

Low Latency Low Loss Scalable Throughput (L4S)

Bob Briscoe, **CableLabs**[®]

<ietf@bobbriscoe.net>



Koen De Schepper, **NOKIA** Bell Labs <koen.de_schepper@nokia.com>

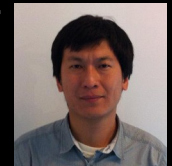


Olga Bondarenko, **simula**

<olgabnd@gmail.com>

Ing-jyh (Inton) Tsang, **NOKIA**

<ing-jyh.tsang@nokia.com>



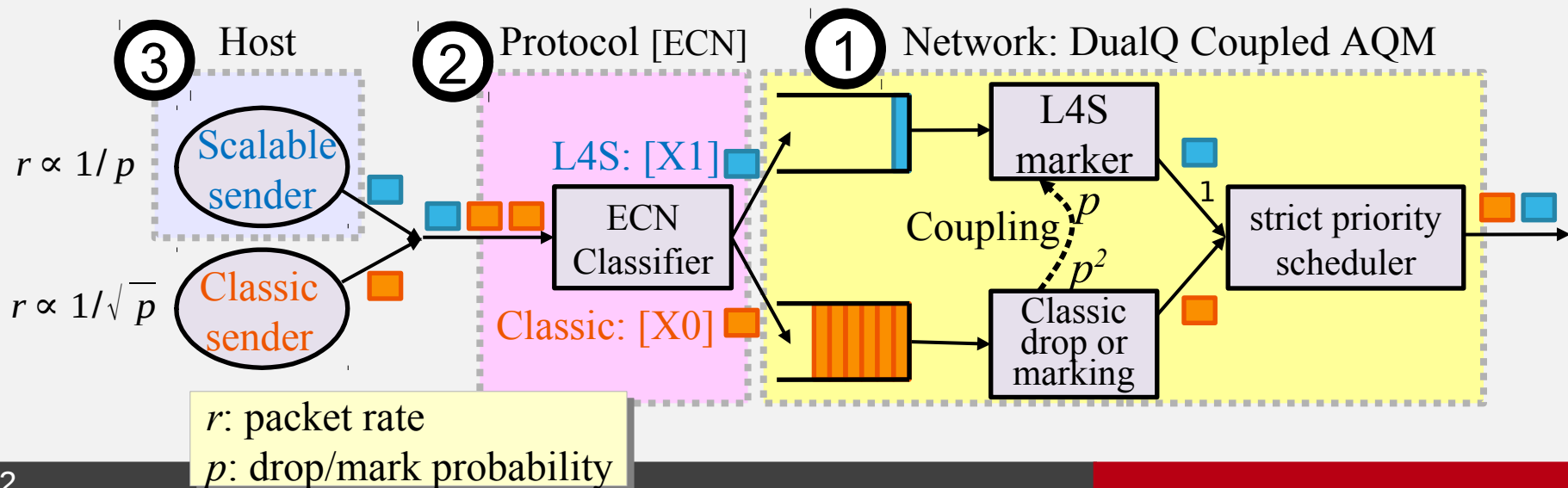
TSVWG, IETF-102, Jul 2018

L4S Recap





- Motivation

- Extremely low queuing delay for all Internet traffic
- already 1-2 orders better than state of the art
- 500 μ s vs 5-15 ms (fq-CoDel or PIE)

- Architecture





Outline

- L4S drafts in tsvwg
 - L4S Internet Service: Architecture
draft-ietf-tsvwg-l4s-arch-02 (-02) [stable]
 - Identifying Modified ECN Semantics for Ultra-Low Queuing Delay (L4S)
draft-ietf-tsvwg-ecn-l4s-id-03 (-02)  New TCP-RACK-like requirement
 - DualQ Coupled AQMs for L4S
draft-ietf-tsvwg-aqm-dualq-coupled-06 (-04)  Handling the unexpected
 - Interactions between L4S and Diffserv
draft-briscoe-tsvwg-l4s-diffserv-01 (-00)  Couple of technical updates
- L4S Status Update, beyond tsvwg
 - tcpm, iccr, implementation, etc  Numerous Heads-ups
- Next Steps

Identifying Modified ECN Semantics for Ultra-Low Queuing Delay (L4S)

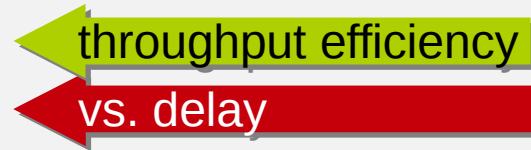
draft-ietf-tsvwg-ecn-l4s-id-03

Proposed 5th Requirement for L4S senders

- 'TCP Prague' Requirements (for all transports, not just TCP)
draft-ietf-tsvwg-ecn-l4s-id-03#section-2.4.1
- to use ECT(1), a scalable congestion control MUST detect loss:
 - by counting in units of time  like TCP RACK
 - not in units of packets  like the TCP 3DupACK rule
- Then link technologies that support L4S can **remove head-of-line blocking delay**
 - see next slides (or Appendix A.1.7) for rationale

Recent ACKnowledgements (RACK): Background

- Loss is when sender deems absence has been long enough
 - Classic TCP: 3 DupACKs
 - TCP RACK: a fraction (ϵ) of the RTT (termed the reordering window)
- Tradeoff – larger ϵ :
 - minimizes spurious retransmissions (before ACKs of reordered packets arrives)
 - but takes longer $(1+\epsilon)*RTT$ to repair genuine losses
- So, RACK adapts the reordering window:
 - starts small (which rapidly repairs losses in short flows)
 - then adapts to measured reordering degree (rapid loss repair less critical for performance of elephants)
- See [draft-ietf-tcpm-rack-04](#)



Benefits of universal RACK to links (1/2)

- as well as e2e (layer-4) benefits, RACK offers potential for link (layer-2) performance improvements

- as flow rates scale up

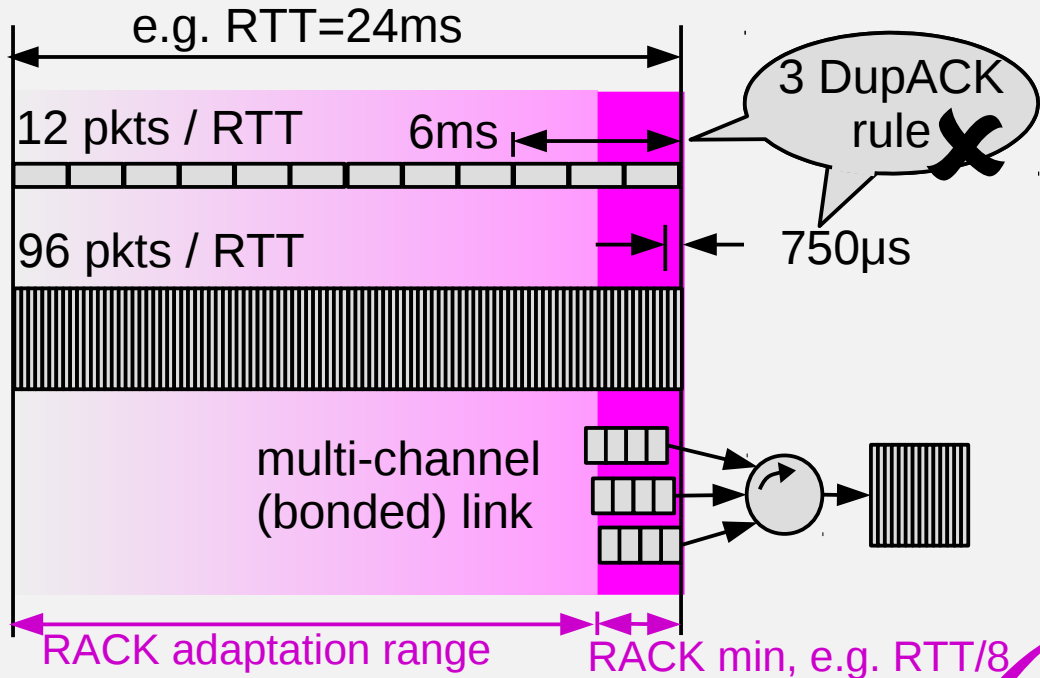
- with 3 DupACK rule

- reordering tolerance time scales down
- for multi-channel (bonded) links, skew tolerance time scales down

- with rule relative to RTT

- tolerance time remains constant

(given min practical e2e RTT remains fairly constant)

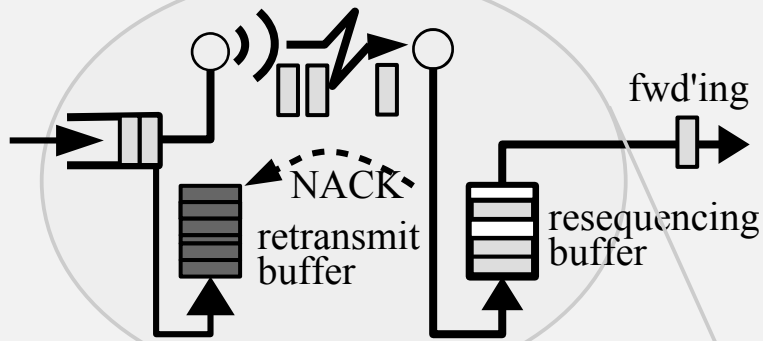


Benefits of universal RACK to links (2/2)

- for lossy links (e.g. radio)

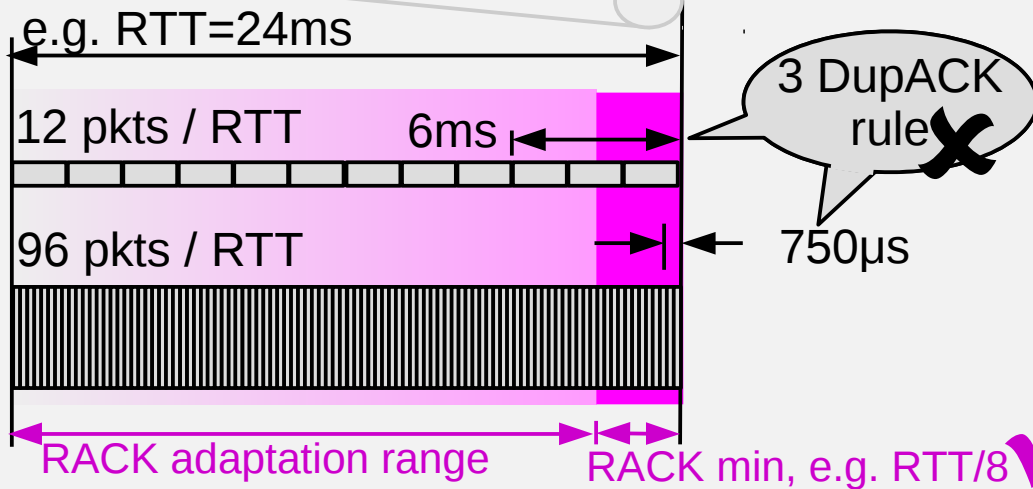
- with 3 DupACK rule

- link rcvr buffers packets behind each gap while link re-xmts
 - head-of-line blocking
 - recall that packets on a link will be from different flows and different streams within flows



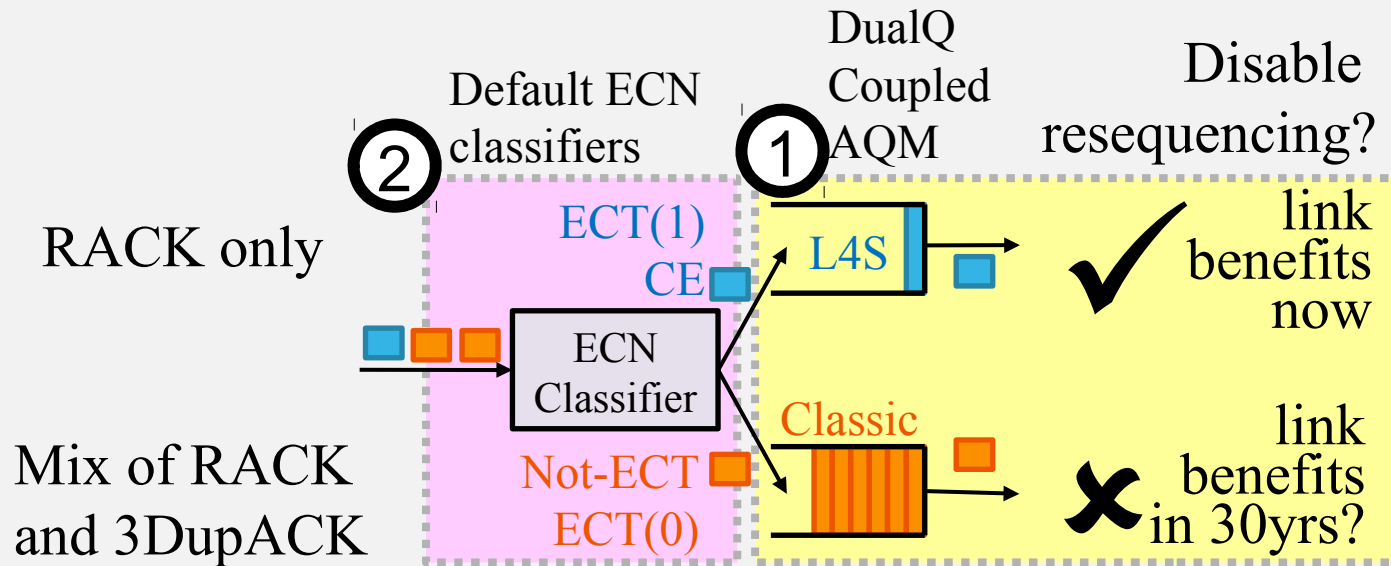
- with rule relative to RTT

- link rcvr can forward packets out of order
 - no reordering buffer
 - in parallel, link rexmt will typically fill gap within min RACK reordering window



Why the “MUST NOT”?

- “to use ECT(1), a scalable congestion control MUST NOT detect loss in units of packets”



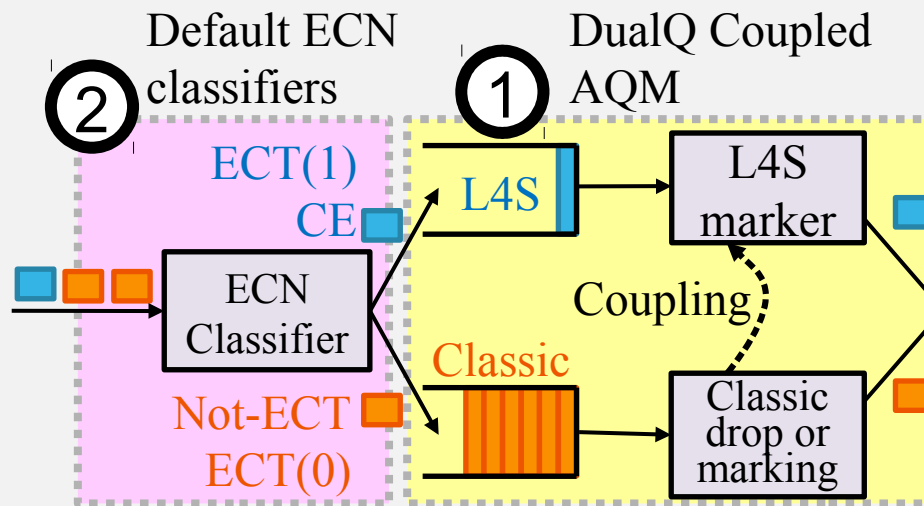
DualQ Coupled AQMs for L4S

draft-ietf-tsvwg-aqm-dualq-coupled-06

Handling the Unexpected (1/2)

draft-ietf-tsvwg-aqm-dualq-coupled-06#section-2.5.1.1

- A consequence of classifier flexibility for operators
- Use-cases for non-default classifiers:
 - **Not-ECT** or **ECT(0)** in **L4S Q**:
 - Unresponsive but low rate, non-queue-building traffic e.g. Diffserv EF, DNS
 - Adaptive to heavy congestion but low rate, e.g. adaptive VoIP
 - **ECT(1)** in **Classic Q**:
 - L4S Policing or Queue Protection
- What should each AQM implementation do...?



Handling the Unexpected (2/2)

- L4S queue, but ECT(0) or Not-ECT
 - if ECT(0), drop prob for Classic congestion control but with target L delay
 - if Not-ECT, depends if there's queue protection against misbehaving flows
 - If yes: AQM ignores packet and forwards it
 - If no: CE-marking prob for Classic congestion control but with target L delay
- Classic queue, but ECT(1)
 - CE-marking using coupled probability p_{CL} ($= k \cdot p'$)
- All requirements expressed as “SHOULDs”, cos we don't know the rationale for these unexpected cases

Interactions between L4S and Diffserv

draft-briscoe-tsvwg-l4s-diffserv-01

Interaction with Diffserv

draft-briscoe-tsvwg-l4s-diffserv-01

- Interaction of Inter-Q scheduler with Diffserv scheduler
 - Even if both WRR, cannot use one scheduler
- CS5 (app signalling) no longer considered L4S-compatible
 - Used as broadcast video by at least one major vendor

L4S status update (1/2)

[ToDo: Update this copy of Mar'18]

- Landing page for code, specs, papers

<https://riteproject.eu/dctth/>

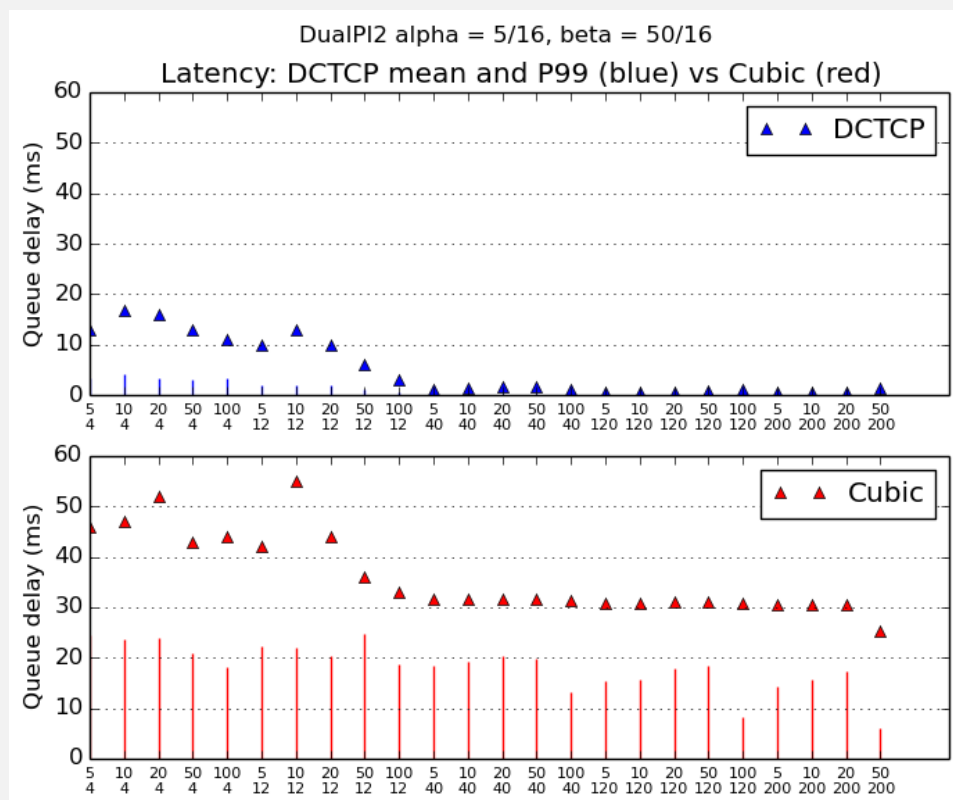
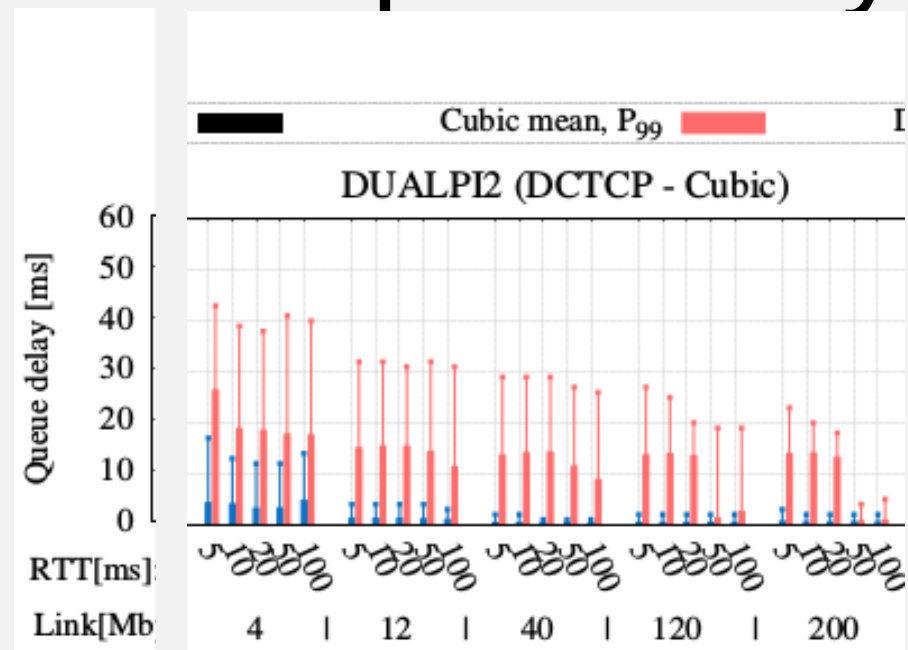
- Source Code

- Dual Queue Coupled AQM, DualPI2 for Linux [UPDATE in progress]
- Data Centre TCP (DCTCP) for Linux (in the mainline kernel), FreeBSD patch, ns2 patch.
- Accurate ECN TCP Feedback for Linux [testing needed]

- Implementations

- DualQ Coupled AQM: in at least one chipset aimed at DC environment [availability TBA]
- L4S Scalable congestion control: rmcat SCReAM
- BBRevo, evolution of BBR with L4S support
- Whole L4S system in ns3 [complete (see next) but evolving, release ns-3.30 (Aug'18)]

repeatability of L4S results



Exerpt from 2015 L4S paper:
testbed from Data Centre over DSL to home

ns3 over DualPI2,
using Linux DCTCP via Direct Code Execution

L4S status update: IETF specs (2/2)

Deltas since last IETF in Red

- L4S Internet Service: Architecture <draft-ietf-tsvwg-l4s-arch-02> [stable]
- Identifying Modified ECN Semantics for Ultra-Low Queuing Delay (L4S) <draft-ietf-tsvwg-ecn-l4s-id-03> [UPDATE]
- DualQ Coupled AQMs for L4S: : <draft-ietf-tsvwg-aqm-dualq-coupled-06> [2 UPDATES]
- Interactions of L4S with Diffserv <draft-briscoe-tsvwg-l4s-diffserv-01> [Minor UPDATES]
- enabled by <RFC8311> [RFC published]
- scalable TCP algorithms, e.g. Data Centre TCP (DCTCP) <RFC8257>, TCP Prague
- Accurate ECN: <draft-ietf-tcpm-accurate-ecn-07> [UPDATED – approaching WGLC]
- ECN++ Adding ECN to TCP control packets: <draft-ietf-tcpm-generalized-ecn-02> [Expired] [Supporting measurement paper published in IEEECommMag]
- ECN support in trill <draft-ietf-trill-ecn-support-07>, motivated by L4S [4 updates, RFC Ed Q]
- ECN in QUIC <draft-ietf-quic-transport-13>, [motivated by L4S] [In v1 of QUIC transport]
- ECN and Congestion Feedback Using the Network Service Header (NSH) <draft-eastlake-sfc-nsh-ecn-support-01> [UPDATE] [supports L4S-ECN]

Next Steps

- We can now leave holding pattern
 - sufficient progress on TCP Prague requirements within the stable architecture
- Acknowledge those who have been contributing
- Tidy up 3 years of piecemeal changes
 - Review all the drafts for inconsistencies
 - Probably have to rewrite the introductions
- Invite reviews
- Then ready for WGLC (target Sep'18)
- Reviewers for the draft on L4S-Diffserv interactions
 - adoption in the next cycle?

