Low Latency Low Loss Scalable Throughput (L4S)

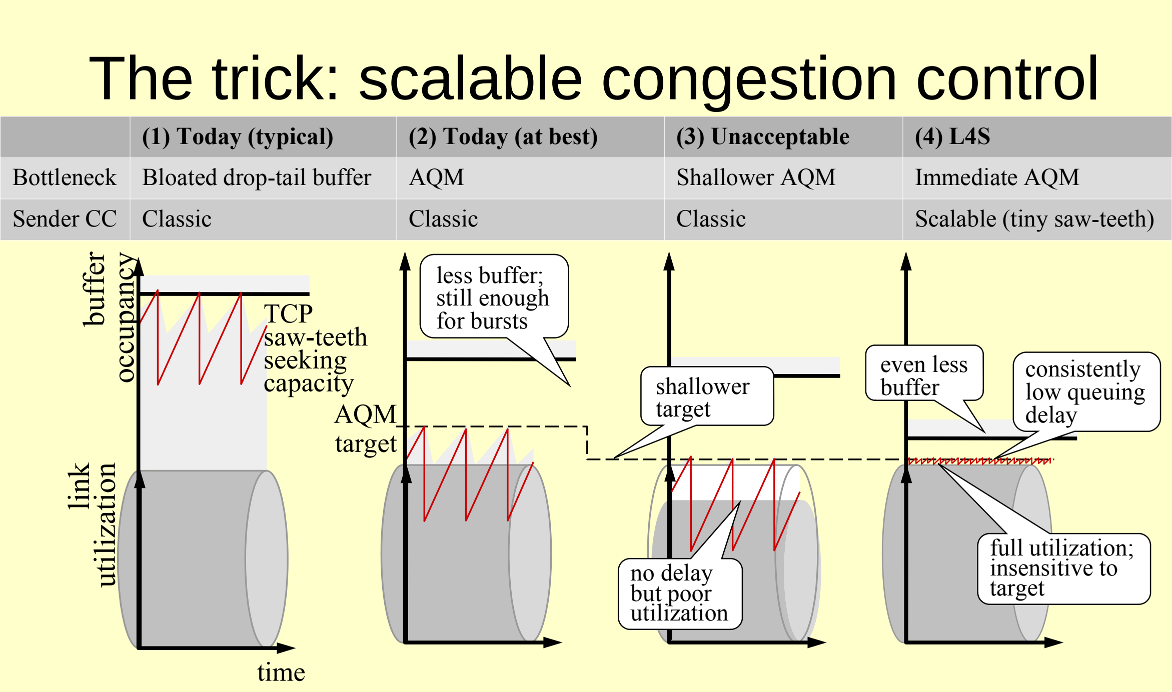
TCP Prague Status
draft-ietf-tsvwg-ecn-l4s-id

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

TSVWG, IETF-106, Nov 2019

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

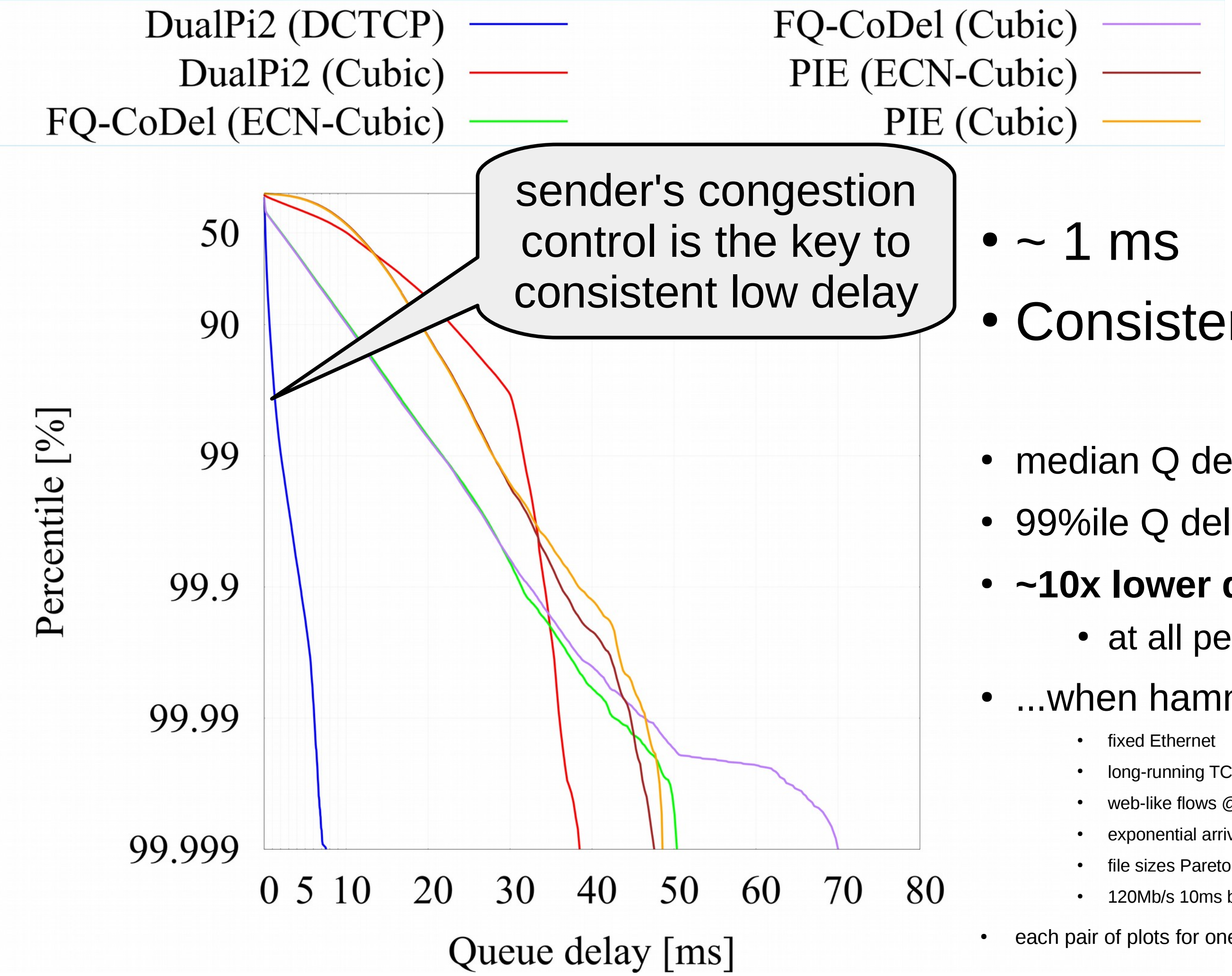


The trick: scalable congestion control

	(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)

buffer occupancy
link utilization
TCP saw-teeth seeking capacity
time
AQM target
less buffer; still enough for bursts
shallower target
no delay but poor utilization
even less buffer
consistently low queuing delay
full utilization; insensitive to target

“Ultra-low” Q delay?



sender's congestion control is the key to consistent low 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

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

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/BBRv2
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			in BBRv2 alpha
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	

Accurate ECN TCP feedback

- Ilpo Järvinen contracted to upstream TCP Prague
    - AccECN first on priority list
- Structured into sequenced patches (Hackathon-106)
    - in prep for upstreaming to Linux base TCP stack

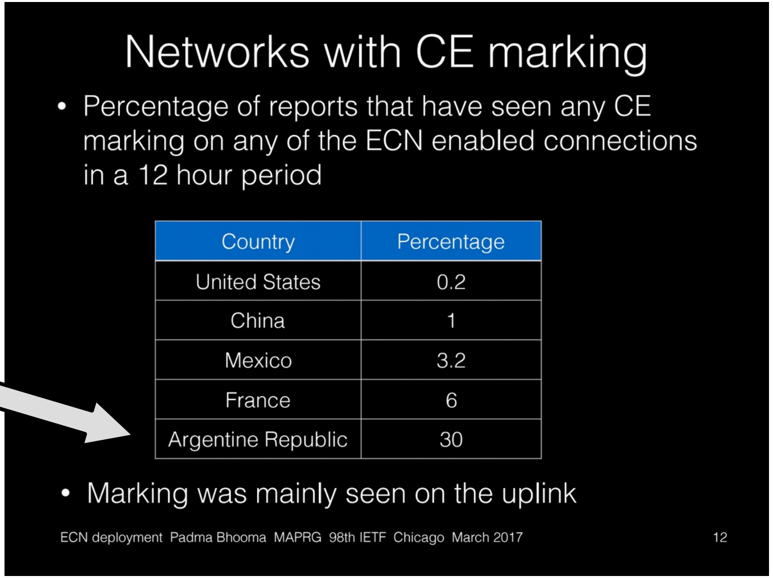
Reno-friendly if classic ECN bottleneck
(tsvwg issue #16)

Solution

- design
  - discussion paper – rationale analysis, pseudocode
- implementation – Asad Ahmed contracted for this
  - branch of Linux TCP Prague ref implementation
- evaluation
  - hackathon-106: testbed build in progress

Prevalence of problem?

- Argentinian ISP identified via Apple data
  - contacted – ToS byte overwrite – being fixed
- search for a single queue 3168 AQM continues



Networks with CE marking

- Percentage of reports that have seen any CE marking on any of the ECN enabled connections in a 12 hour period

Country	Percentage
United States	0.2
China	1
Mexico	3.2
France	6
Argentine Republic	30

- Marking was mainly seen on the uplink

ECN deployment Padma Bhooma MAPRG 98th IETF Chicago March 2017 12

Reduce RTT dependence

- Introduced into L4S ECN side of BBRv2
- Tested in various combinations of CC & AQM
  - during Hackathon-106
- More testing then design iteration in progress

Scale down to fractional window

- Designed, implemented (Linux base stack)
  and evaluated (Reno & TCP Prague)
  - works smoothly – complex design process, simple code
  - Research prototype
  - Not yet tested with other TCP Prague components
- Masters thesis of Asad Ahmed and open source code
  - link from L4S landing page
- Booked session to present in iccrg at IETF-107
  - brief preview in TCP Prague side meeting on Thu 08:30 (see next)

More this week...

TCP Prague Status Update: side meeting

- 08:30 – 09:30 Thu 21 Nov, Canning, IETF-106 Singapore
- Thursday, before tsvwg pt2, in same room
- will post remote access details (no meetecho)

L4S slot in tsvwg pt2

- DualQ Coupled AQM implementations and interops
    - Nokia L4S integration in WiFi Beacons product at BBWF19 (available Q1'2020)
    - Low Latency DOCSIS interops: CM hardware + CMTS implementations
    - 3GPP L4S ECN proposal into 3GPP

Open Source links

- Dual Queue Coupled AQM (Linux)
- L4S Demo/Test GUI (Linux)
- TCP Prague (ECT(1), ECN++, AccECN) (Linux)
- QUIC Prague (Linux, FreeBSD, Windows)
- SCReAM with L4S support (Linux, FreeBSD, Windows)
- BBRv2 with L4S support (Linux)
- ns3 network simulator L4S test suite
- Paced Chirping (Linux)
- all linked via L4S landing page https://riteproject.eu/dctth/#code

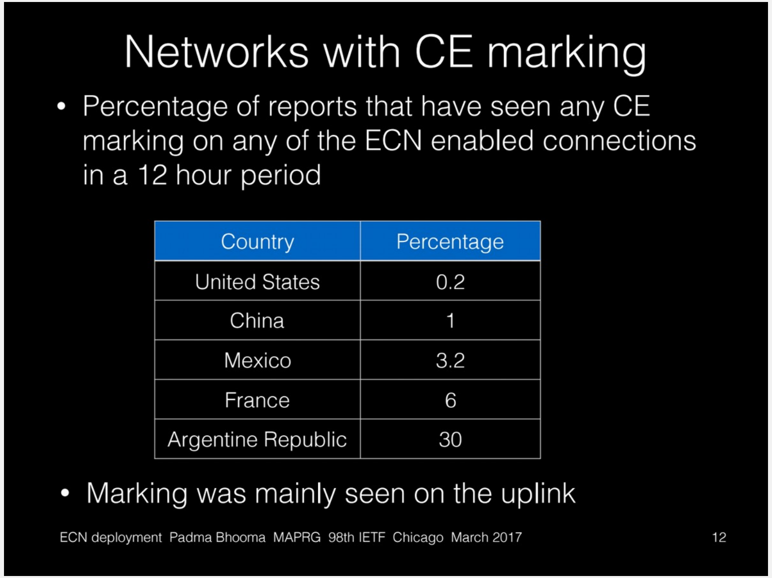
Low Latency Low Loss Scalable Throughput (L4S)

Q&A

Open issues #16:
RFC3168 ECN in a FIFO

- Nov 2016, after 16 months of deliberation
  - WG chose ECT(1) for L4S ECN
  - CE ambiguous, but least worst compromise
  - L4S ECN coexists with 3168 ECN, if it's all FQ
- All academic ECN studies over the years (incl. 2017, 2019) found virtually no CE marking
  - using active measurement
- Mar 2017 study by Apple found CE marking
  - using passive measurement

Codepoint	IP-ECN bits	Meaning
Not-ECT	00	Not ECN-Capable Transport
ECT(0)	10	Classic ECN-Capable Transport
ECT(1)	01	L4S ECN-Capable Transport
CE	11	Congestion Experienced



Networks with CE marking

- Percentage of reports that have seen any CE marking on any of the ECN enabled connections in a 12 hour period

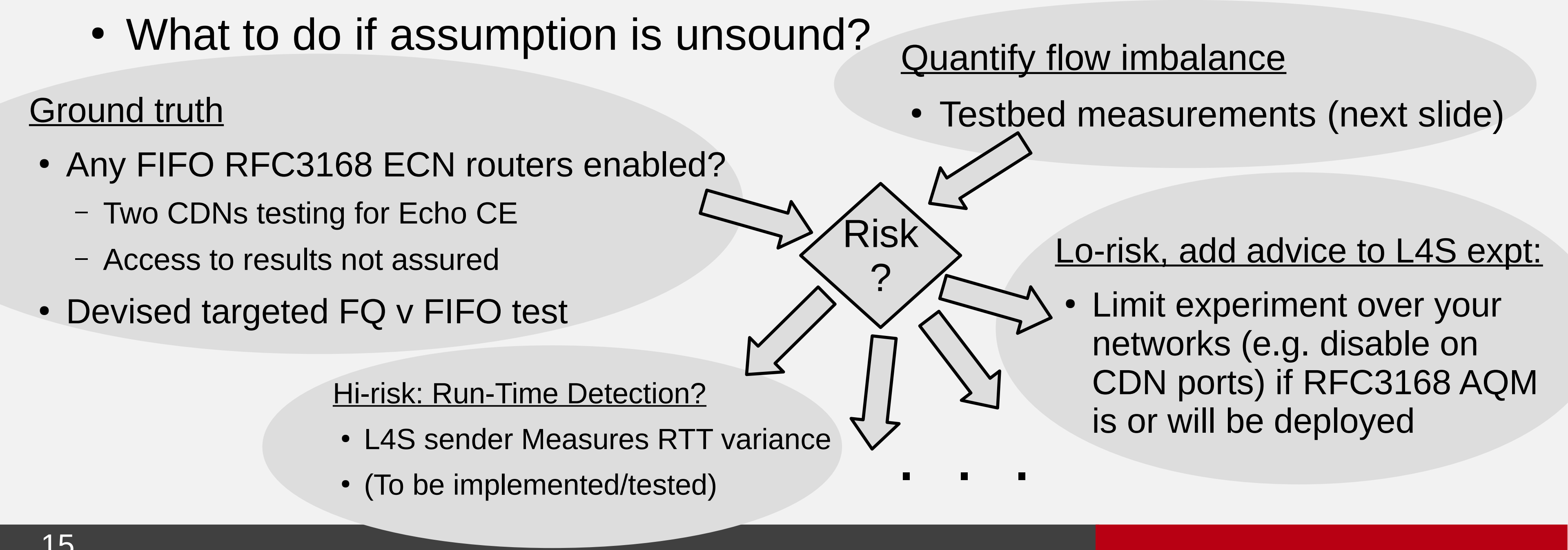
Country	Percentage
United States	0.2
China	1
Mexico	3.2
France	6
Argentine Republic	30

- Marking was mainly seen on the uplink

ECN deployment Padma Bhooma MAPRG 98th IETF Chicago March 2017 12

Open issues #1:
RFC3168 ECN in a FIFO

- Assumed all RFC3168 ECN AQMs likely to be FQ_CoDel
  - So L4S traffic would coexist with TCP-Friendly
- What to do if assumption is unsound?



<u>Ground truth</u>

- Any FIFO RFC3168 ECN routers enabled?
  - Two CDNs testing for Echo CE
  - Access to results not assured
- Devised targeted FQ v FIFO test

<u>Quantify flow imbalance</u>

- Testbed measurements (next slide)

Risk ?

<u>Lo-risk, add advice to L4S expt:</u>

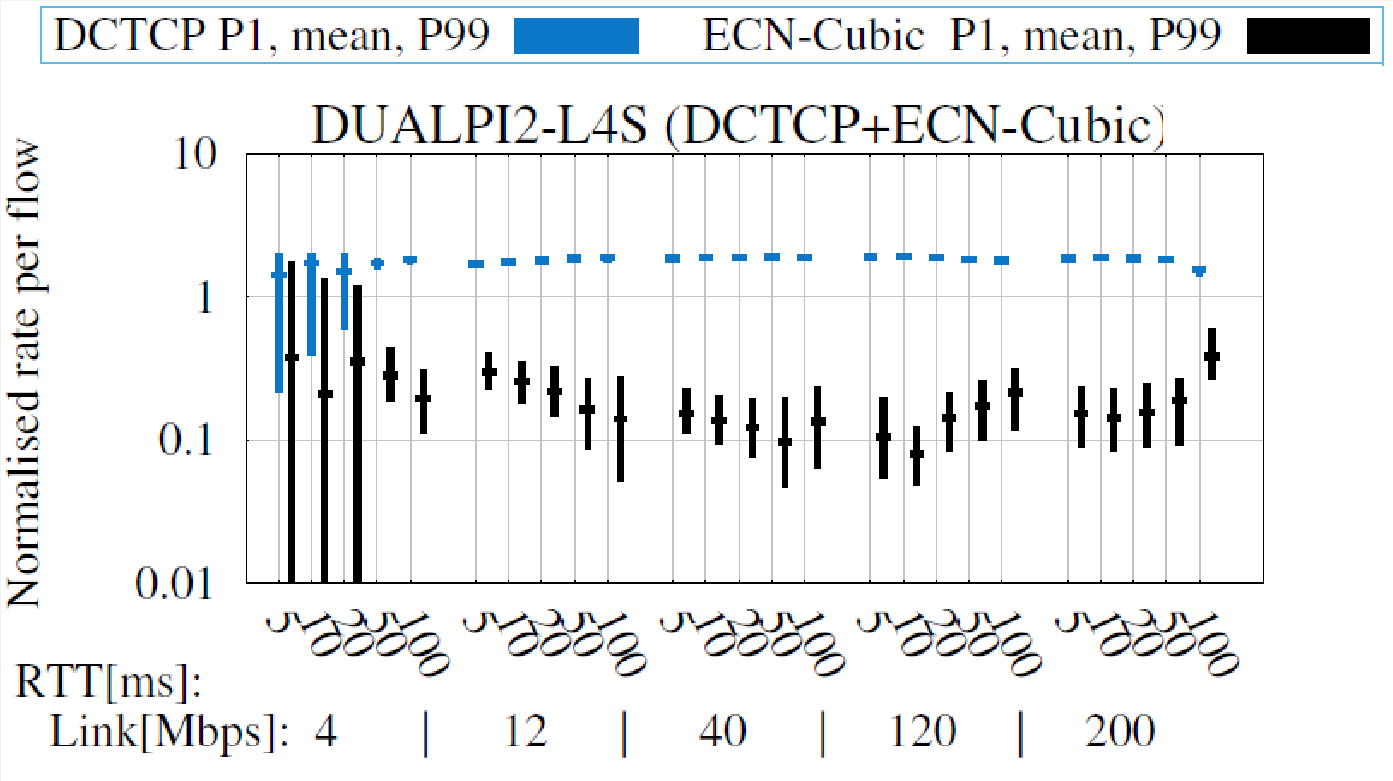
- Limit experiment over your networks (e.g. disable on CDN ports) if RFC3168 AQM is or will be deployed

<u>Hi-risk: Run-Time Detection?</u>

- L4S sender Measures RTT variance
- (To be implemented/tested)

. . .

Open issues #1:
RFC3168 ECN in a FIFO


DCTCP P1, mean, P99
ECN-Cubic P1, mean, P99
DUALPI2-L4S (DCTCP+ECN-Cubic)
Normalised rate per flow
10
1
0.1
0.01
5 10 20 50 100
5 10 20 50 100
5 10 20 50 100
5 10 20 50 100
5 10 20 50 100
RTT[ms]:
Link[Mbps]: 4 | 12 | 40 | 120 | 200