L4S: Ultra-Low Queuing Delay for All

Low Latency, Low Loss, Scalable throughput

Bob Briscoe

bobbriscoe.net Ltd

Nov 2018

With particular acknowledgement of the work of Koen De Schepper (Nokia Bell Labs), Olga Bondarenko (Simula), Inton Tsang (Nokia) and Greg White (CableLabs) and all the support of CableLabs.

R / T E

Reducing Internet Transport Latency

The authors' contributions are part-funded by the European Community under its Seventh Framework Programme through the Reducing Internet Transport Latency (RITE) project (ICT-317700). The views expressed here are solely those of the authors.

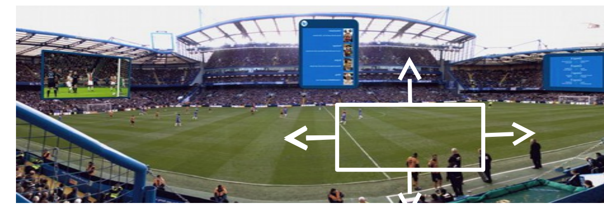
Seventh Framework Programme

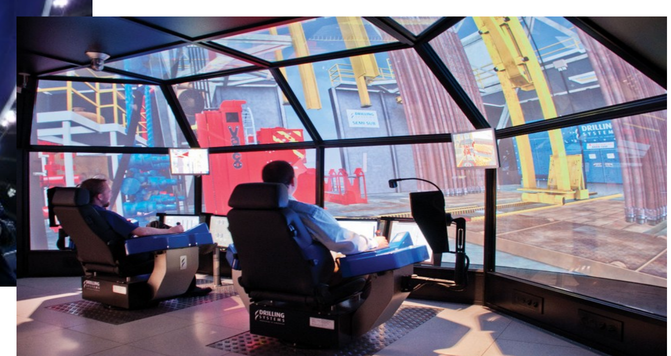
To introduce myself

- Career in BT (1980-2015); always computing AND network
  - …, sys-admin, distributed systems research, Edge Lab Head, Chief Researcher for Network Infrastructure (mostly interface/protocols between hosts and network)
- Standards background – only as necessary (not a standards goer)
  - Ended up as IETF co-ordinator for the BT Group
  - Helped create ETSI NFV, Chaired NFV Security Expert Group
  - Minor interaction with IEEE (and 3GPP) via liaison statements
- Expertise
  - Traffic control, cross-layer
  - Public policy, interactions between IPR / open-source / standards
  - Grasping nettles
- Lately: research consultant
  - Primarily with CableLabs, independent hat on today

application profile is evolving

- increasingly nearly *all* apps require low delay (and often high bit rate too)
  - interactive web, web services
  - voice,
  - conversational video, interactive video, interactive remote presence
  - instant messaging
  - online gaming
  - virtual reality, augmented reality
  - remote desktop, cloud-based apps
  - video assisted remote control of machinery & industrial processes



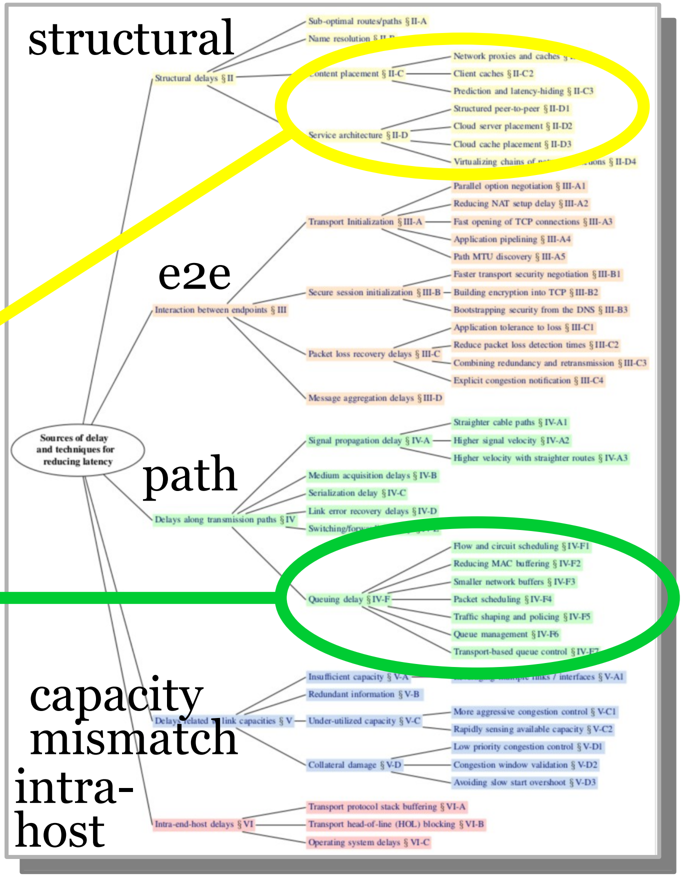


Main contributions to delay

- Delay: multifaceted problem [Briscoe14]

  1) Caches have cut base (speed-of-light) delay, where they can

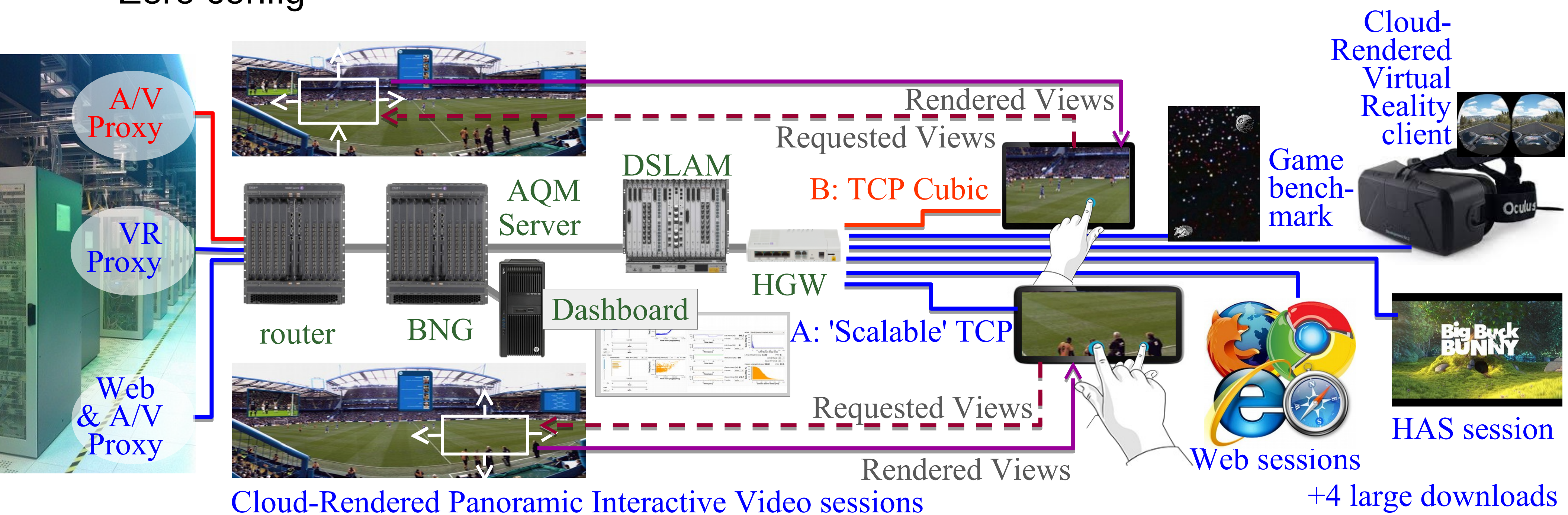
  2) Remaining major component of delay: **queuing**
     - intermittent – solely under load
     - at best, **doubles the base delay;** otherwise under-utilizes capacity



Demo of the L4S vision @MMSys'16

new default service for the Internet

- Multiple demanding applications over the same broadband line, in one FIFO queue
  - Set-up: 40Mb/s downstream over DSL access, 7 ms base round trip time
  - Outcome: per-packet L4S queuing delay: mean ~500μs, 99%-ile ~1000μs zero packet drop, full utilization
- Applications unchanged (update to TCP in OS); coexists with existing TCP traffic
- Zero config



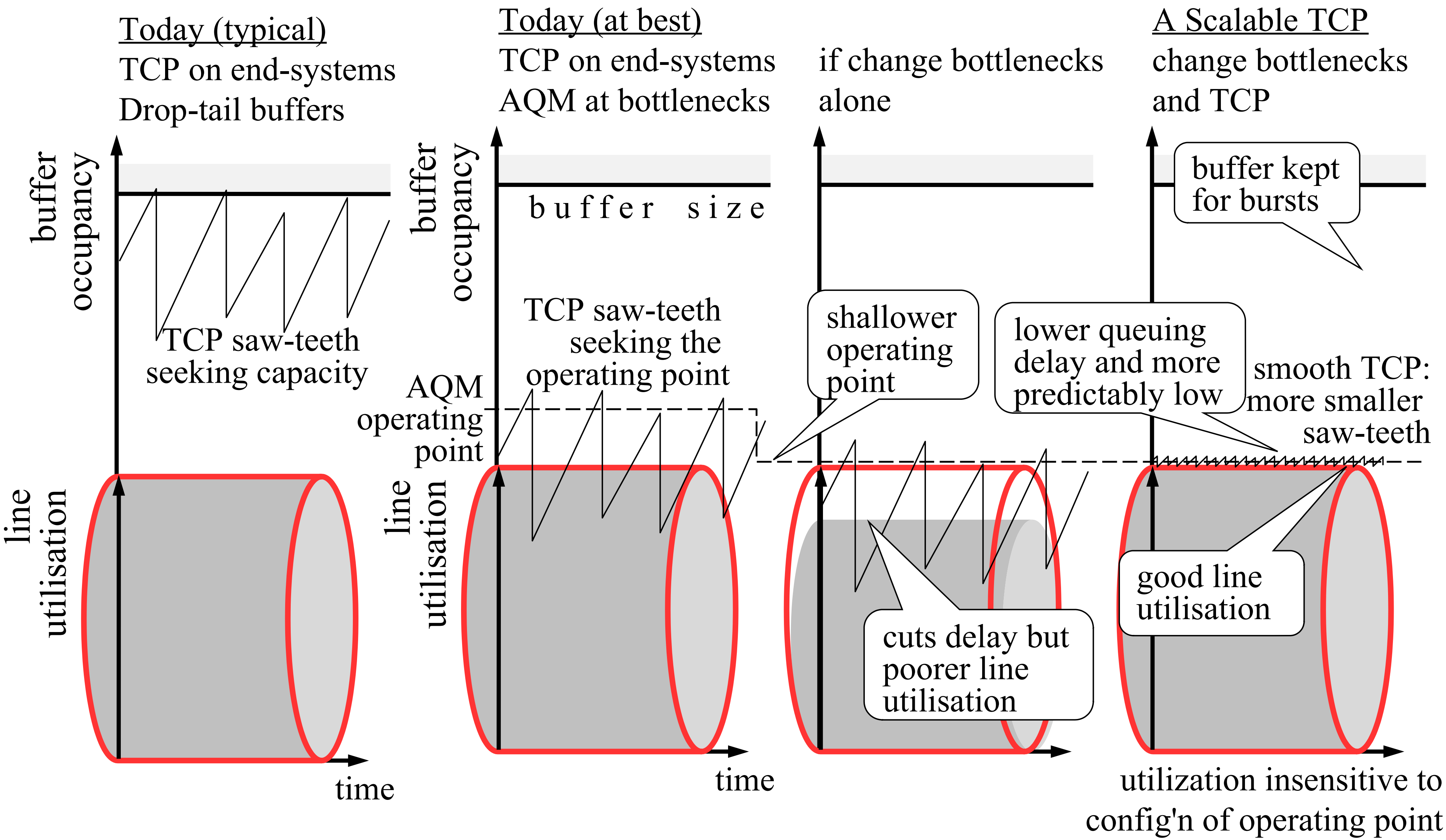
- Video (part): https://riteproject.eu/dctth/#1511dispatchwg

Myths

- you solve queuing delay in the queues
- you have to have low utilization for low delay
- congestion signals are bad

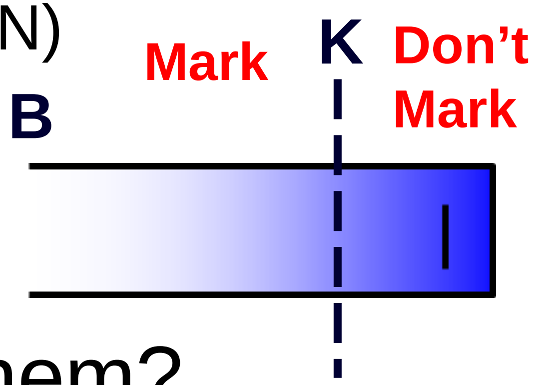
Resolving the dilemma:

Finer saw-teeth of a 'Scalable' TCP (e.g. DCTCP)



We're done, aren't we?

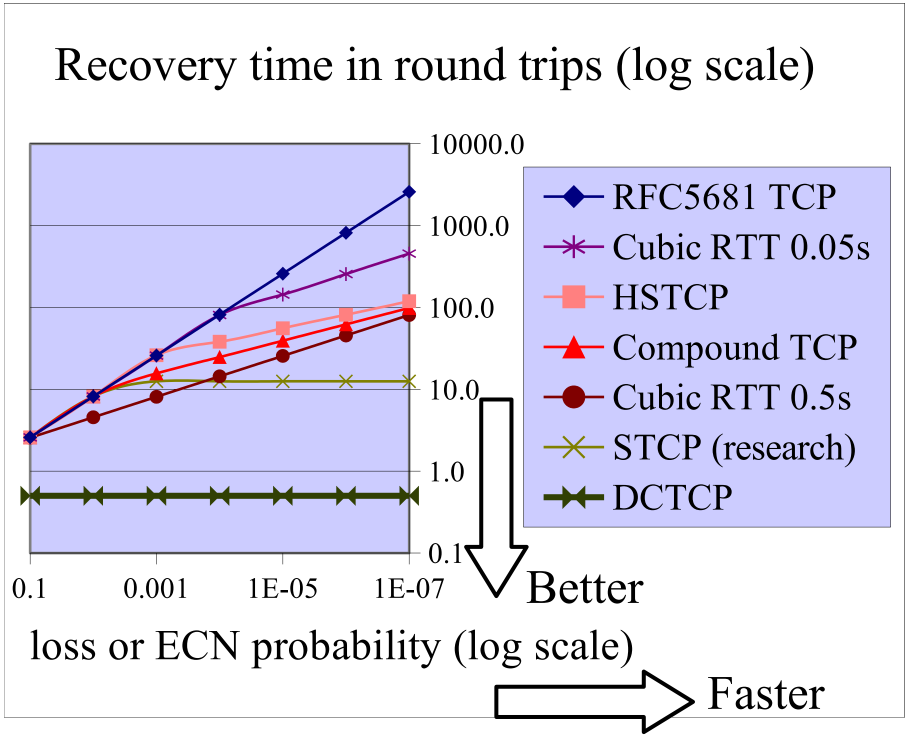
- Hosts
  - DCTCP exists
  - it's in Windows, Linux & FreeBSD
- Switches
  - Need Explicit Congestion Notification (ECN)
  - because drop would be too frequent

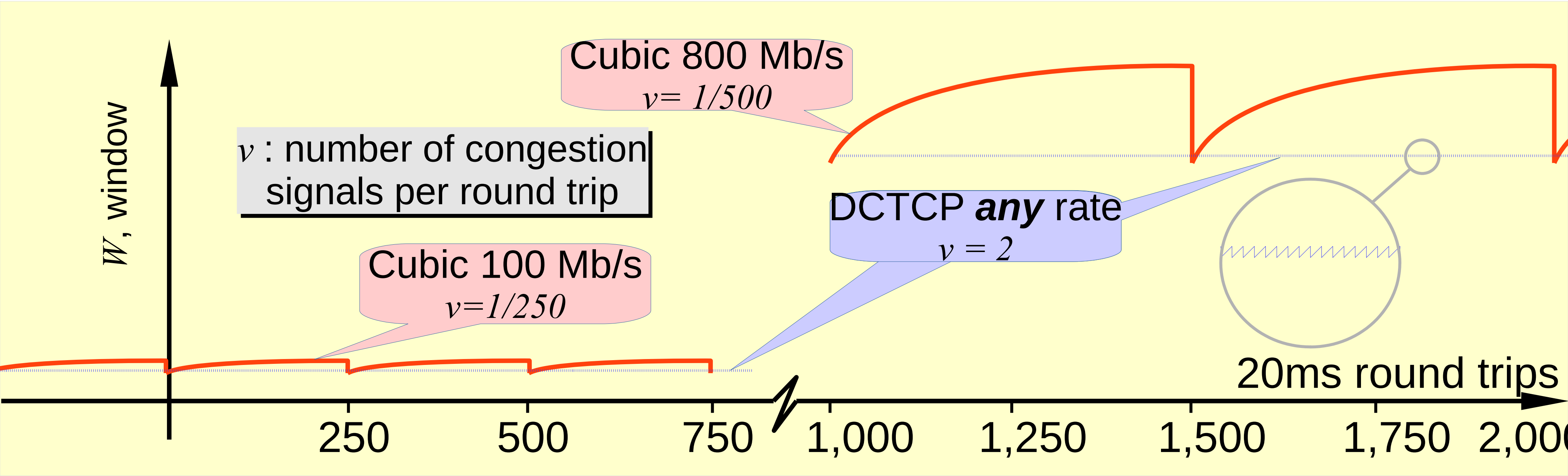


- We've got these. Why not just use them?

Tutorial: sawteeth

- 1988: TCP developed
  - footnote: it's unscalable
- 1990s: Recognized TCP Reno scaling problem
- 2000s: TCP Cubic etc. deployed
  - “less unscalable”
- 2015: DCTCP deployed - scalable
  - only in single admin DCs 'cos does not coexist
- 2020s: Cubic scaling insufficient

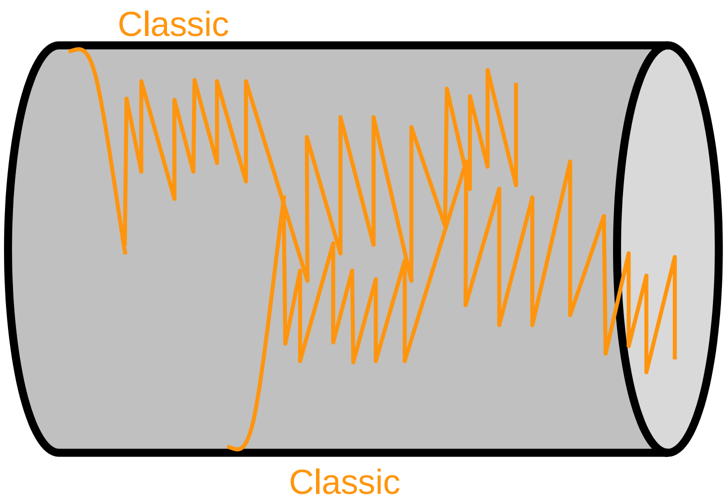


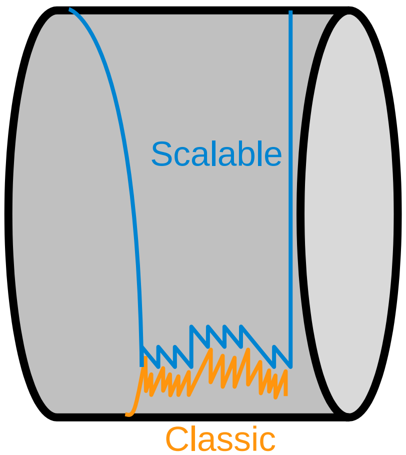


Fine saw-teeth are not fine...

...in cloud DCs, interconnected DCs

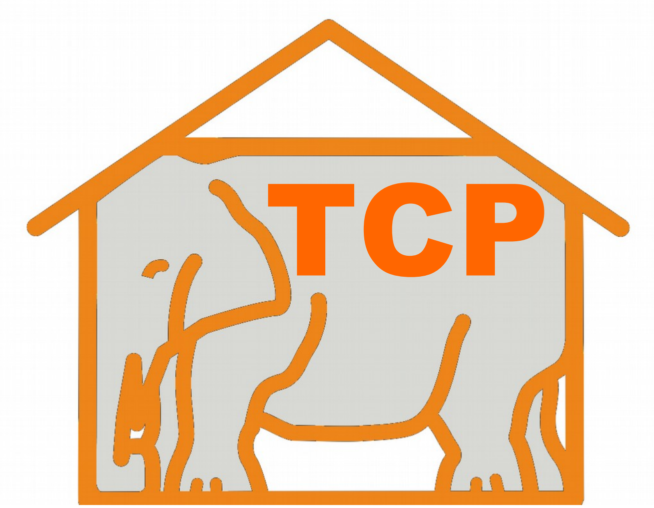
- unless the 'coexistence problem' is solved
  - one 'Scalable' flow with frequent sawteeth
    looks like many 'Classic' flows to a 'Classic' TCP flow
  - so the Classic flow starves itself





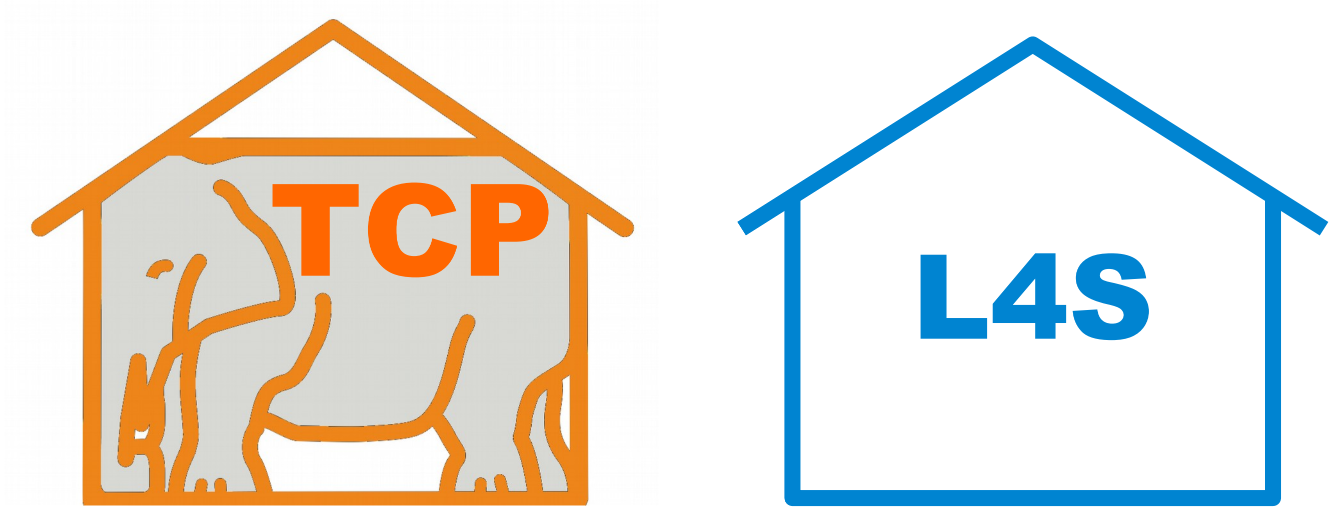
Problem: very high level summary

- Problem: Classic TCP is the elephant in the room
- Solution: build another room without the elephant



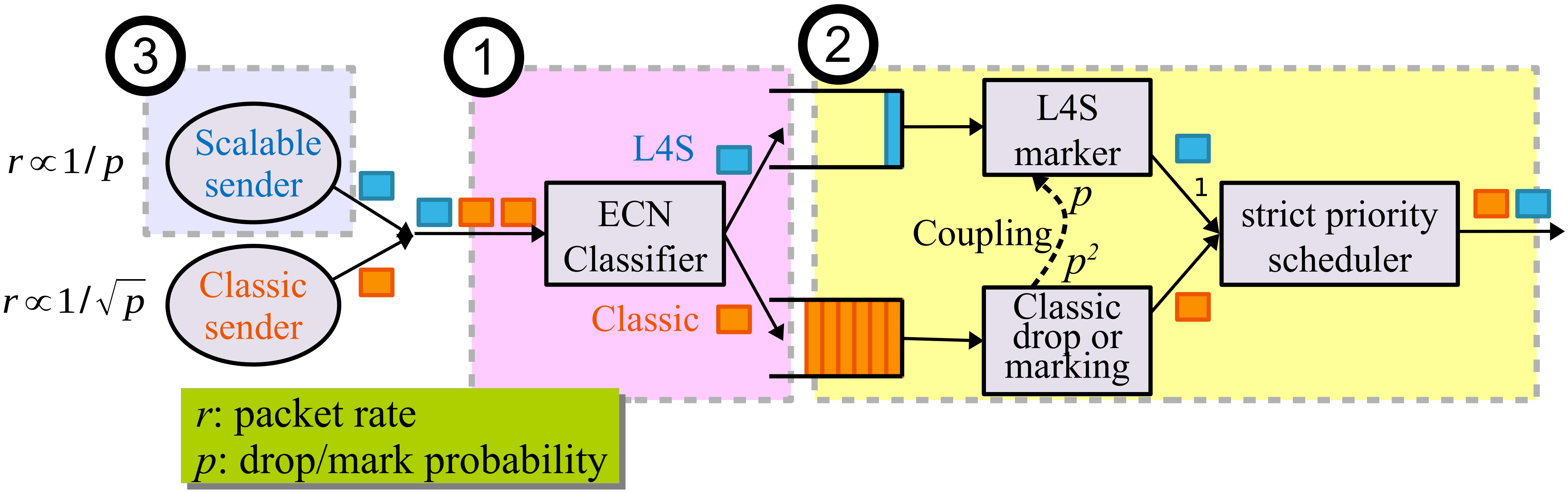
Solution: very high level summary

- Problem: Classic TCP is the elephant in the room
- Solution: build another room without the elephant



Coexistence: Solution

- At bottlenecks...
- DualQ Coupled AQM (1) & (2): a 'semi-permeable membrane' that:
  - isolates latency (separate queues for L4S & Classic)
  - but pools bandwidth (shared by apps/transport, not by network)

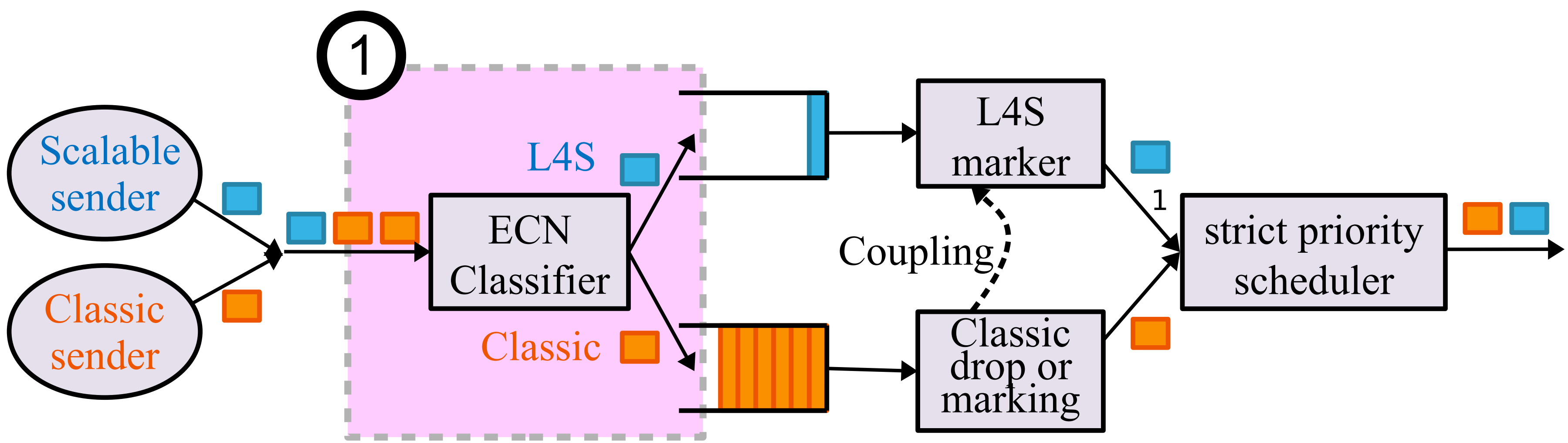


Details: [l4s-arch], [dualq-aqm]

Coexistence: Solution (2)

- Identifier for L4S classifier ① ?
- ECT(1) codepoint in IP header (v4&v6)

Codepoint	ECN bits
Not-ECT	00
ECT(0)	10
ECT(1)	01
CE	11



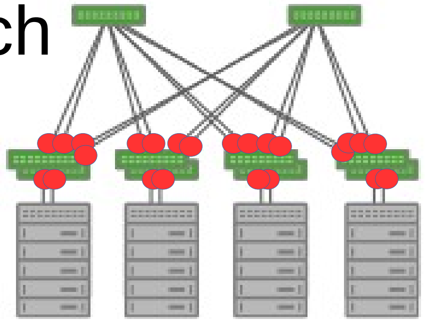
Details: [RFC8311] [ecn-l4s-id]

Other solutions - in context

- Priority classes (Differentiated Services)?
  - only solves latency if low latency traffic is a small proportion of the link (not for *all*)
  - complementary to L4S to schedule bandwidth allocation (where required)
- DCB, 802.1Qau (Congestion Notification), PFC (Priority-based Flow Control)
  - Not applicable to multi-subnet
  - Necessary for sub-RTT traffic flows
  - Complementary to L4S which provides interconnect and interaction with L4, L7 (see later)
- Single Queue Active Queue Management (AQM)
  - a solution 'for all' – promising direction
  - but Classic TCP (literally) remains as the elephant in the room – min queue doubles RTT
- Per-microflow queue and per-queue AQM (per-flow queuing)?
  - isolates each flow from the delay of others, but overkill...
    1. individual app flows not always visible to network (e.g. encrypted aggregates)
    2. computationally expensive
    3. anyway, doesn't protect a flow from the delay it inflicts on *itself*
- BBR (Google research)
  - Attempt to reduce queuing delay without changing network
  - Queuing delay similar to single queue AQM (doubles RTT or more), plus spikes
  - Problems interacting with AQM: toggles between starving others or itself

Deployment scenarios

- Non-blocking core
  - ingress and egress bottleneck would typically give nearly all the benefit
  - e.g. all the outputs of the top-of-rack switch
  - and ingress to inter-DC WAN links



- Blocking core
  - DualQ Coupled AQM is simple
  - not infeasible for DC core switches

L4S maturity status

- IETF: L4S adopted for standardization (experimental status)
  - Architecture, Identifiers and Network AQM: approaching WG last-call Dec-2018 or Jan-2019
  - Host Congestion Control: DCTCP [RFC8257] + “TCP Prague Requirements” [ecn-l4s-id]
    - Some adopted for standardization, others still IRTF (research)
- Numerous companies involved
  - equipment vendors
  - operators
  - OS developers
  - hardware developers
- Mostly access network bottleneck scenarios
  - DSL, DOCSIS, LTE
- One merchant Si implementation of DualQ Coupled AQM
  - for core, metro, backhaul SoC solutions in switches
  - 'in its birth throes' 'will take some time for testing' (Nov-18)

L4S

where IETF / IEEE joint work is needed

Engineering

- DualQ Coupled AQM algorithms for switches
  - two simple examples in [dualq-aqm]: DualPI2 & Curvy RED
  - instantaneous queue (no filtering/smoothing)
    - unlike Classic AQMs (e.g. RED)
  - must measure queue delay in time units
    - variable drain rates between dualQs (needed for priority Qs anyway)
  - virtual queue [RFC5670] [HULL]
    - near-zero queue
    - ECN marks as if link is slightly lower capacity
- Simplifying 802.1p / Diffserv QoS arrangements
  - L4S for latency, 802.1p or RFC2474 for bandwidth [l4s-diffserv]

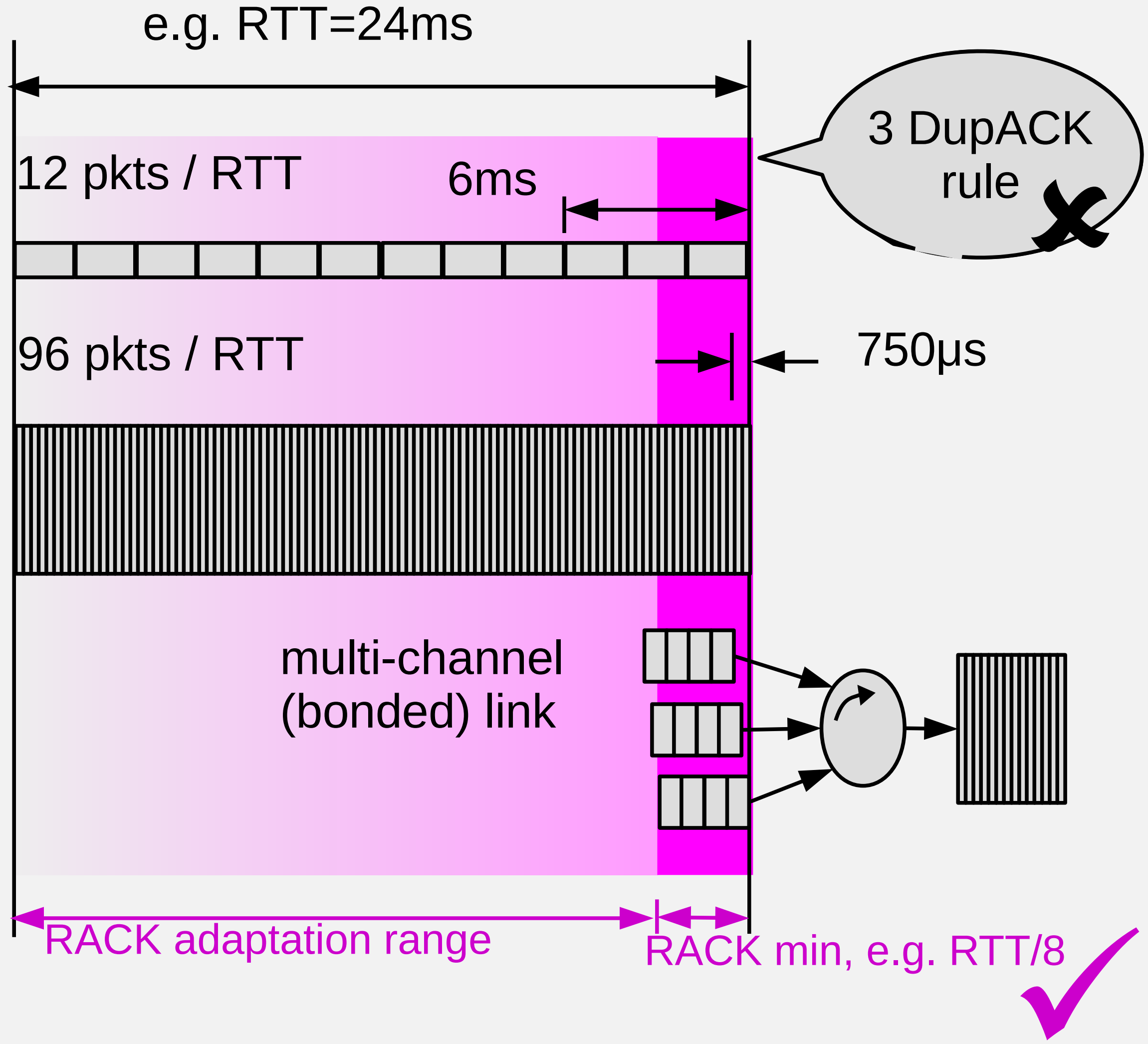
research / open issues / opportunities

- TCP Prague
  - Safety & performance enhancements to DCTCP
  - Sub-single-packet window
  - RTT-independence
  - Getting up to Speed fast with no overshoot [paced-chirping]
- Removal of L4 edge gateways
  - No rate mismatch at DC border
- Relaxation of Ordering Requirements
  - All L4S sources required to use RACK
- Queue Protection algorithms (policing)
  - At ToR or hypervisor [conex-dc-policing]
- integration of L2 (sub-RTT DCB) and L3 (super-RTT L4S) Congestion Control
  - credit-based remote queue protection from edge [conex-dc-policing]
  - potential for single FIFO as common storage and data queue

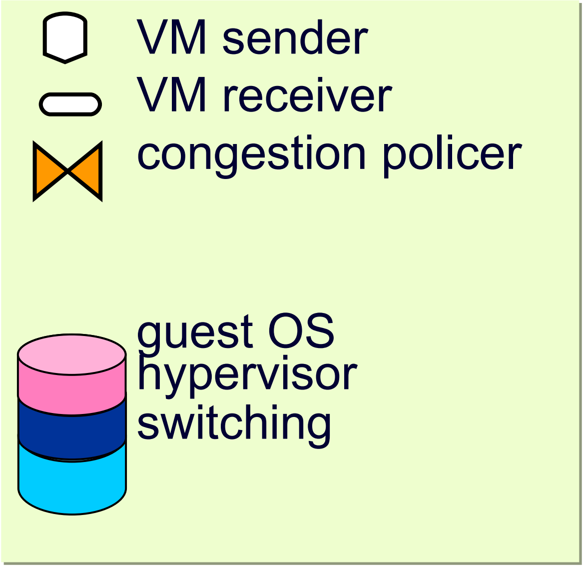
Benefits of universal RACK to links (1/2)

- 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

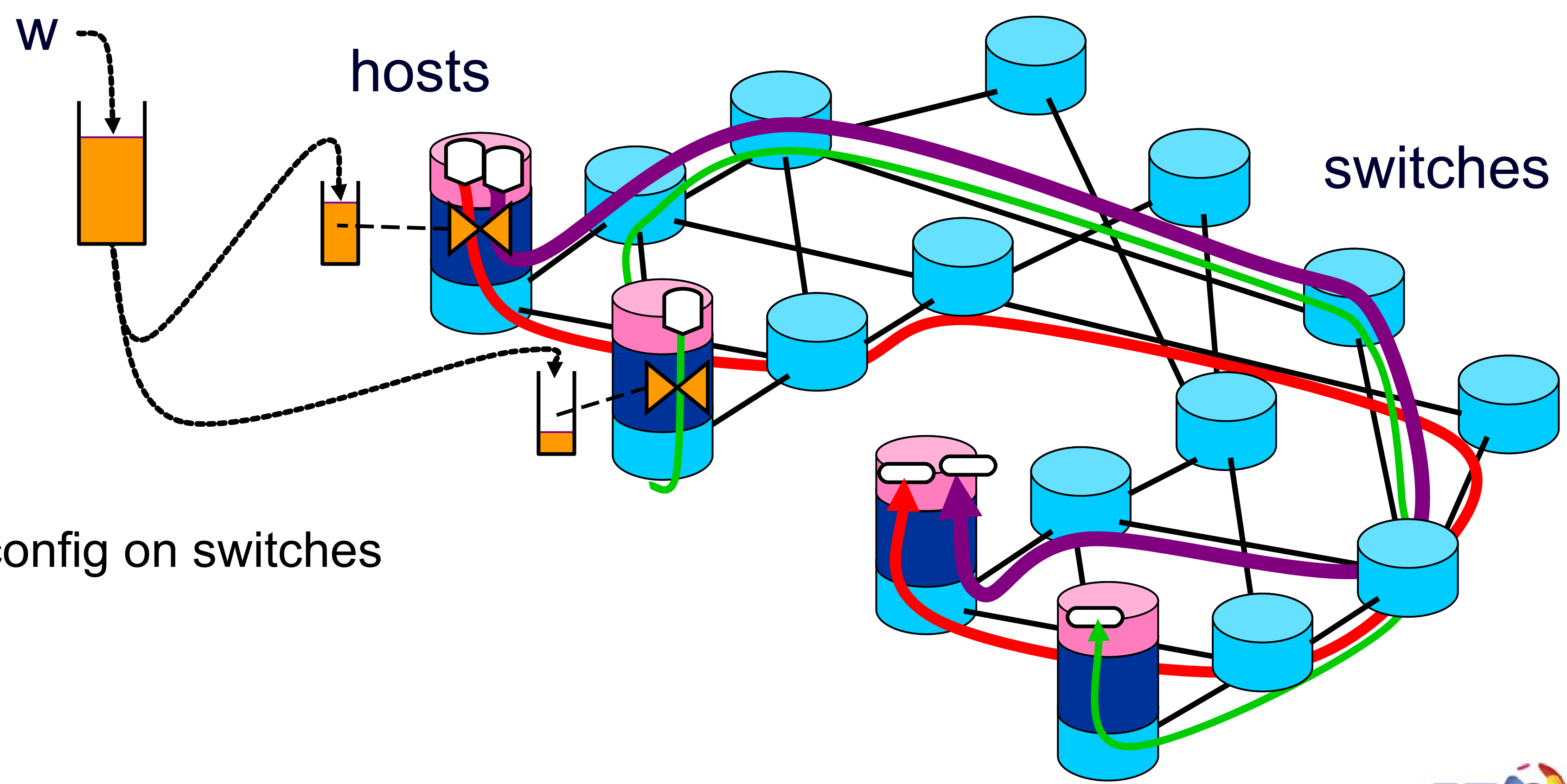
      (given min practical e2e RTT remains fairly constant)



edge bottlenecks by capacity design



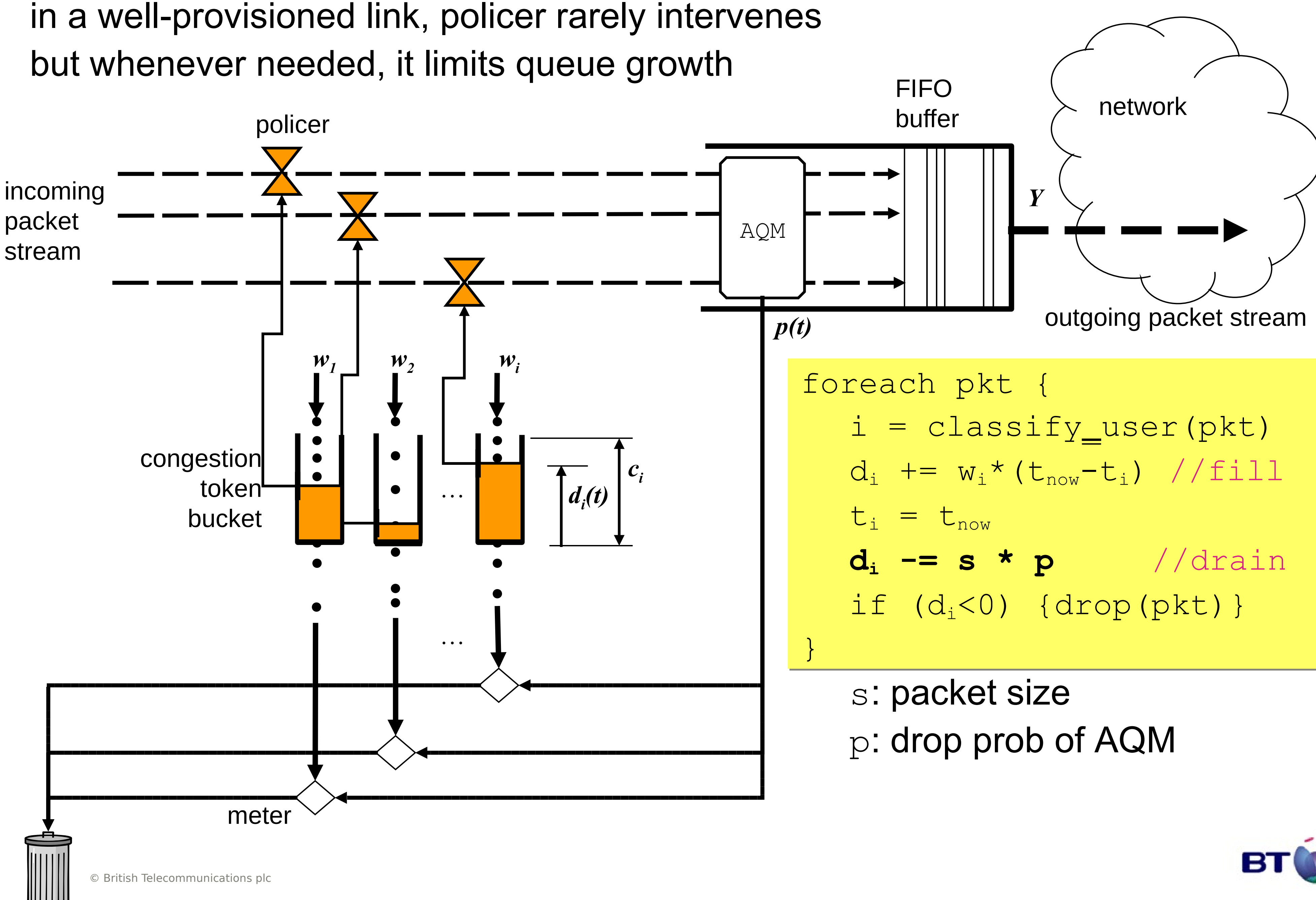
- Edge policing like Diffserv
  - but congestion policing (per guest)
- isolation within FIFO queue



- no config on switches

bottleneck congestion policer

- in a well-provisioned link, policer rarely intervenes
- but whenever needed, it limits queue growth




```
foreach pkt {
   i = classify_user(pkt)
   di += wi*(tnow-ti)  //fill
   ti = tnow
   di -= s * p        //drain
   if (di<0) {drop(pkt)}
}
```

`s`: packet size

`p`: drop prob of AQM

L4S: more info

- Landing Page: https://riteproject.eu/dctth/ Search “DCttH”

- [Briscoe14] Briscoe, B., Brunstrom, A., Petlund, A., Hayes, D., Ros, D., Tsang, I.-J., Gjessing, S., Fairhurst, G., Griwodz, C. & Welzl, M., "Reducing Internet Latency: A Survey of Techniques and their Merits," IEEE Communications Surveys & Tutorials 16(4) IEEE (Nov 2014)
- [RFC7560] Kühlewind, M., Scheffenegger, R. & Briscoe, B. “Problem Statement and Requirements for Increased Accuracy in Explicit Congestion Notification (ECN) Feedback” IETF RFC7560 (2015)
- [Briscoe17] Briscoe, B., Scheffenegger, R. & Kühlewind, M., "More Accurate ECN Feedback in TCP," IETF Internet Draft draft-ietf-tcpm-accurate-ecn-07 (Nov 2018) (Work in Progress)
- [l4s-id] De Schepper, K., Briscoe (Ed.), B. & Tsang, I.-J., "Identifying Modified Explicit Congestion Notification (ECN) Semantics for Ultra-Low Queuing Delay," Internet Engineering Task Force Internet Draft draft-ietf-tsvwg-ecn-l4s-id-05 (Nov 2018) (Work in Progress)
- [dualq-aqm] De Schepper, K., Briscoe (Ed.), B., Bondarenko, O. & Tsang, I.-J., "DualQ Coupled AQM for Low Latency, Low Loss and Scalable Throughput," Internet Engineering Task Force Internet Draft draft-ietf-tsvwg-aqm-dualq-coupled-08 (Nov 2018) (Work in Progress)
- [l4s-arch] Briscoe (Ed.), B., De Schepper, K. & Bagnulo, M., "Low Latency, Low Loss, Scalable Throughput (L4S) Internet Service: Architecture," Internet Engineering Task Force Internet Draft draft-ietf-tsvwg-l4s-arch-03 (Nov 2018) (Work in Progress)
- [l4s-diffserv] Briscoe, B., "Interactions between L4S and Diffserv," Internet Engineering Task Force Internet Draft draft-briscoe-tsvwg-l4s-diffserv-02 (Nov 2018) (Work in Progress)
- [conex-dc-policing] Briscoe, B. & Sridharan, M., "Network Performance Isolation in Data Centres using Congestion Policing," Internet Engineering Task Force Internet Draft draft-briscoe-conex-data-centre-02 (February 2014) (Work in progress)

- [PI2] De Schepper, K., Bondarenko, O., Tsang, I.-J. & Briscoe, B., "PI^2 : A Linearized AQM for both Classic and Scalable TCP," In: Proc. ACM CoNEXT 2016 pp.105-119 ACM (December 2016)
- [DCttH] De Schepper, K., Bondarenko, O., Tsang, I.-J. & Briscoe, B., "`Data Centre to the Home': Deployable Ultra-Low Queuing Delay for All," (January 2017) (Under Submission)
- [HULL] Alizadeh, M., Kabbani, A., Edsall, T., Prabhakar, B., Vahdat, A. & Yasuda, M., "Less Is More: Trading a Little Bandwidth for Ultra-Low Latency in the Data Center," In: Proc. USENIX Symposium on Networked Systems Design and Implementation (NSDI'12) (April 2012)
- [paced-chirping] Misund, J & Briscoe, B. “Flow-start: Faster and Less Overshoot with Paced Chirping” IRTF Internet Congestion Control Research Group (July 2018)

Conclusions

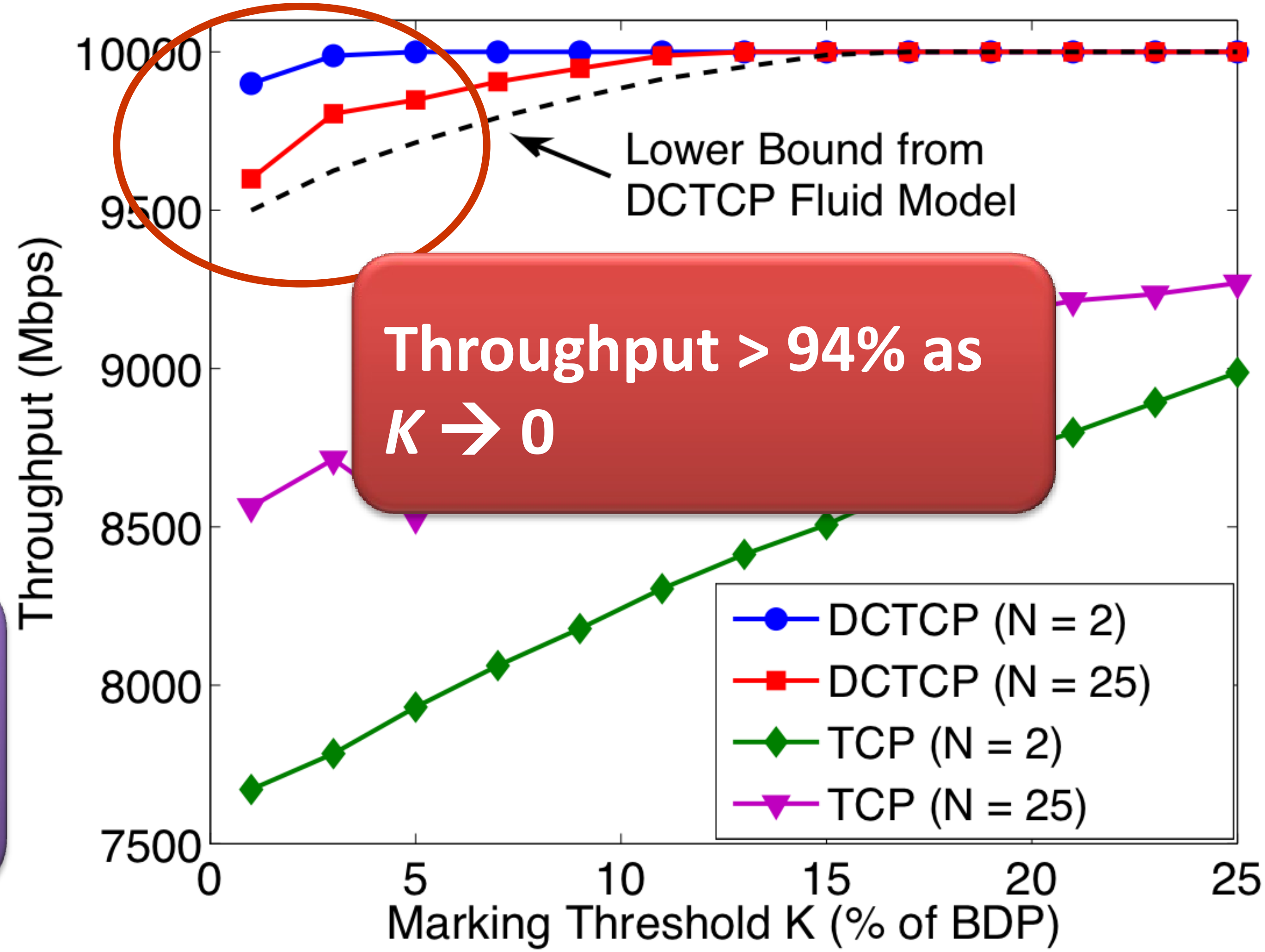
- Enables previously infeasible interactive apps
- Technical problem: 'Classic' TCP
- Technical solution:
    - "Scalable" TCP with L4S ECN codepoint
    - Incremental deployment via DualQ Coupled AQM

- Low Latency for *all* Traffic
    - the classic queue is for legacy, not for life
    - leaves only bandwidth to manage

Q&A

large saw teeth can ruin the quality of your experience

DCTCP Throughput-Latency Tradeoff

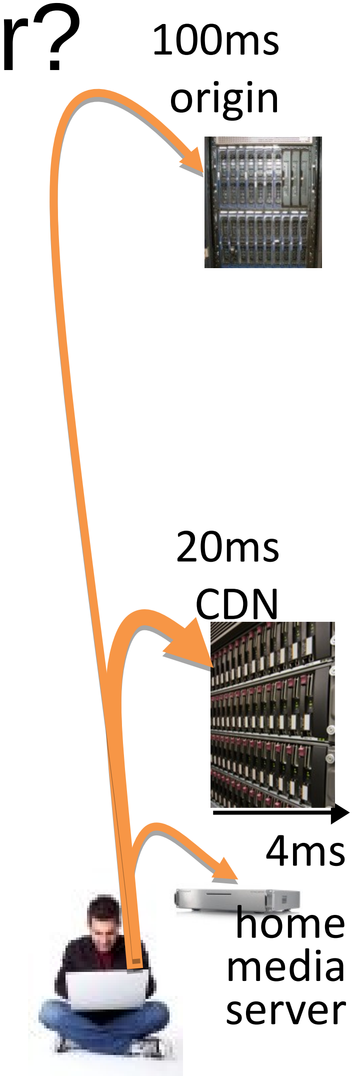


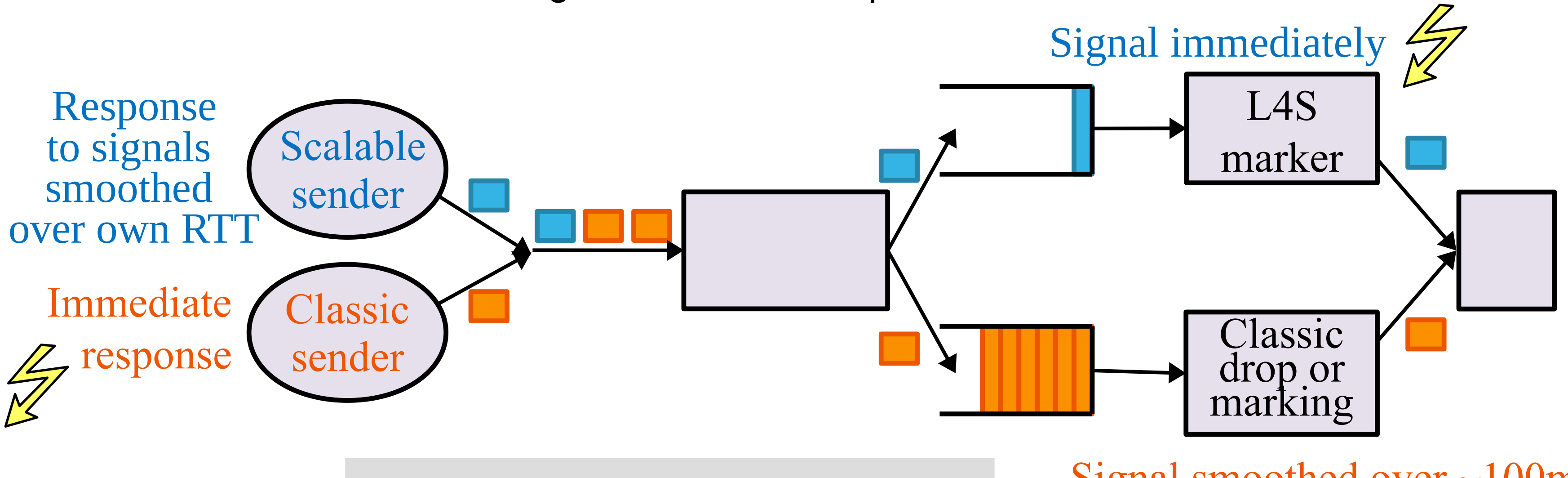
For TCP:
Throughput → 75%

Parameters:
- link capacity = 10Gbps
- RTT = 480μs
- smoothing constant (at source), g = 0.05.

Why is performance so much better?
Immediate signalling

- Today's AQMs defer drop for 1 worst-case RTT
  1)to allow time for a worst-case RTT response
  because: the network doesn't know each packet's RTT
  2)to avoid drop unless the queue proves persistent
  because: drop is an impairment as well as a signal
- Using ECN for L4S makes it feasible to signal immediately
  - because ECN is a signal but not an impairment





Problem with the Classic approach:
a flow with RTT=5ms
gets no signal for 20 round-trips