Video at the Edge

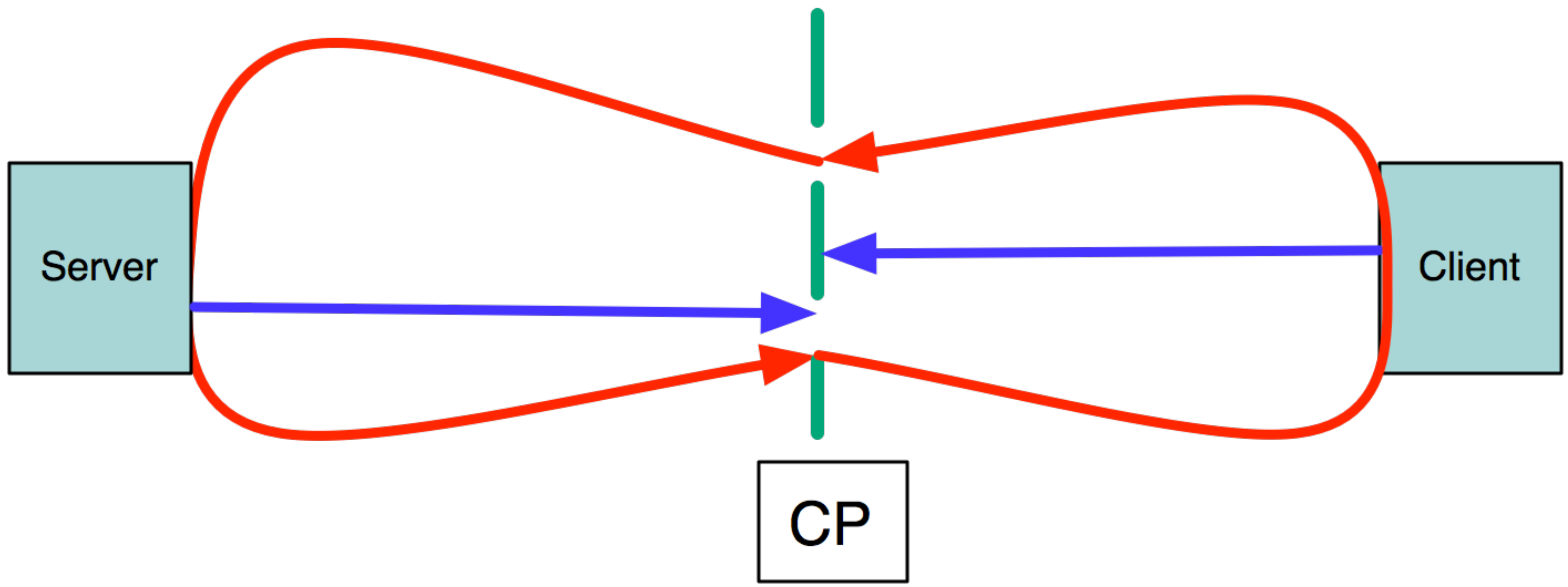
passive delay measurements

Kathleen Nichols
Pollere, Inc
nichols@pollere.net
November 17, 2016

Talk Roadmap

- Netflix and YouTube network characterization
  - delay profiles
  - delay localization
- Passive measurement rocks!
  - a wealth of information available in packet headers that can be post-processed
  - also possible to extract information from packet headers in ***real time***
- Visualization of information as it streams

Diagram of Measurement Setup



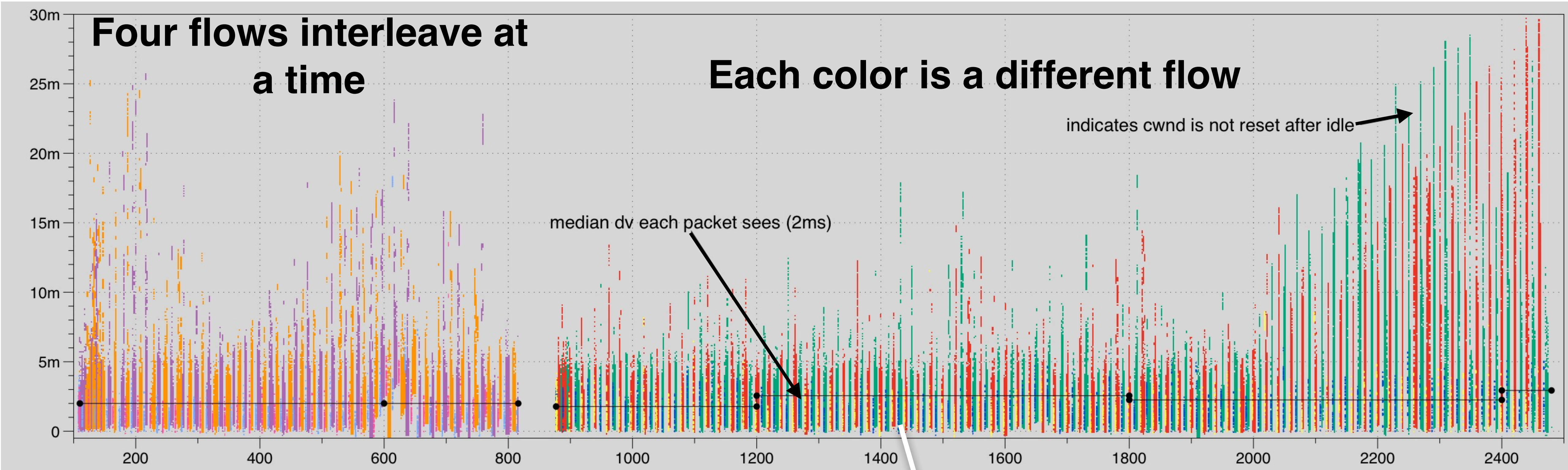
Delay is relative to the packet capture point (CP).

- red lines are round trip delay (matching packets from reverse flows)
- blue lines are *delay variation* (relative to the minimum seen)

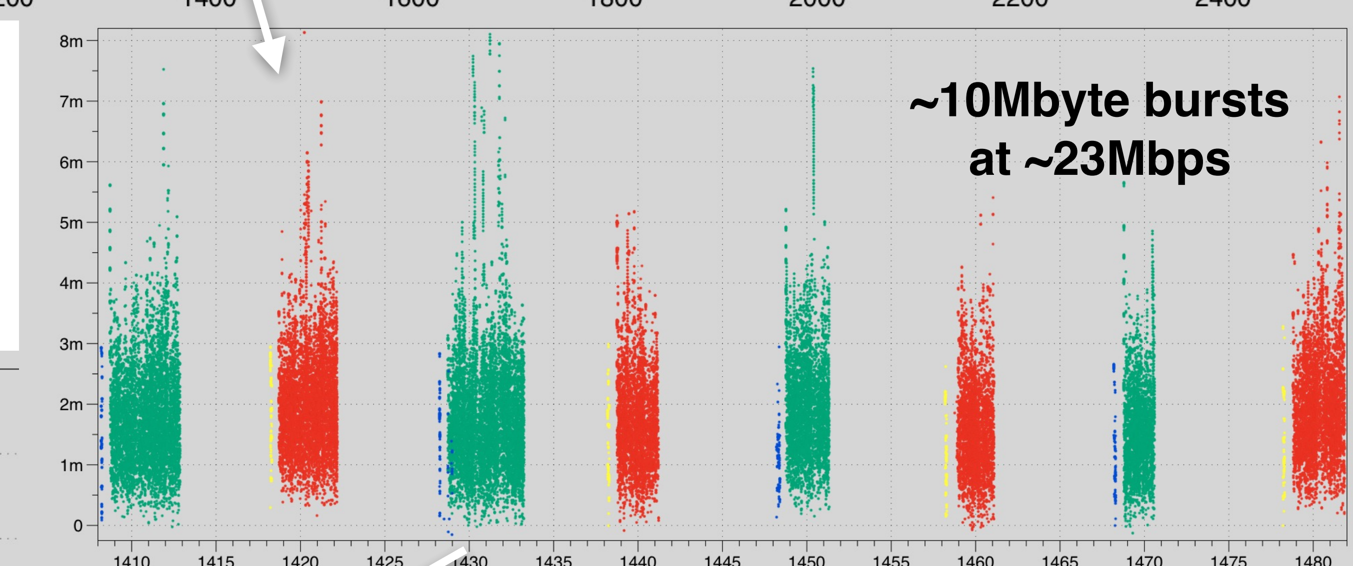
Moving CP gives different information

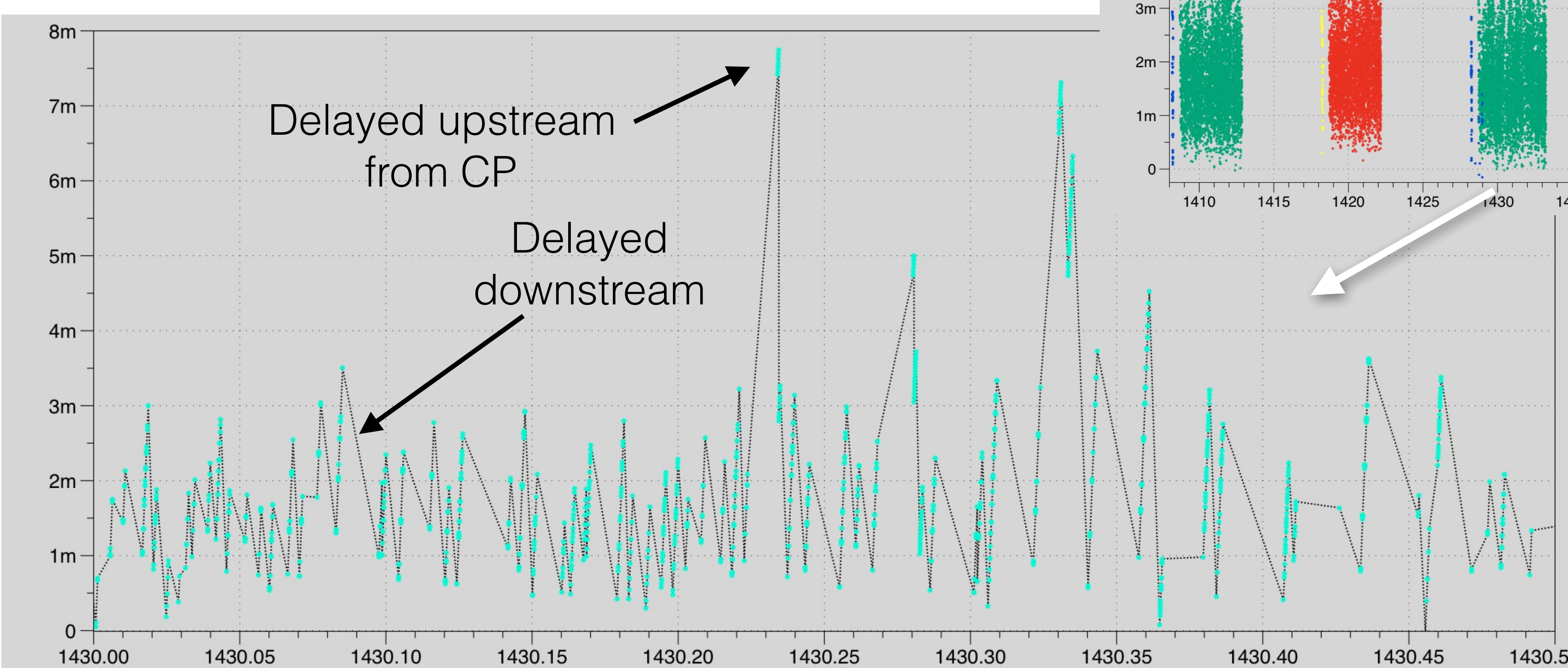
- at the edge, usually both flow directions available
- in the Internet, might only see one direction
- most of the experiments have CP next to modem

Netflix Video Delay Variation: Server to CP



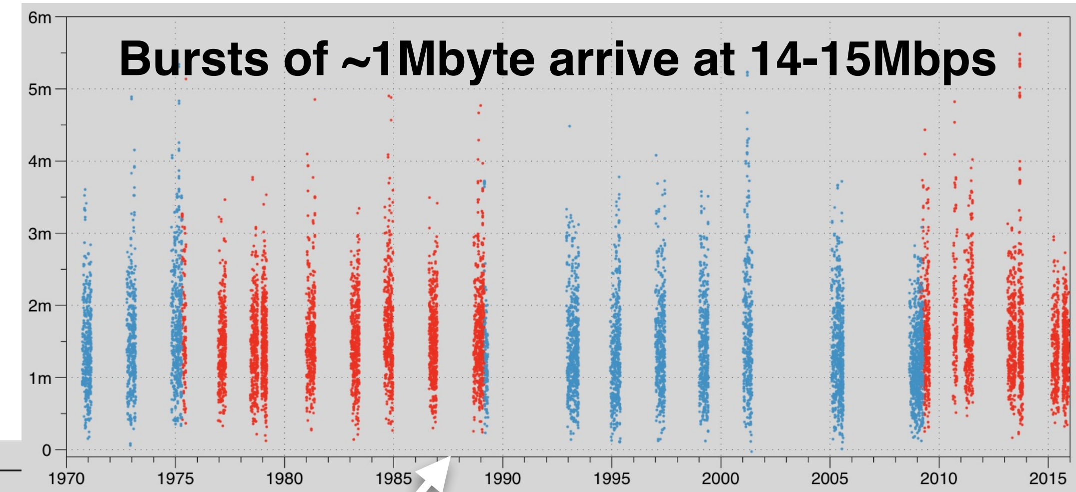
The video is in multiple interleaved flows

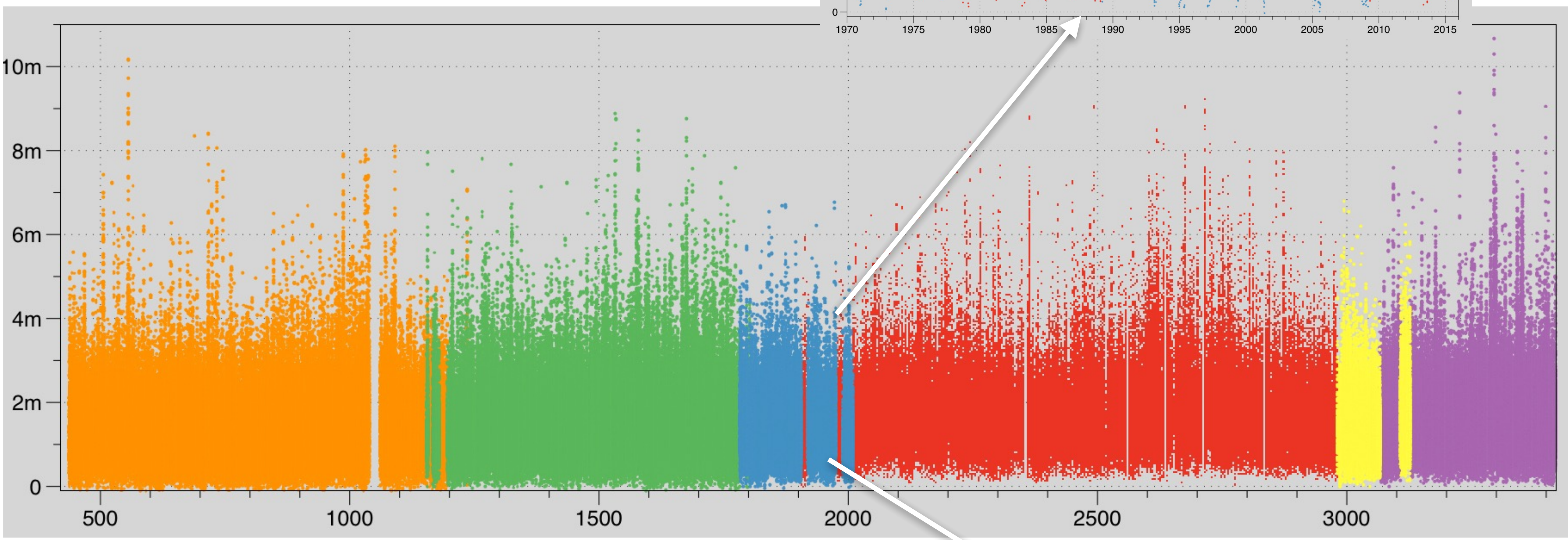


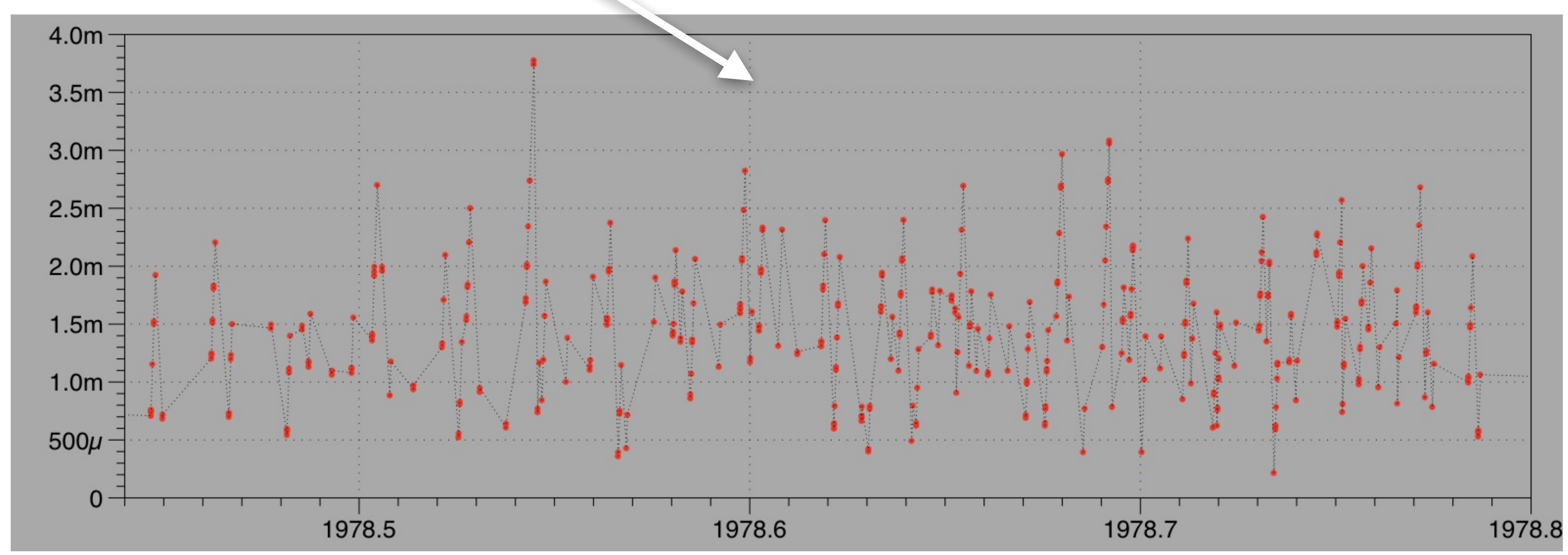


AppleTV client
09.11.16
180Mbps ISP link,
CP at modem

Netflix video, Chromecast client, 10.03.16,
Apple wifi,
CP at modem





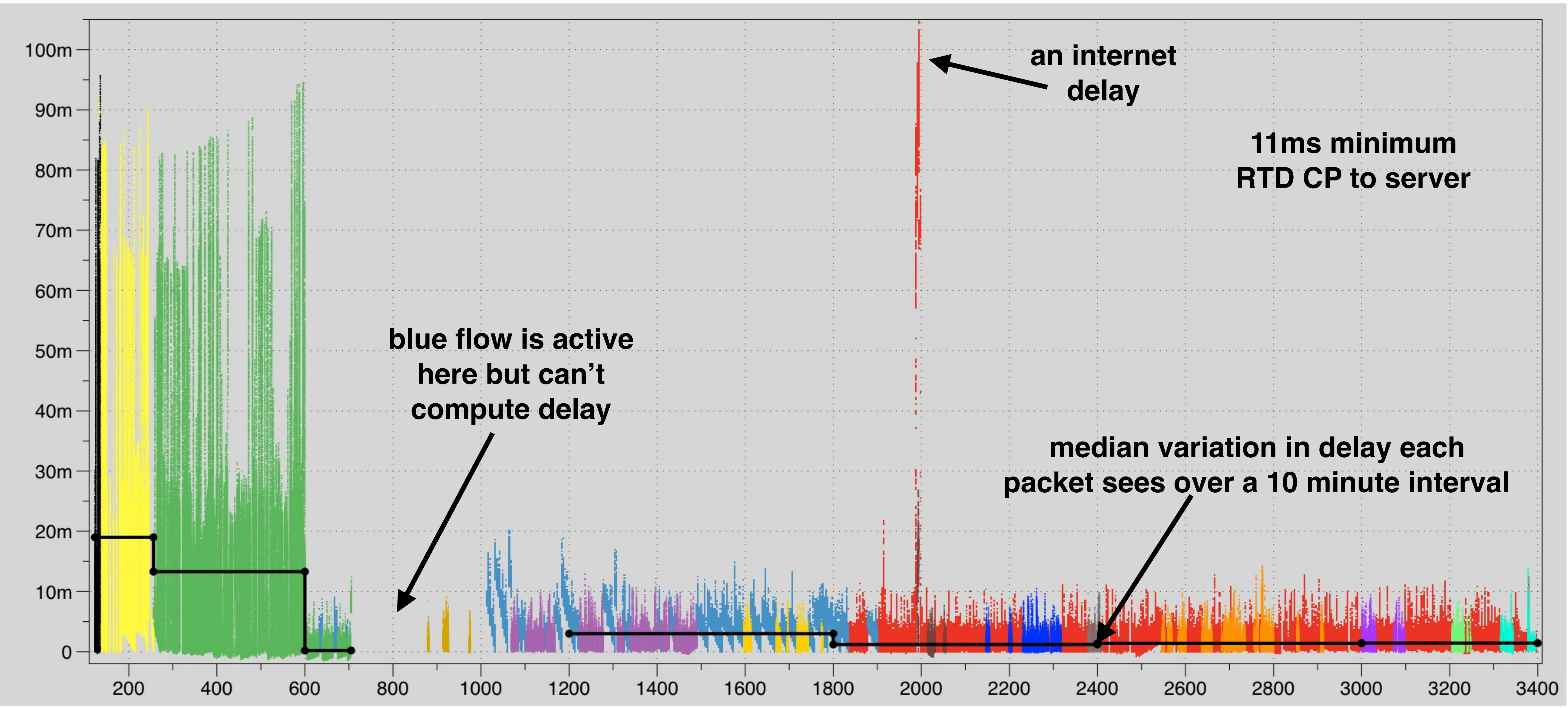


All flows from same server IP. No interleaving, multiple sequential flows

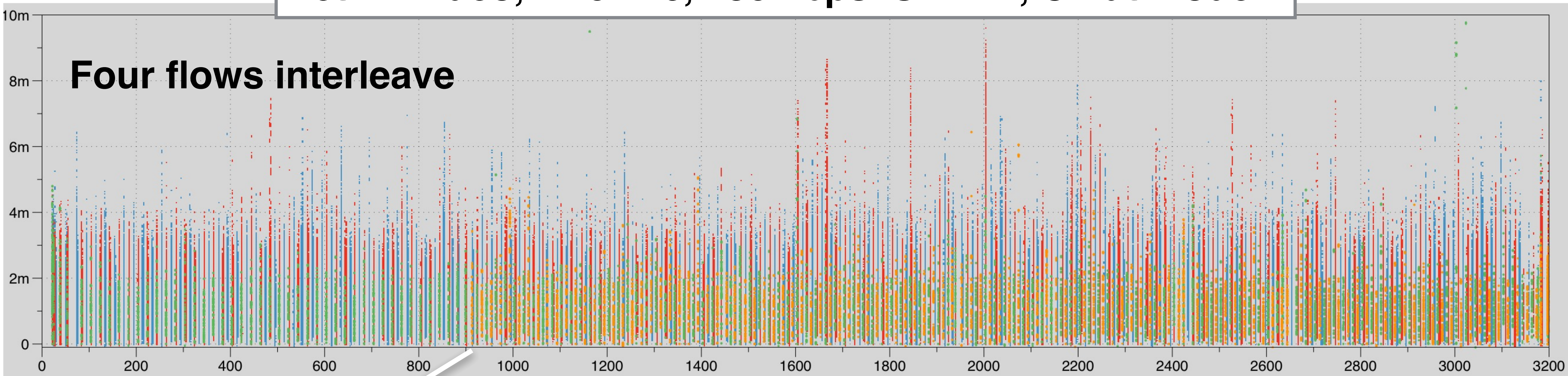
Netflix to chromecast client 10.07.16

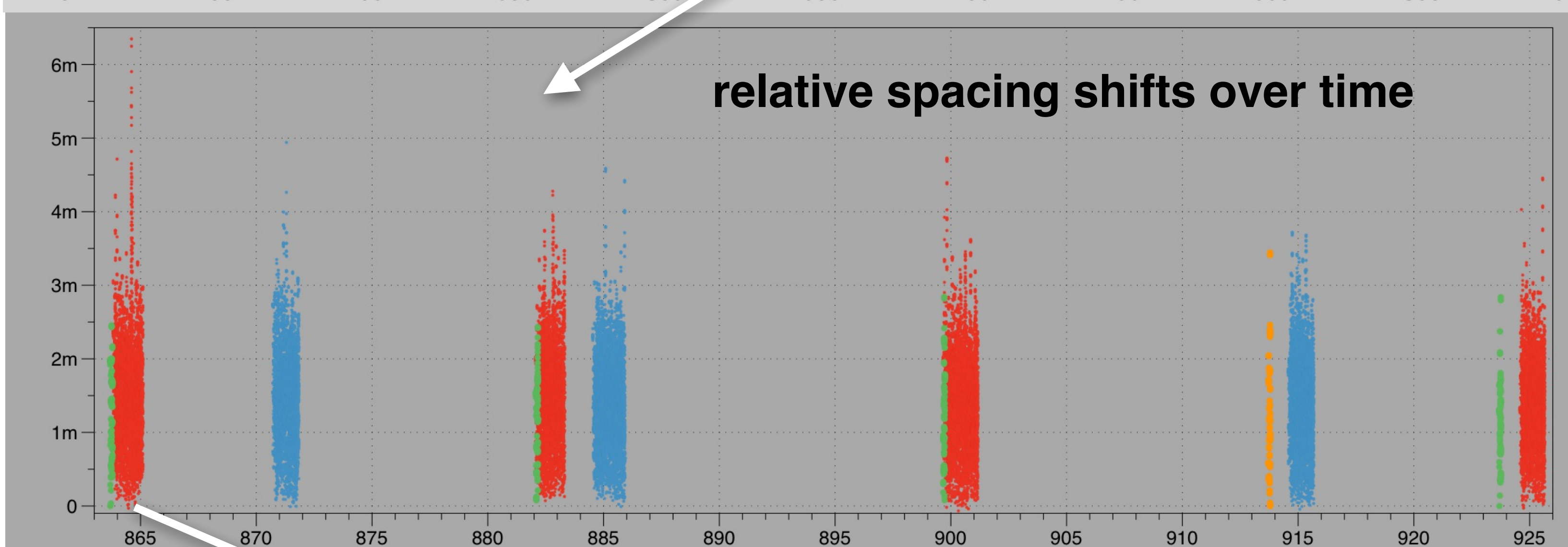
Slower cable connection (40Mbps ISP link), google wifi
CP at modem

Shows queue delay *upstream* of the CP (from server to modem)



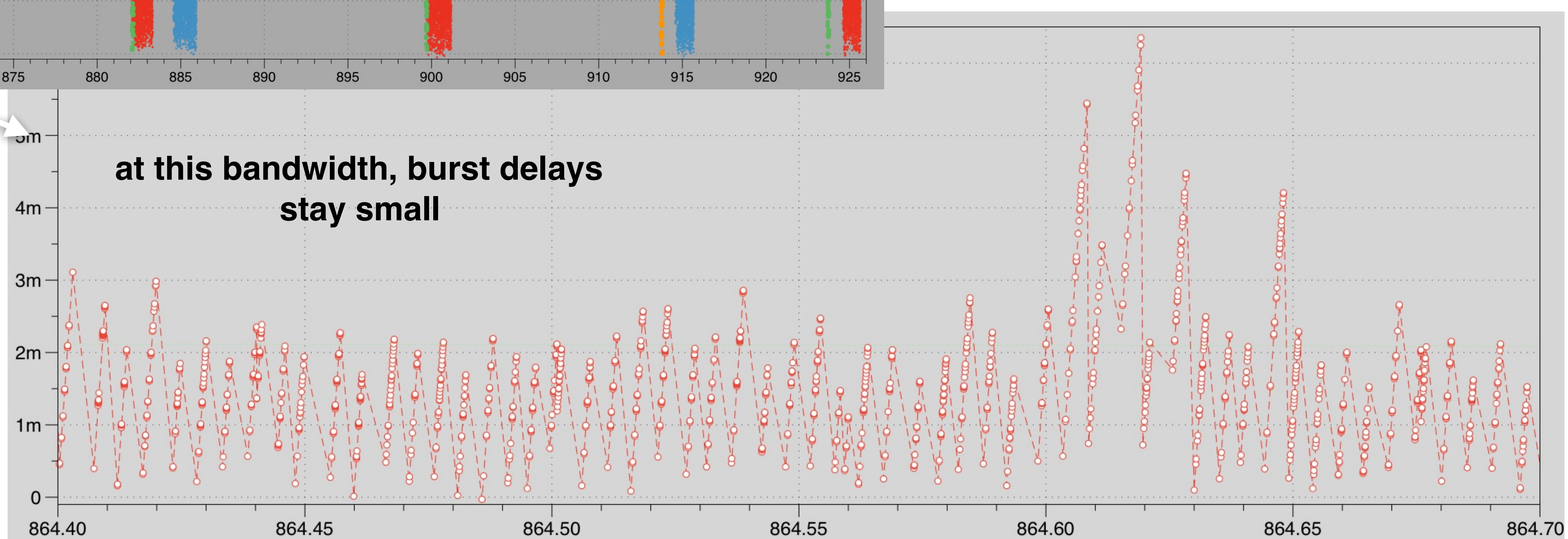
Netflix video, 11.02.16, 180Mbps ISP link, CP at modem



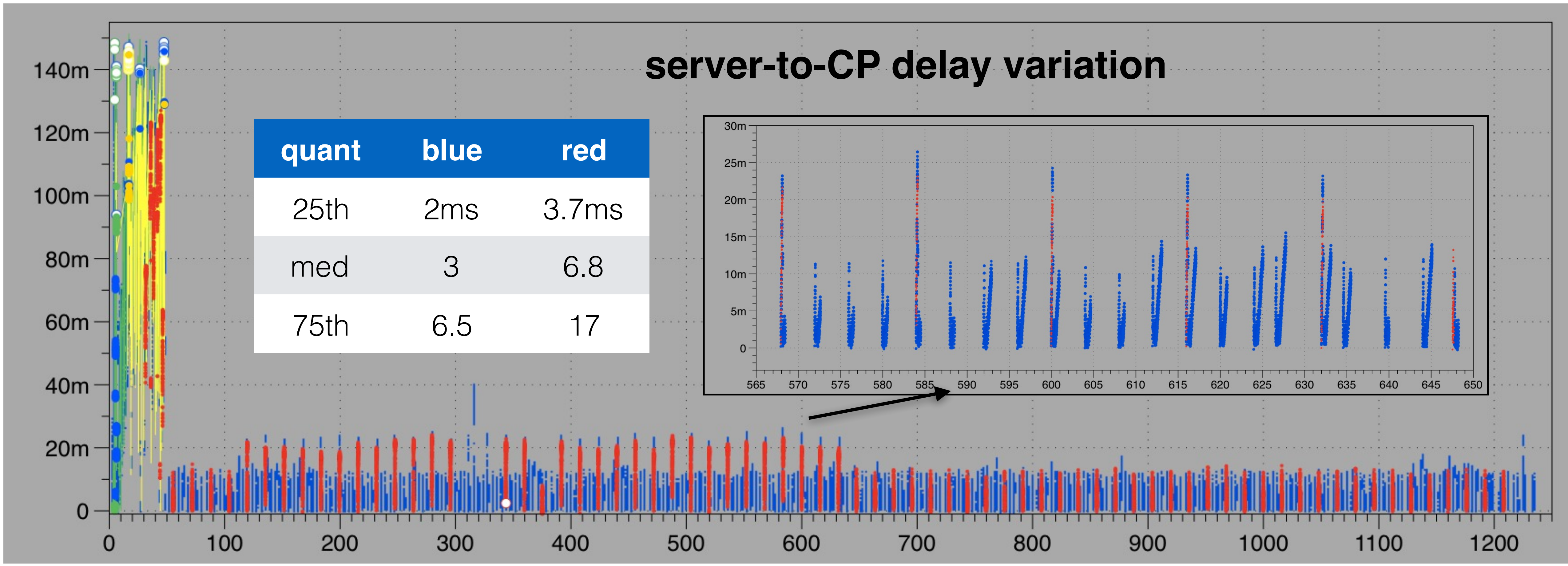


Apple Netflix app behavior clearly differs from Chromecast Netflix app

iPad client (wifi)



NF110916, HP desktop running Windows 10 in Chrome browser, CP near client all Ethernet, DSL ISP 20Mbps



quant	blue	red
25th	2ms	3.7ms
med	3	6.8
75th	6.5	17

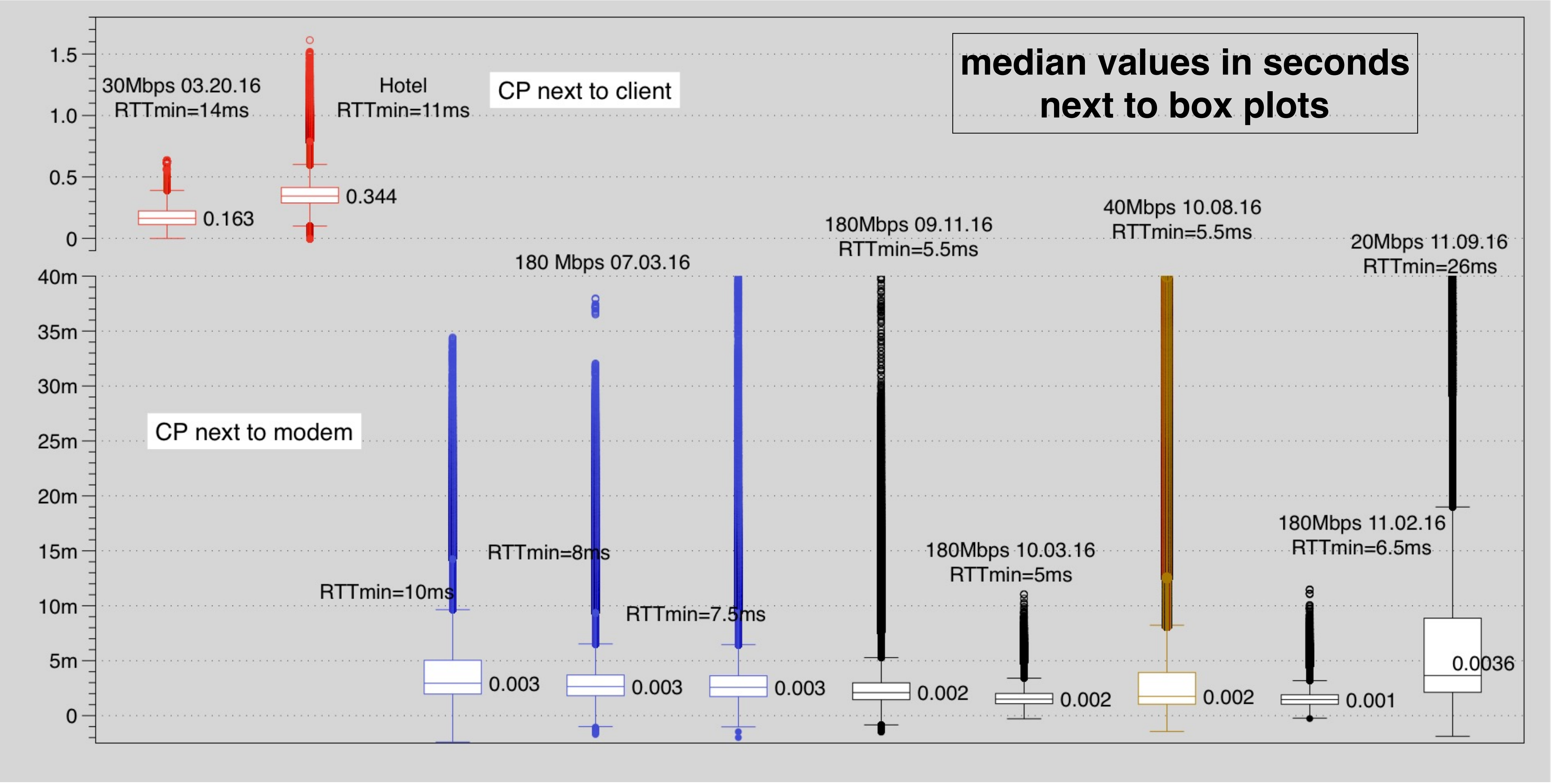
After pre-load with four flows, two flows remain

- blue one is 3.4Mbps overall mostly in 2.5MByte chunks every 4sec bursting to 18Mbps (line rate)
- red one is 96Kbps overall in 200KByte chunks every 16 sec sent in 8Mbps bursts. Often gets delayed by blue flow

Overall: 26ms minimum RTD to server, 50 microsec to client

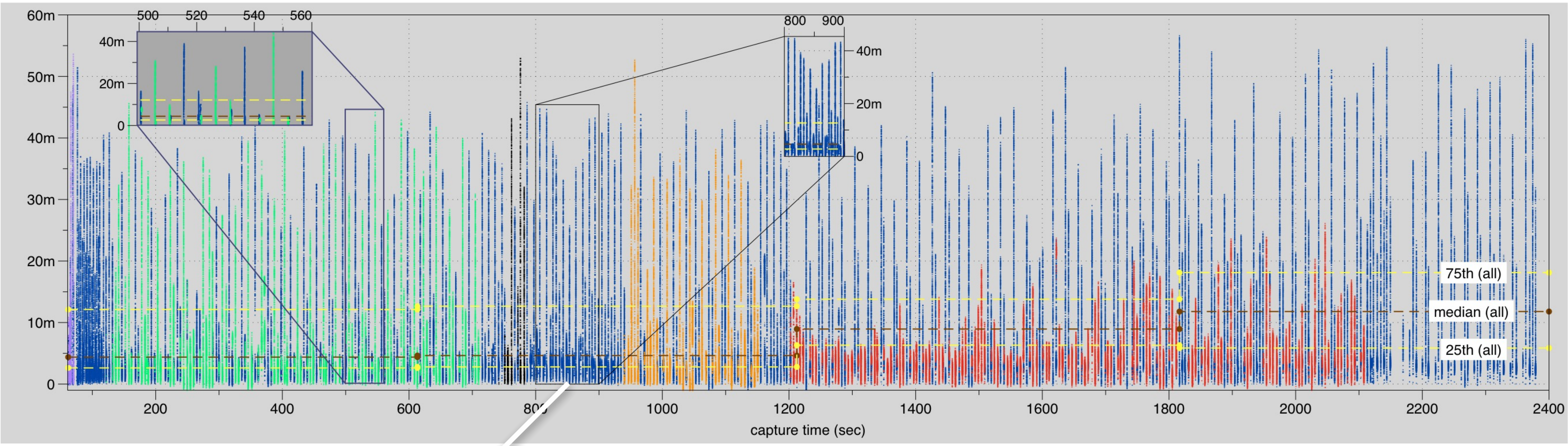
- the statistics reflect the delays the red bursts see
- client-to-server delay variation had a median of 1ms
- server to client median delay variation is 2.8ms for blue flow and 6.8ms for red

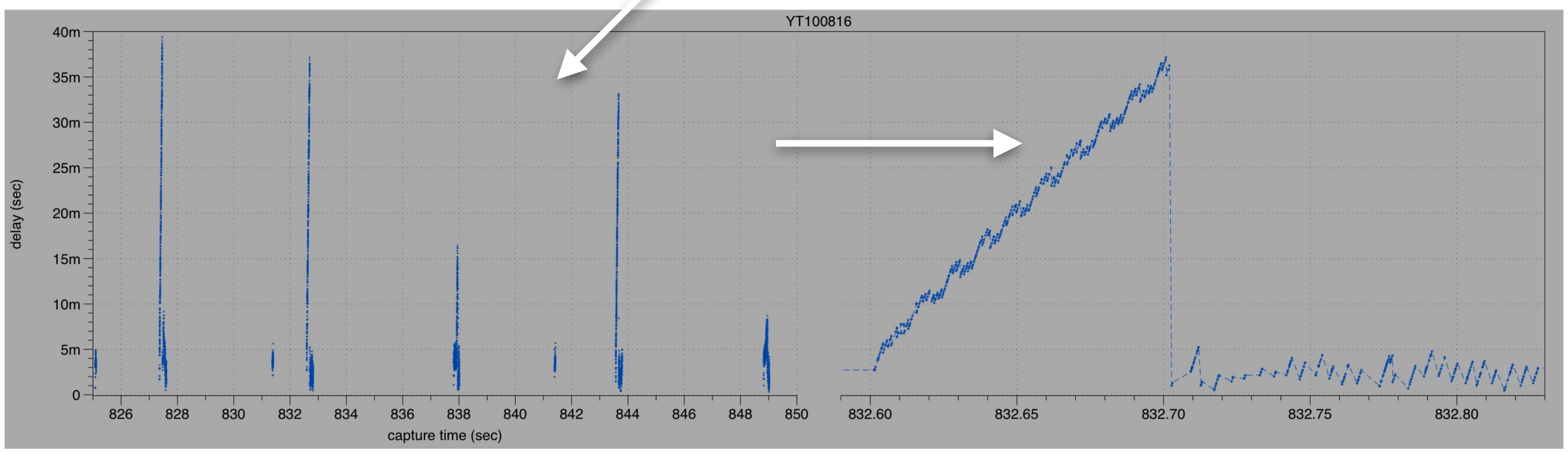
Per-packet Delay Variation of Netflix video for a range of experiments



- Serious delays when the delay from the server includes client network (likely to be oversubscription in hotel network)
- IQR wider for lower rate downlinks; bursty nature creates more delay with lower speeds, bigger bottlenecks

YouTube video: 40Mbps ISP link, chromecast client

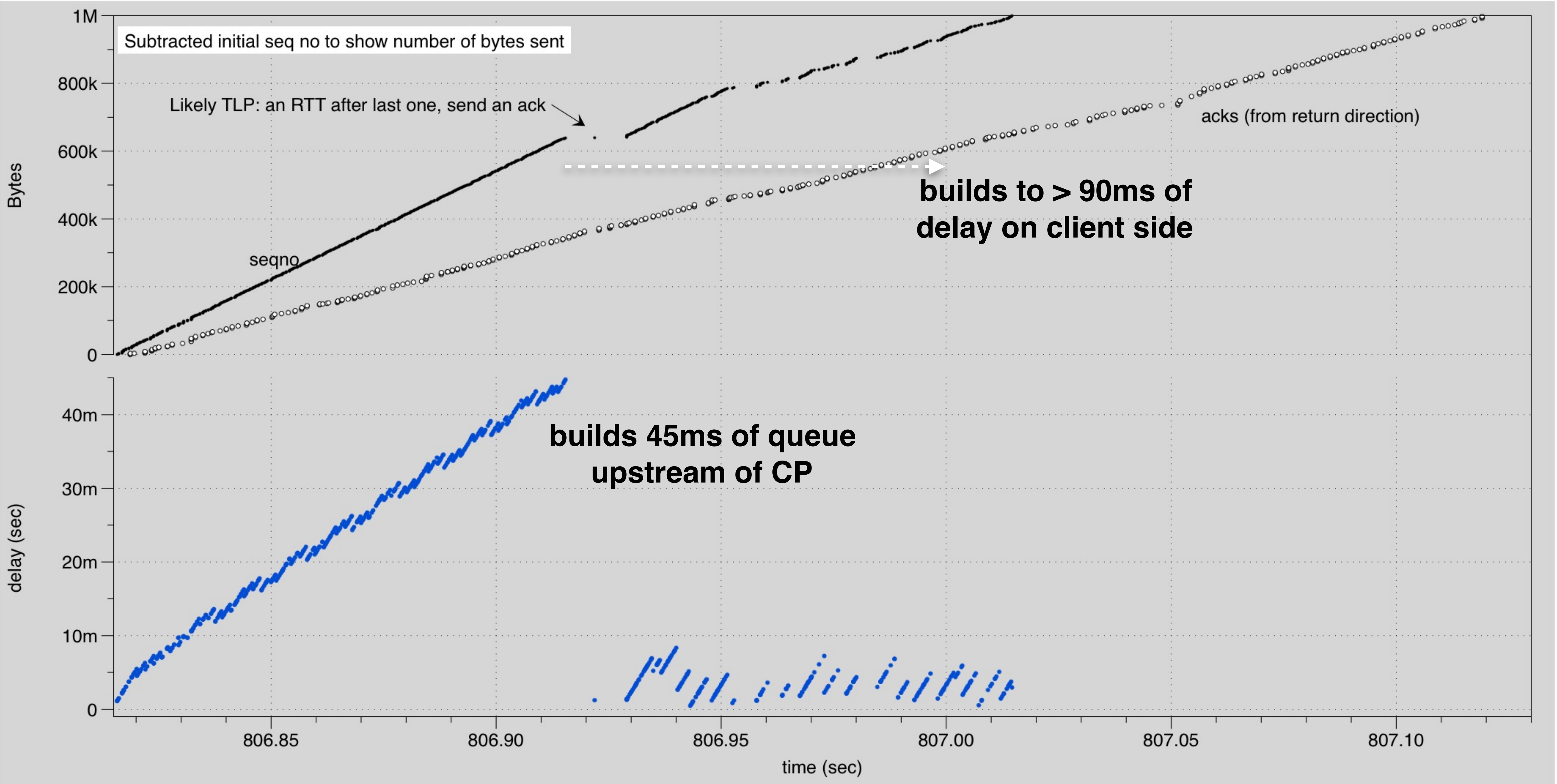




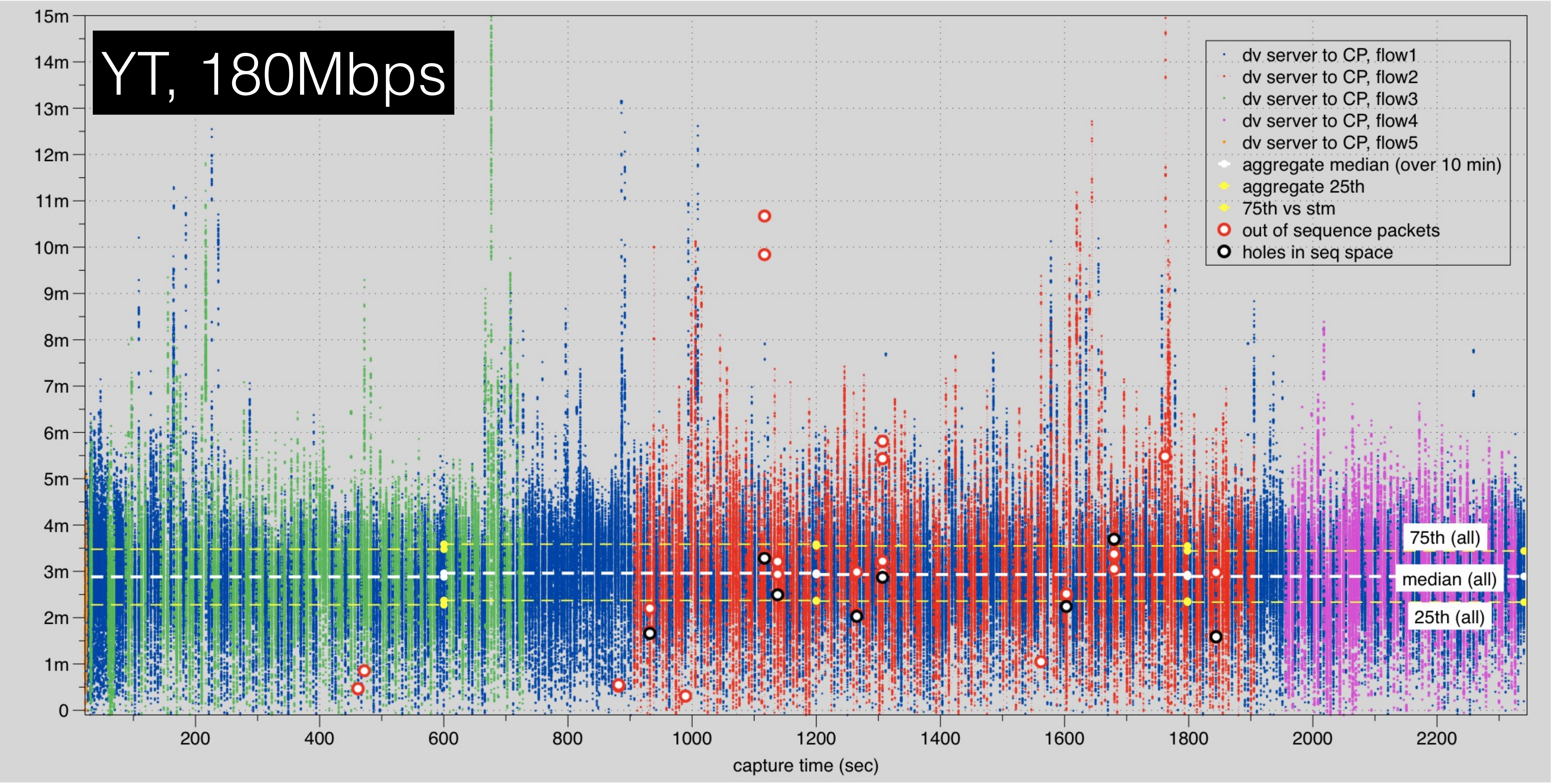
Taken 10.08.16
Seven flows from same server IP
Server minimum RTD is 88ms

- **Blue flow ~880Kbps overall (768Kbps after burst) in bursts**
- **Burst pattern of one short (~175KB) two long (~1MB) every 20 seconds.**
- **Arrival at CP up to 50Mbps**

More analysis possible adding sequence numbers



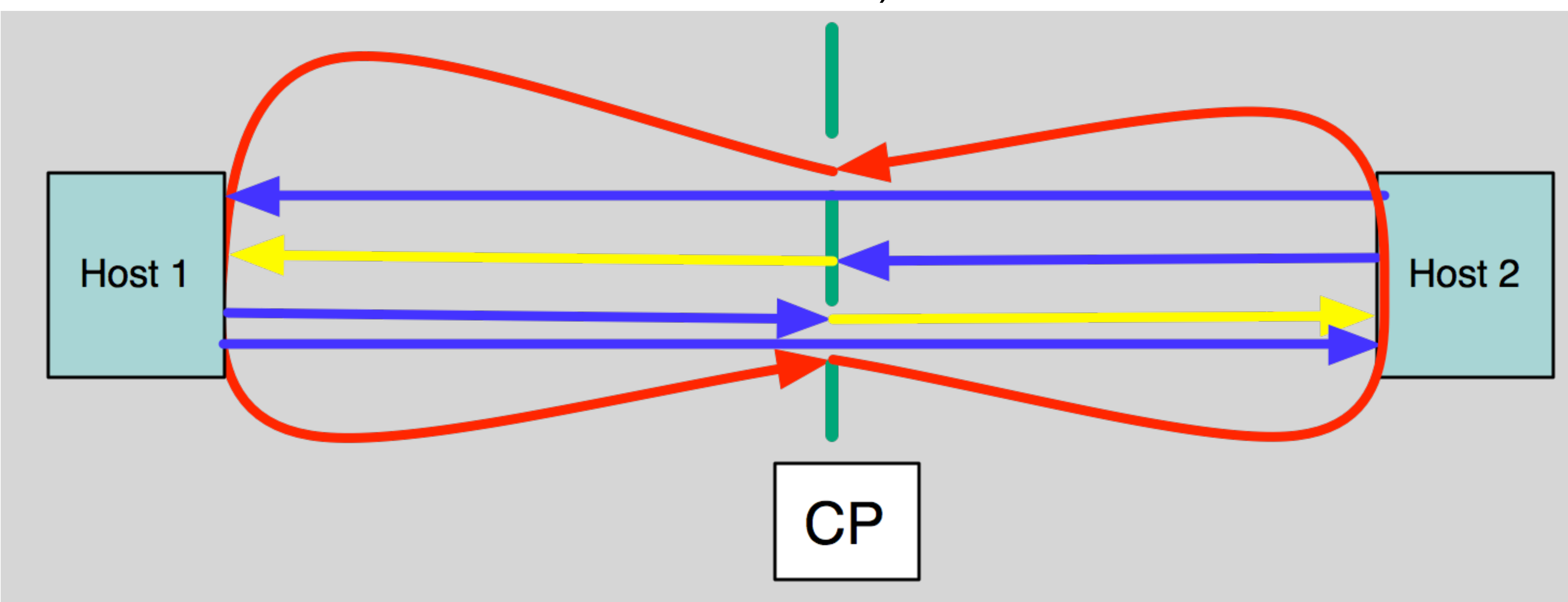
- This comes from post-processing packet trace
- Exploring ways to use seqno data in real time



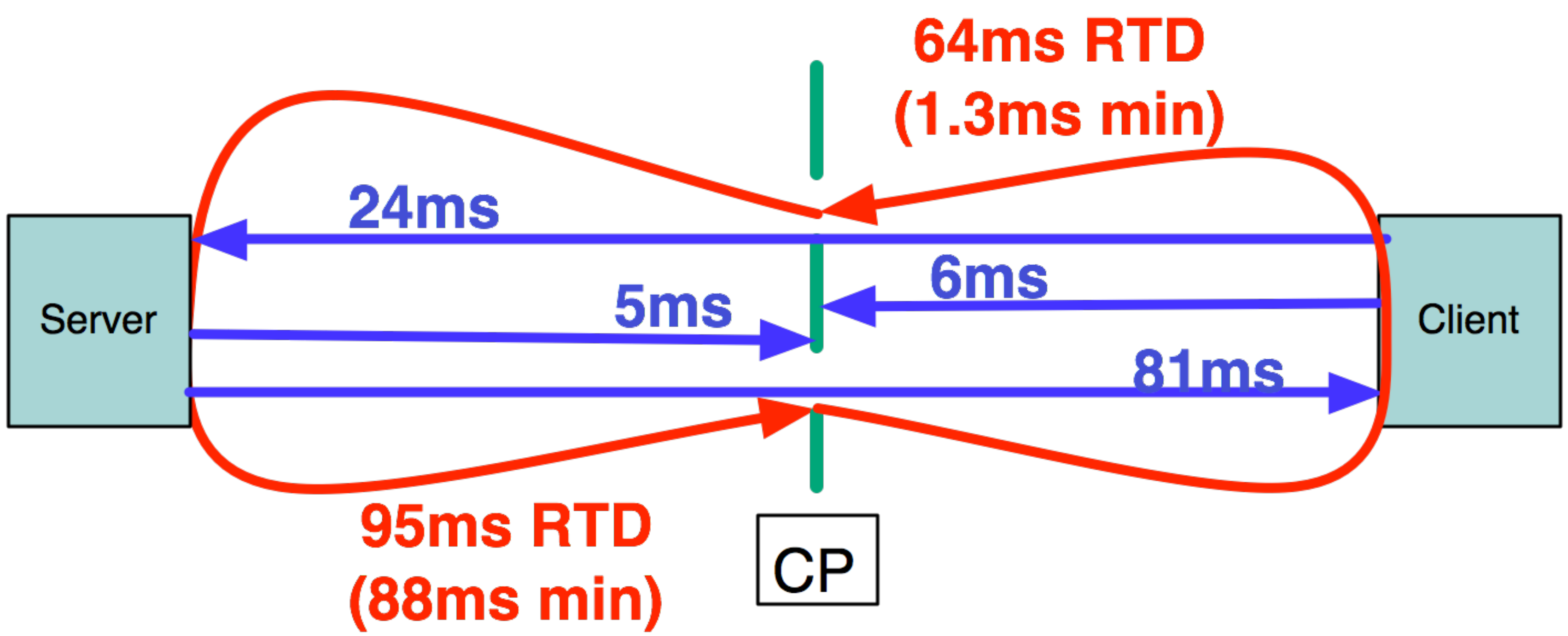
- Same YT video, different location on 10.26.16:180Mbps ISP link,11ms RTD, wifi link *seeing* ~45Mbps
- Only opens 5 flows (1 is only briefly active)
- Annotated with sequence space holes and out-of-orders

Host-to-CP delay variation just the tip of the iceberg

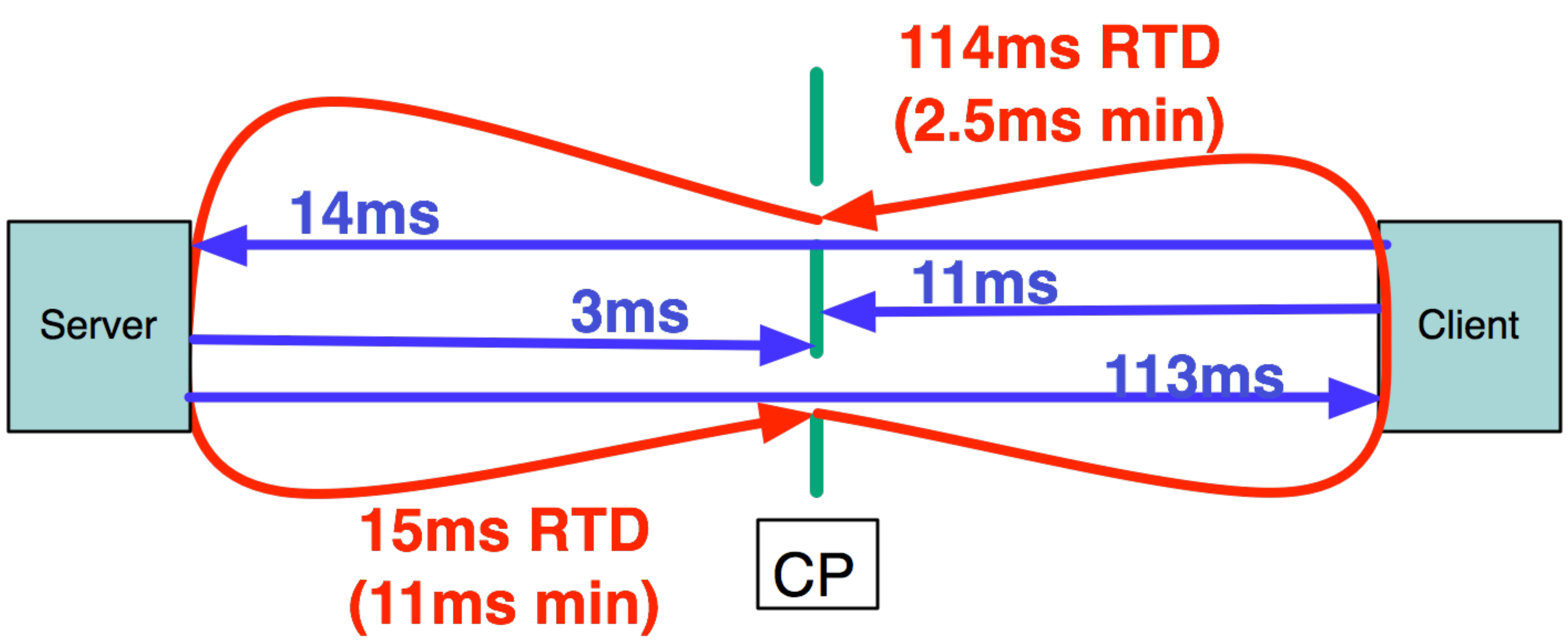
- Every packet provides delay estimates for several path segments (contrast this to ping probes)
- Packet header data can be used to *localize* delay
  - blue lines are delay variations
  - yellow lines are a noisier delay variation (available when CP sees both directions of a stream)



Localizing delay for YT10.08.16



Localizing delay for YT10.26.16

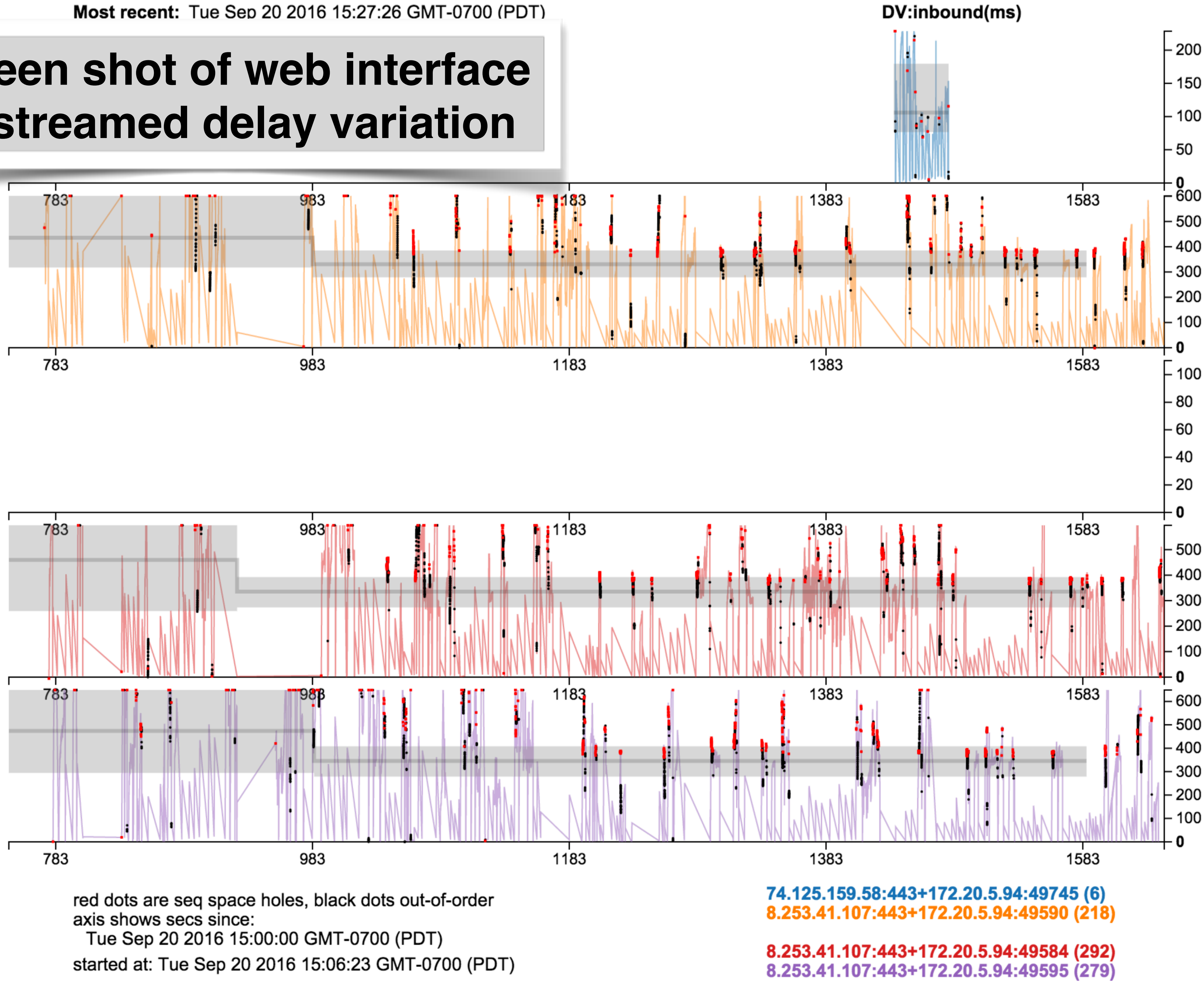


CP to client path has a large delay, could be application or wifi or both. (Same delays affect the server to client delay estimate.)

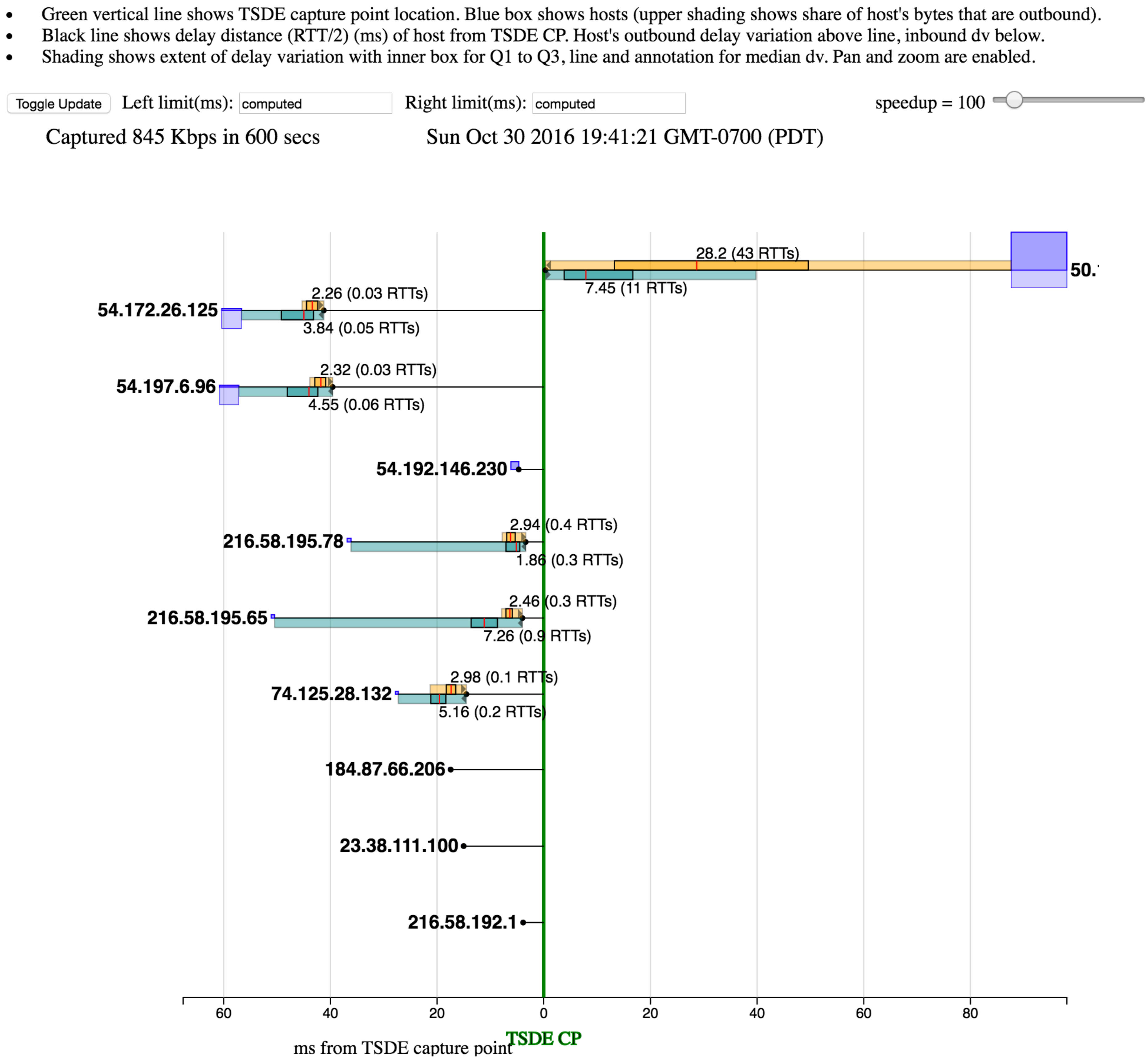
Building on Passive Packet Capture

- Packet capture a fundamental tool since early days of networking
- Facilitated by high-speed capture, sampling techniques ("heavy hitters"), span ports, etc.
- A *wealth* of information in packet headers
- Extracting data from headers and displaying in ***real-time*** harder than post-processing
- This presentation emphasizes delay since active measurement probes reveal little about application delay
- Would like to see more work using passive measurement of actual application traffic

Screen shot of web interface of streamed delay variation



This is a “delay topology” map. It updates on statistics periods which are usually set at 5 to 10 minutes. Stats are from a high quality “on the fly” estimator.



Video Streaming Takeaway

- Video streaming clearly shows the influence of the storage and application chunk structure
- Network behavior varies by client application (Apple “big bursts” average about 8 MBytes)
- Video is not a river of flowing bytes but looks more like big ocean waves
  - Innocuous looking waves turn ugly when they crash onto the beach of small bandwidth ISP tails, end-user wifi networks, low-speed device interfaces and other fast-to-slow pipes
  - Also some evidence of entire bursts being delayed in Internet
- For high speed provider links, client networks often are the problem and wifi can be the bottleneck

Passive Measurement Takeaway

- Packet header capture provides rich information (payload encryption doesn't matter) that active probes can't get
- Packet header capture capabilities in all devices would provide a basis for great diagnostics
- Good TSvals allow more and better information extraction
- Extracting information in real time is an interesting challenge
- Making sense of information in real time is a visualization challenge
- Challenging yourself is good, so get to it!

The Data and Its Processing

- The data used in this talk was collected via packet header capture (**tcpdump**) in end networks, mostly home networks. Although these pcap files will not be publicly available, it is easy to obtain similar ones.

- Netflix and YouTube videos were run on a variety of clients (Apple TV, iPad, Mac laptop, Chromecast, Windows desktop) connected via ethernet, Google and Apple 802.11ac routers to cable modems (unknown for hotel capture)

- Most packet captures were done using a bump-in-the-wire device but one was captured *on* the client

- Easy to replicate and extend analysis; post-processing of packet captures can be done with simple graphing tools and statistical packages

- This data used a proprietary method to extract clocks from the data; older ways exist to do this post-processing (V. Paxson, S. Moon).

- Round trip delays can be extracted from a two-way packet stream, see for example Marcondes et al 2007.

Resources

- V. Paxson, “On Calibrating Measurements of Packet Transit Times”, ACM Sigmetrics, 1998. [removing skew from traces]
- S. Moon, P. Skelly, and D. Towsley, “Estimation and Removal of Clock Skew from Network Delay Measurments”, Proceedings of INFOCOM 1999. [removing skew from traces: patented technique]
- C. Marcondes et. al., “Regenerating TCP Dynamics from Traces Path Characteristics”, 3rd International Conference on Testbeds and Research Infrastructure for Dir of Networks and Communications”, Orlando, FL, April 2007 [round trip delays from bidirectional packet traces]
- J. Martin et. al., “Characterizing Netflix Bandwidth Consumption”, IEEE Consumer Communications and Networking Conference, 2013
- More data like this at http://pollere.net/Pdfdocs/FunWithTSDE.pdf [real-time and post-processed delay, uses patent pending technique]