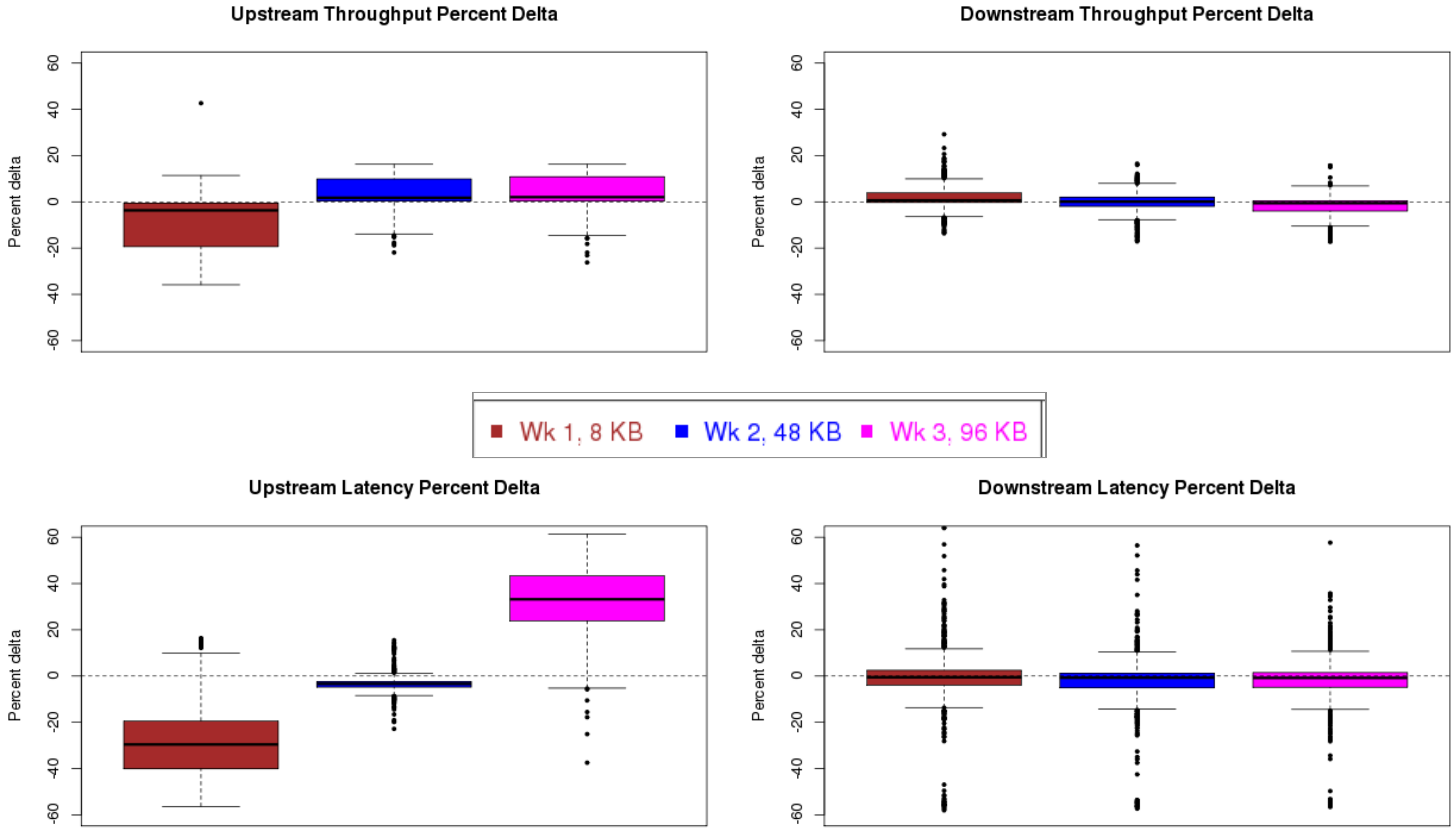
How did we analyze the data?

- Verified that the 3 weeks of this experiment were similar to others in terms of throughput & latency using 2 datasets external to this experiment

- Model fitting: for each “target variable”, fitted linear regression to determine which variables were “significant predictors” and which were not significant predictors
  - Target variables: Percent Deltas for each of the 4 metrics
  - Full model predictors: Week, Day of Week, Time of Day

If there is **no systematic difference** among the 3 weeks in the values of the given metric, **then the week should NOT be a significant predictor** for the “percent delta”.

What did the distributions of “percent deltas” look like?



What were the results?

- The week (buffer size) was a significant predictor for the following metrics
  - Upload Throughput (Mbps): Week 2 14% higher than week 1. Week 3 14% higher than week 1
  - Download Throughput (Mbps): Week 2 2% lower than week 1. Week 3 3% lower than week 1
  - Upload Latency (ms): Week 2 27% higher than week 1. Week 3 63% higher than week 1

What were the results?

- None of the predictors were significant for download latency
- Day of week and time of day were not significant predictors for any of the regressions

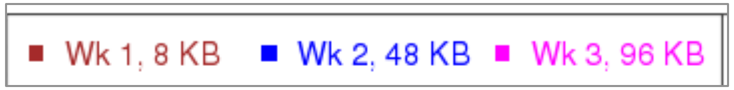
What do the results suggest?

- Upload **latency** showed the expected pattern: the week with the higher buffer size corresponded to higher average percent delta latency
  - Week 2 (buffer size 48 KB) 27% higher than week 1 (buffer size 8 KB). Week 3 (buffer size 96 KB) 63% higher than week 1 (buffer size 8 KB).
  - Lower variability was seen in week 2 than week 1 and week 3

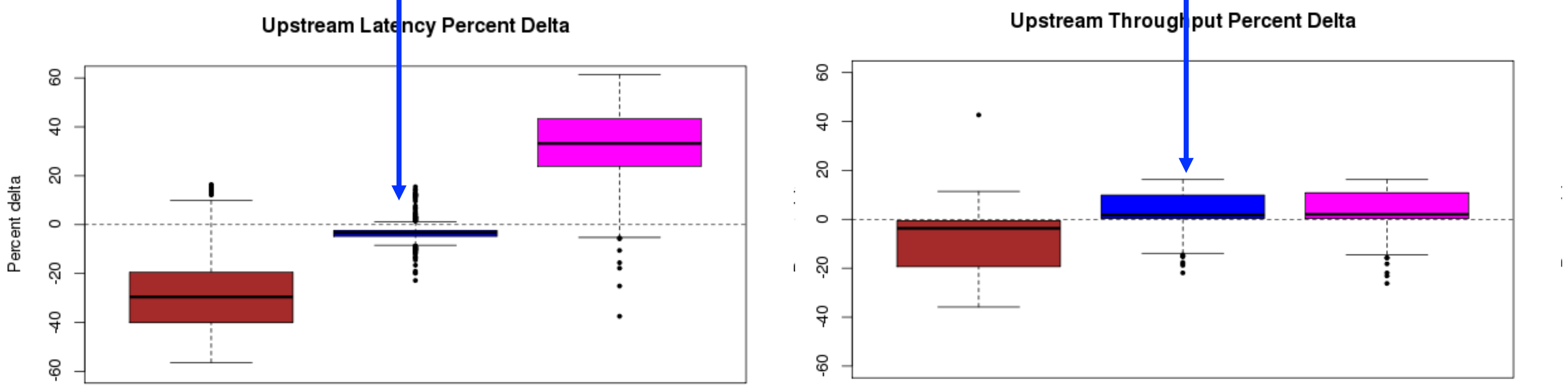
- Upload **Throughput**: the week with the lowest buffer size also had lowest average percent delta Upload Mbps, but the other two weeks were similar to each other
  - Week 2 and week 3 each 14% higher than week 1.

What do the results suggest?

- There may be a tradeoff between upload latency and upload throughput, and that tradeoff is not necessarily linear: there may be a "sweet spot" where latency is noticeably reduced, while the impact on throughput is negligible



In our test, at 48 KB, as compared to the default 96 KB, latency noticeably reduced, while the impact on throughput is negligible



What happens next?

- Cautious optimism that AQM based bufferbloat mitigation can be successful as well
- Fixed buffer size setting impractical for scaled usage
- Working with DOCSIS 3.1 modem and CMTS vendors to implement PIE AQM
- Also working internally to retrofit DOCSIS 3.0 modems with FQ_CoDel AQM