L4S: Low Latency, Low Loss, Scalable Throughput Internet Service

TSVWG IETF98 Chicago

draft-briscoe-tsvwg-l4s-arch-01

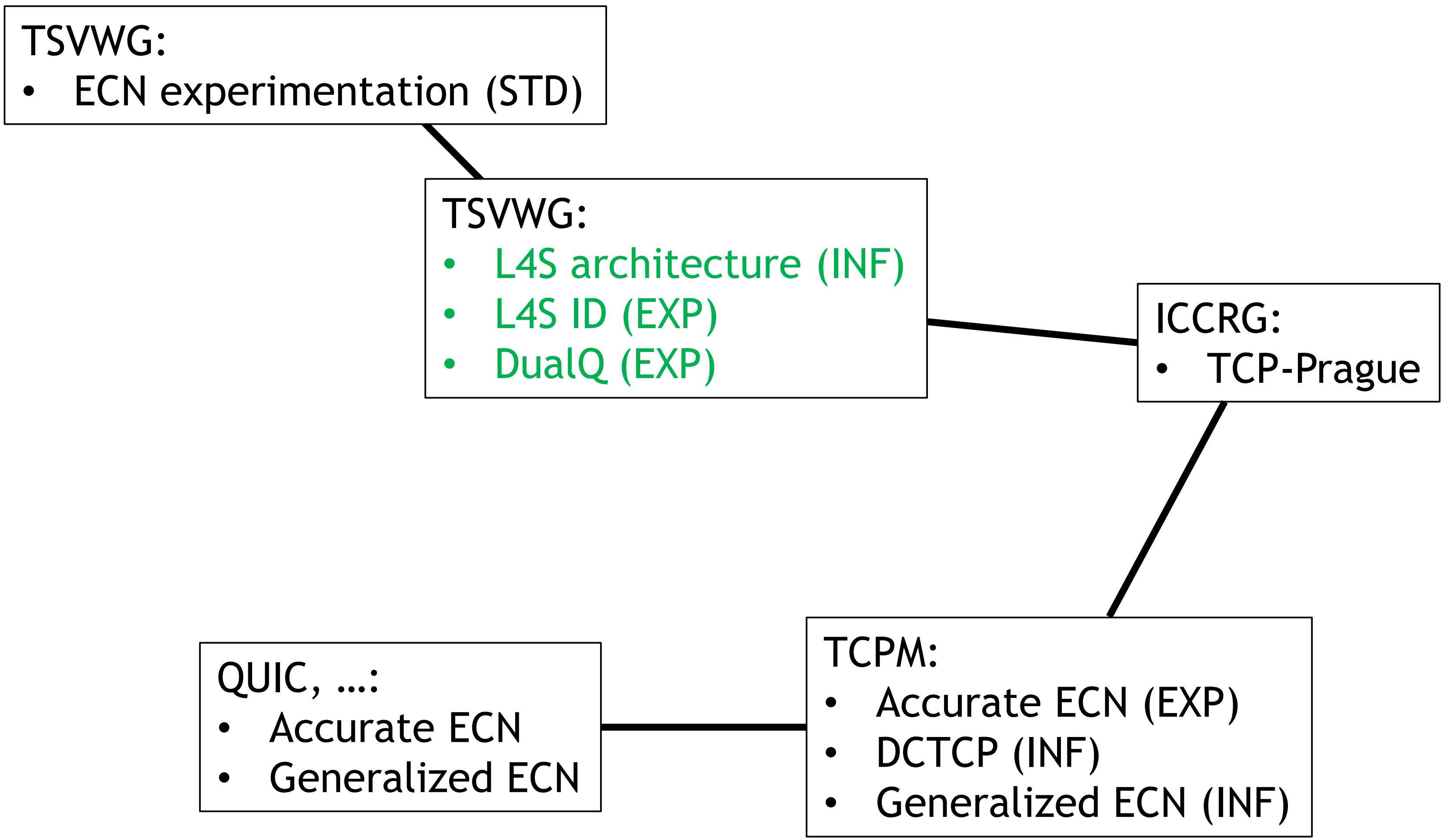
draft-briscoe-tsvwg-ecn-l4s-id-02

draft-briscoe-tsvwg-aqm-dualq-coupled-00

March 30th, 2017

Bob Briscoe, Koen De Schepper, Marcelo Bagnulo,
Olga Bondarenko, Inton Tsang

Drafts for L4S

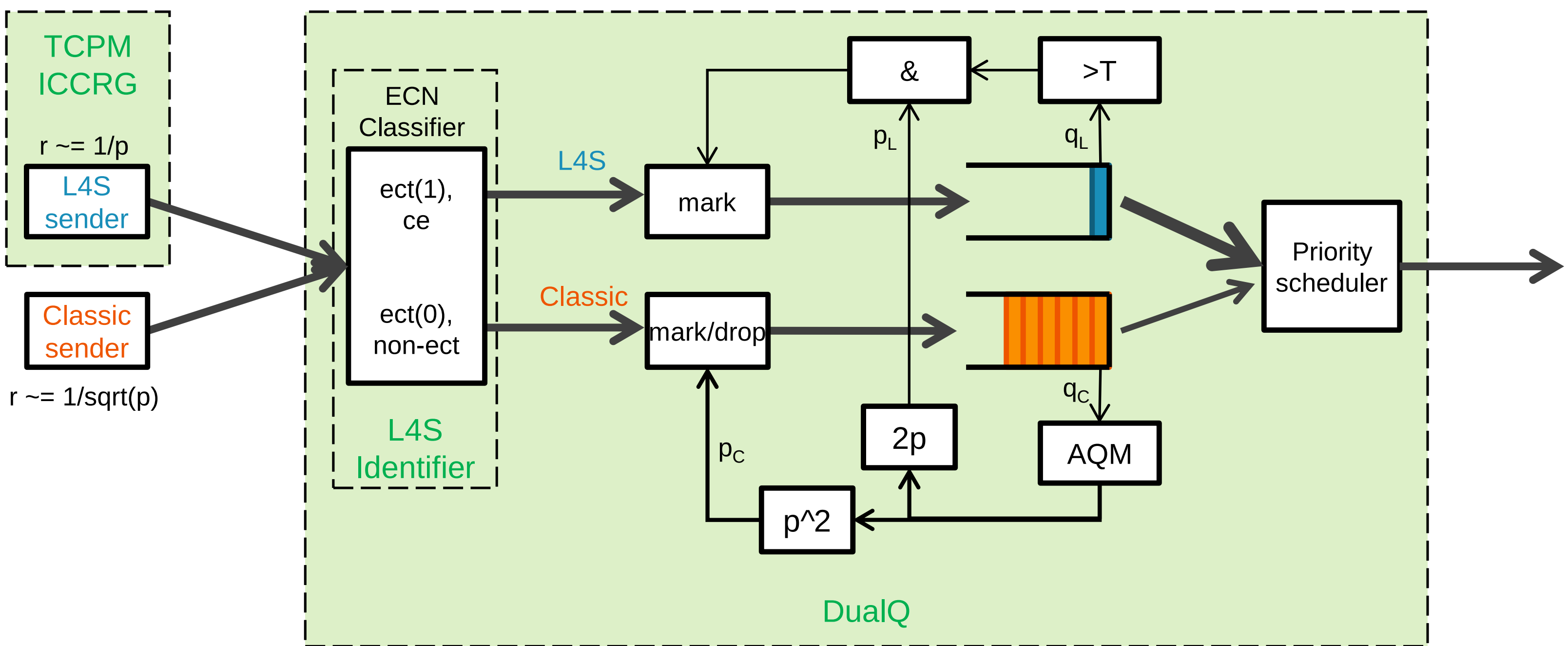


Recap

Motivation:

- Support Low Latency Congestion Controls (DCTCP, TCP-Prague)
- Compatibility with Classic Congestion Controls to support deployability

Architecture and draft mapping:



Status L4S Architecture

Updated sections in draft:

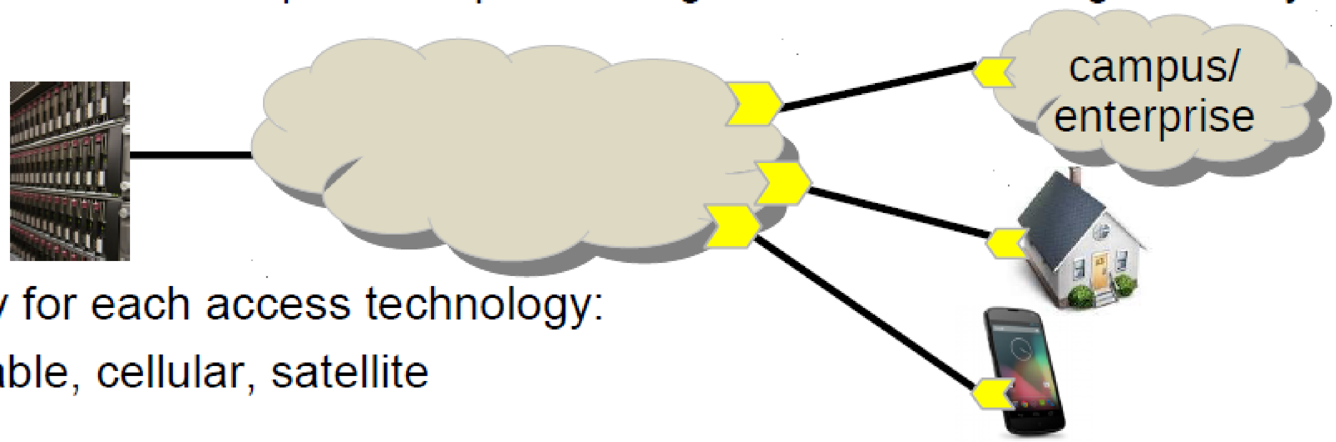
- Deployment
- Policing

thanks to Karen Nielssen, Wes Eddy & David Black

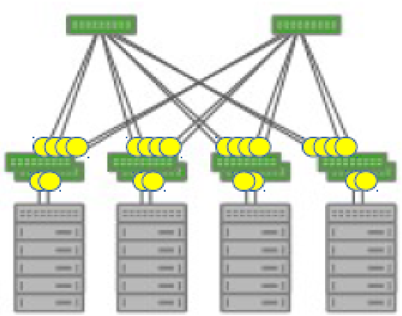
No other open issues?

DualQ AQM Deployment

- per 'site' (home, office, campus or mobile device)
    - typically one access bottleneck in each direction
    - deploying DualQ at each end gives nearly all the benefit
    - downstream as leaf queues of pre-existing inter-site scheduling hierarchy



- similar topology for each access technology:
    - xDSL, cable, cellular, satellite

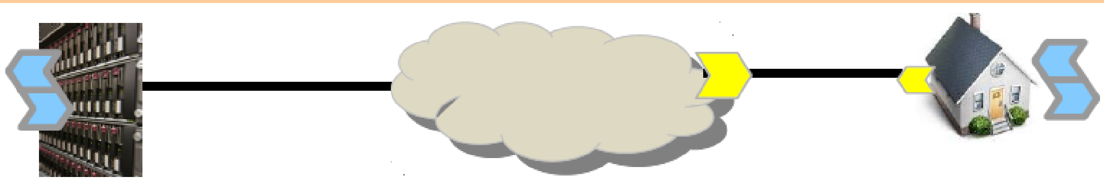


- mesh topology, e.g. data centre
    - ingress and egress bottleneck gives nearly all the benefit
    - e.g. all the outputs of the top-of-rack switch
- eventually deploy at internetwork bottlenecks, e.g.
    - inter-DC WAN links, Internet exchanges

L4S Deployment Sequences

Significant benefit realized at each deployment stage

	servers or proxies	access link	client
1.	DCTCP (existing)	DualQ AQM downstream	DCTCP (existing)
	works downstream for controlled trials		
2.	TCP Prague		AccECN (already in progress – DCTCP/BBR)
	works downstream		
3.		DualQ AQM upstream	TCP Prague
	works upstream & downstream		



Where a stage involves 2 moves:

- The benefit after the 2nd move has to be worth the 1st mover's investment risk
- new services or products, not just incremental performance improvement

When TCP Prague hits non-DualQ bottleneck?

- on drop, DCTCP already falls back to Reno for 1RTT
- but prevalent drop would degrade L4S
- main reasons for prevalent drop:
  - congestion loss (bursty traffic on shallow Q, long RTT)
  - transmission loss (high link rates)
  - policer loss
- 3 complementary approaches to address these (all research)
  1. include evolved BBR-like[1] behaviour in TCP Prague if there's consensus on how to safely interop with drop based CC (RTT-Independence?)
  2. exploit RACK[2]/link ARQ/L4S combination (research to appear)
  3. operator deploys L4S-ECN-enabled policers (see text in L4S-arch draft)

[1] BBR: Bottleneck Bandwidth & RTT – see ICCRG talks

[2] RACK: Recent ACKnowledgement – see TCPM

Status L4S Identifier

Draft is stable

Open issues:

- Ect(1) behavior for classic only single queue AQM
  - Default: Drop to avoid unnecessary Classic ECN detection
  - Optionally configurable: Also classic ECN marking
  - Optionally configurable: Also L4S ECN marking: 2* sqrt(p) marking

Status DualQ

DualQ AQM was main focus up to now

- Classic and DCTCP window compatibility
- PI2 as the classic AQM
- Overload handling
- Large number of experiments: flow numbers, RTTs, dynamic flows, overload

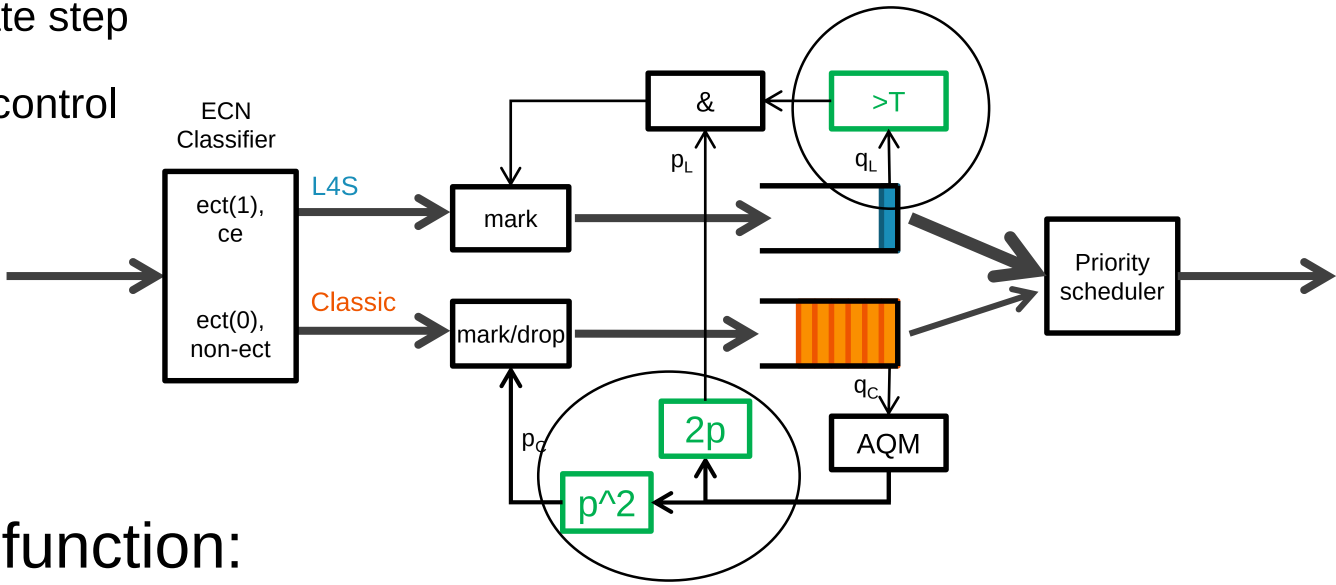
DualQ concept proven for DCTCP

- Linux open source released
- Cleanup and Linux upstream submission ongoing

DualQ open issues

L4S-only AQM:

- DCTCP-like immediate step
- AQM with gradual p control



DualQ Coupling function:

- Classic TCP-fairness is well known: 1/sqrt(p)   but future?
- Also coupling is determined by how DCTCP / TCP-Prague behaves
- RTT-independent related coupling

DualQ: minor open issues

- PI2 Classic AQM: all PIE heuristics have been assessed. Write-up will follow why each one is not relevant to PI2. Any heuristics that the PIE authors believe we should not have left out?
- PI2 API: Inter-dependency between parameters (e.g. coupling factor and ECN overload switch-over threshold and gain factors, etc)
- Experimentation to prove time-shift value for shifted-FIFO scheduler is optimal

Related recent TCP-Prague work

Internet-safety:

- 4.1: Fall back to Reno/Cubic congestion control on packet loss
- 4.2: Fall back to Reno/Cubic congestion control on classic ECN bottlenecks
- 4.3: Reduce RTT dependence
- 4.4: Scaling down the congestion window
- tcpm: Accurate ECN and negotiation draft-ietf-tcpm-accurate-ecn

Performance improvements:

- 5.1: Setting ECT in SYN, SYN/ACK and pure ACK packets
- 5.2: Faster than additive increase
- 5.3: Faster convergence to fairness

no impact   work in progress, maybe impact

TCP-Prague discussion points

- Use TCP-Prague also in DC?
- Compatible with DCTCP ?
- Interoperability/coexistence needed between DC and public Internet?
- Possible new congestion control features that L4S hosts are required to support
  - RACK-like support (why relevant? - writing up in progress)
  - others...?
  - any legacy features we could require to not be supported?

Next steps

L4S - DualQ concept proven and usable with DCTCP

- Independent evaluation will help improve the drafts
- Hands-on experience is required before designing HW. Many pitfalls exist (alternative designs might have unexpected impact)

L4S: opportunity for new/existing improvements

- What other CC improvements can we bring to the Internet together with L4S - DualQ?
- Limited opportunity until tsvwg drafts go for last call

Please evaluate, review and comment

- From the authors perspective, the tsvwg drafts are in good shape

Milestones

ECN Experimentation:

- as early as poss - so experimental work can proceed

L4S arch (INF), L4S ID (EXP), DualQ (EXP)

- should go together
- probably Nov 2017 for earliest WGLC
- allows time for at least one working implementation of the TCP Prague requirements
- deadline to avoid TCP Prague work to spin out of control?

TCPM: Accurate ECN (EXP), DCTCP (INF), Generalized ECN (INF)

- Dependency?

Questions

koen.de_schepper@nokia.com