Low Latency Low Loss Scalable Throughput (L4S)

TCP Prague Status pt1
draft-ietf-tsvwg-ecn-l4s-id

Bob Briscoe, Independent <ietf@bobbriscoe.net>
about the work of people too numerous to list

TSVWG, IETF-106, Nov 2019

The 'Prague L4S requirements'

- for scalable congestion ctrls over Internet
  - Assuming only partial deployment of either FQ or DualQ Coupled AQM isolation for L4S
  - Jul 2015 Prague IETF,
    ad hoc meeting of ~30 DCTCP folks
  - categorized as safety (mandatory)
    or performance (optional)
- not just for TCP
  - behaviour for any wire protocol (TCP, QUIC, RTP, etc)
- evolved into draft IETF conditions for setting ECT(1) in IP
  - draft-ietf-tsvwg-ecn-l4s-id

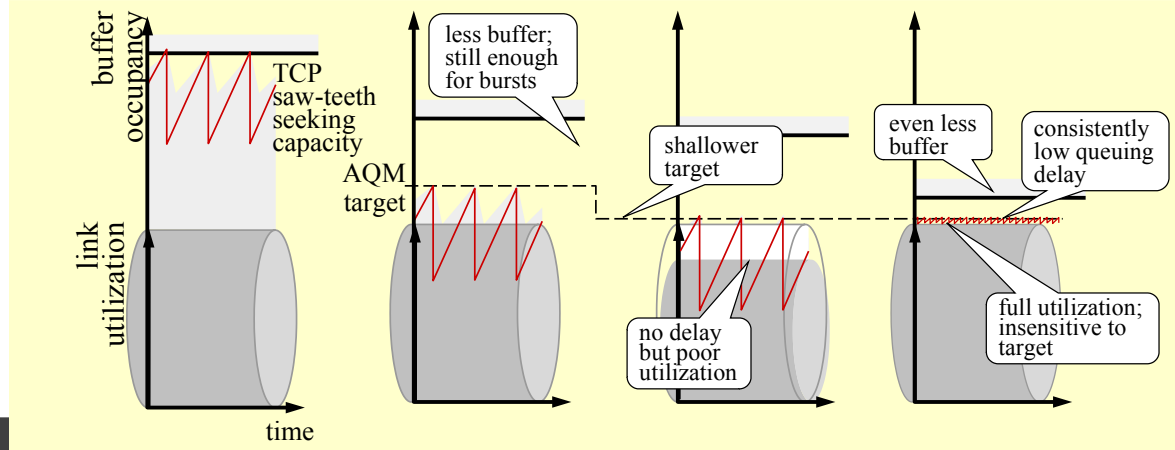
Requirements
L4S-ECN Packet Identification: ECT(1)
Accurate ECN TCP feedback
Reno-friendly on loss
Reno-friendly if Classic ECN bottleneck
Reduce RTT dependence
Scale down to fractional window
Detecting loss in units of time
Optimizations
ECN-capable TCP control packets
Faster flow start
Faster than additive increase

Motivation – recap

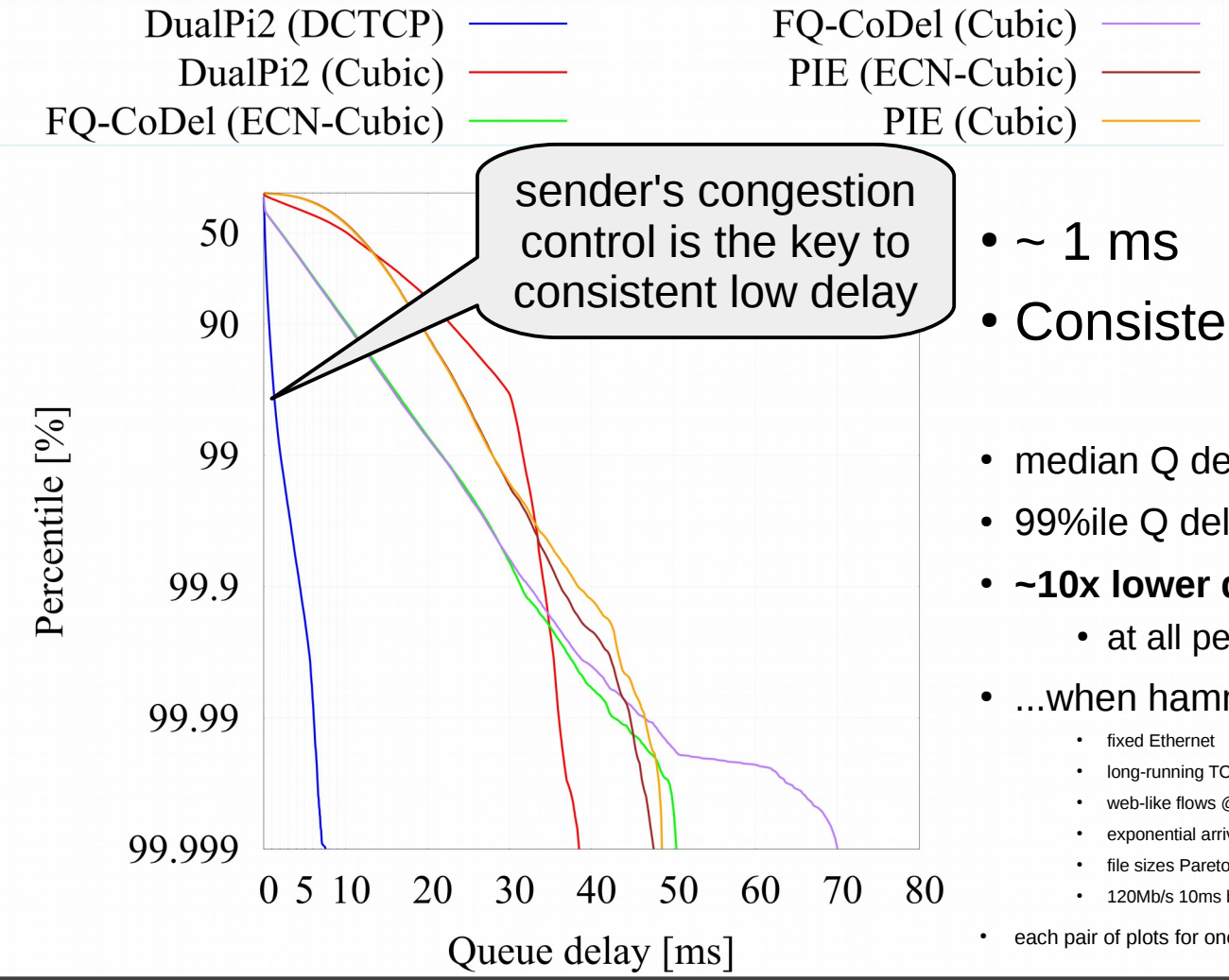
- Ultra-low queuing delay for *all* Internet applications
  - including capacity-seeking (TCP-like)
- Transition mechanisms
  - network side (not this talk)
  - dualQ coupled AQM
  - per-flow queuing

The trick: scalable congestion control

	(1) Today (typical)	(2) Today (at best)	(3) Unacceptable	(4) L4S
Bottleneck	Bloated drop-tail buffer	AQM	Shallower AQM	Immediate AQM
Sender CC	Classic	Classic	Classic	Scalable (tiny saw-teeth)



“Ultra-low” Q delay?



- ~ 1 ms
- Consistently – for real-time apps

- median Q delay: 100-200µs
- 99%ile Q delay: 1-2ms
- **~10x lower delay than best 2nd gen. AQM**
  - at all percentiles
- ...when hammering each AQM
  - fixed Ethernet
  - long-running TCPs: 1 ECN 1 non-ECN
  - web-like flows @ 300/s ECN, 300/s non-ECN
  - exponential arrival process
  - file sizes Pareto distr. α=0.9 1KB min 1MB max
  - 120Mb/s 10ms base RTT
- each pair of plots for one AQM is one experiment run

Status against Prague L4S requirements (Jul'19)

Linux code:	none	none (simulated)	research private	research opened	**RFC**	**mainline**

Requirements	base TCP	DCTCP	TCP Prague
L4S-ECN Packet Identification: ECT(1)		**module option**	**mandatory**
Accurate ECN TCP feedback	**sysctl option**	**?**	**mandatory**
Reno-friendly on loss		**inherent**	**inherent**
Reno-friendly if classic ECN bottleneck			open issue
Reduce RTT dependence			simulated
Scale down to fractional window	thesis write-up	thesis write-up	thesis write-up
Detecting loss in units of time	**default RACK**	**default RACK**	mandatory?
Optimizations			
ECN-capable TCP control packets	**module option off**	**on**	**default off→on later**
Faster flow start	in progress		
Faster than additive increase		in progress	

Status against Prague L4S requirements (Nov'19)

Linux code:	none	none (simulated)	research private	research opened	**RFC**	**mainline**

Requirements	base TCP	DCTCP	TCP Prague
L4S-ECN Packet Identification: ECT(1)		**module option**	**mandatory**
Accurate ECN TCP feedback	**sysctl option**	**?**	**mandatory**
Reno-friendly on loss		**inherent**	**inherent**
Reno-friendly if classic ECN bottleneck			evaluat'n in progress
Reduce RTT dependence			research code
Scale down to fractional window	research code	research code	research code
Detecting loss in units of time	**default RACK**	**default RACK**	mandatory?
Optimizations			
ECN-capable TCP control packets	**module option off**	**on**	**default off→on later**
Faster flow start	in progress		
Faster than additive increase		in progress	