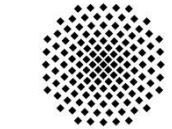
Immediate ECN

Mirja Kuhlewind, David Wagner,
Juan Manuel Reyes Espinosa, Stuttgart Uni

Bob Briscoe, BT

IETF-88 TSVAREA
Nov 2013



Bob Briscoe was part-funded by the European Community
under its Seventh Framework Programme through the
Reducing Internet Transport Latency (RITE) project ICT-317700

summary & context

- Promising early results towards the aim of:
  - the *predictably* low queuing delay of DCTCP*
  - deployable on the public Internet, with existing hardware
  - zero config or config-insensitive

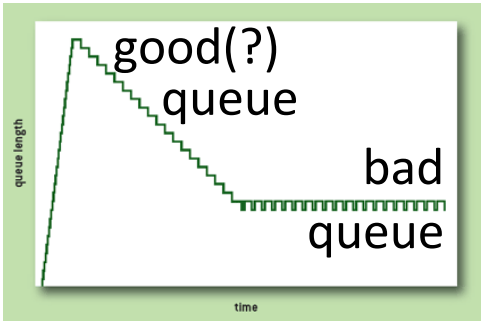
- At IAB rmcat workshop, we foresaw a need to address:

problem	IETF wg formed	this proposal
real-time media congestion avoidance	rmcat	-
prevent TCP bloating queues	aqm	✓
make TCP smoother	-	✓

- Why bring such early results to the IETF?
  - to test the water on a redefinition of ECN
    - to foster ECN deployment through more significant benefits

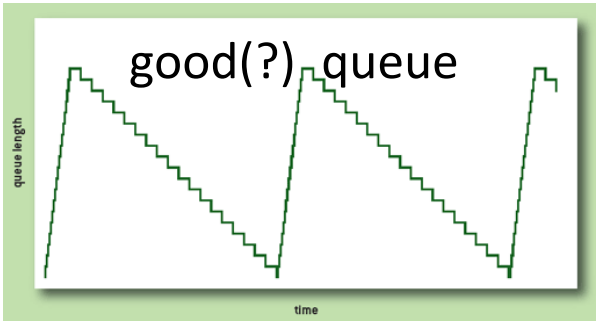
---

* DCTCP: data centre TCP



problem

AQM dynamics



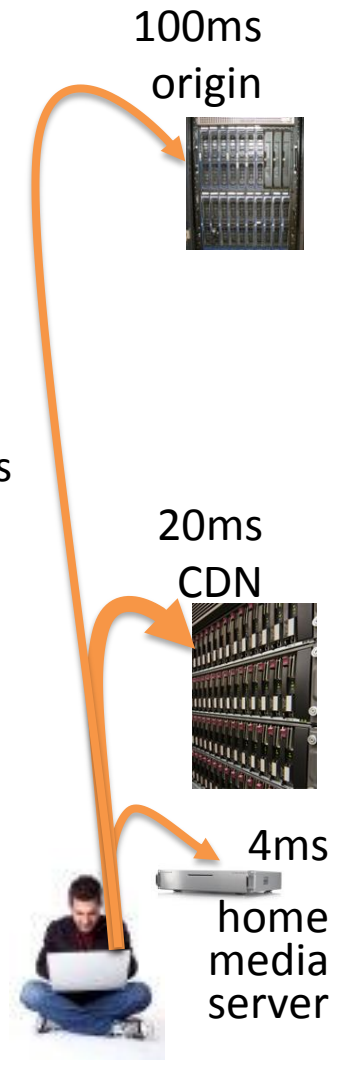
- buffer's job: absorb bursts that dissipate by themselves
- all AQMs defer dropping for c.1 worst-case RTT

RED	w_q	512 pkt*
PIE	max_burst	100ms
CoDel	interval	100ms

- for a flow with RTT of 20ms or 4ms
  - e.g. content distribution network or home media server
  - these AQMs suppress any signal for 5 or 25 of the flow's own RTTs

- CoDel, PIE: auto-tune for varying line-rate
- also need to auto-tune for varying RTT

- it's not 'good' to hold back from signalling for 100ms
  it's just necessary if the alternative is *drop*

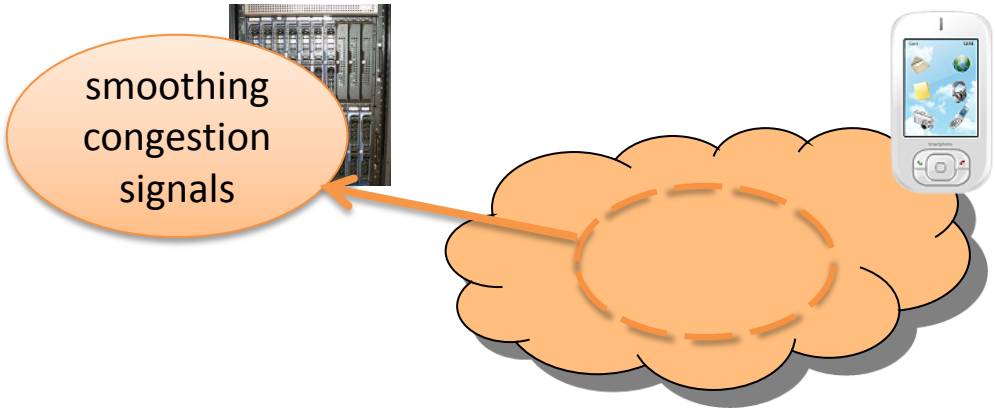


* @10Mb/s & 700B/pkt, 512pkt → 3s for moving ave of queue to reach 63% of inst queue, but not comparable with PIE & CoDel delays, which are absolute

AQM dynamics

solution

[from DCTCP]

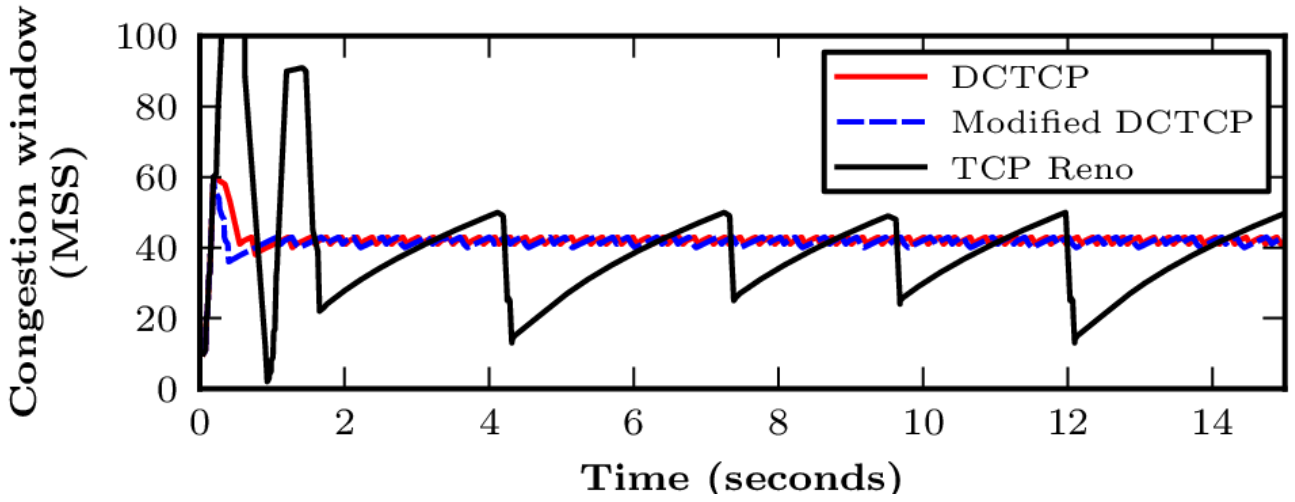


- For ECN-capable packets
  - shift the job of smoothing congestion signals from network to host
    - the network signals ECN with no smoothing delay
    - the transport can hide bursts of ECN signals from itself

- the transport knows
  - whether it's TCP or RTP etc
  - whether its in cong avoidance or slow-start
  - and it knows its own RTT
- so it can decide
  - whether to respond immediately
  - or to smooth the signals
  - and, if so, over what time
- then short RTT flows can smooth the signals themselves delayed only by their *own* RTT
  - so they can fill troughs and absorb peaks that longer RTT flows cannot

aims: real performance gain
(and avoid RTT-sensitive config)

- DCTCP on host uses immediate ECN
- DCTCP only smooths the ECN signals while in congestion avoidance
- DCTCP in slow-start responds without smoothing, immediately reducing overshoot

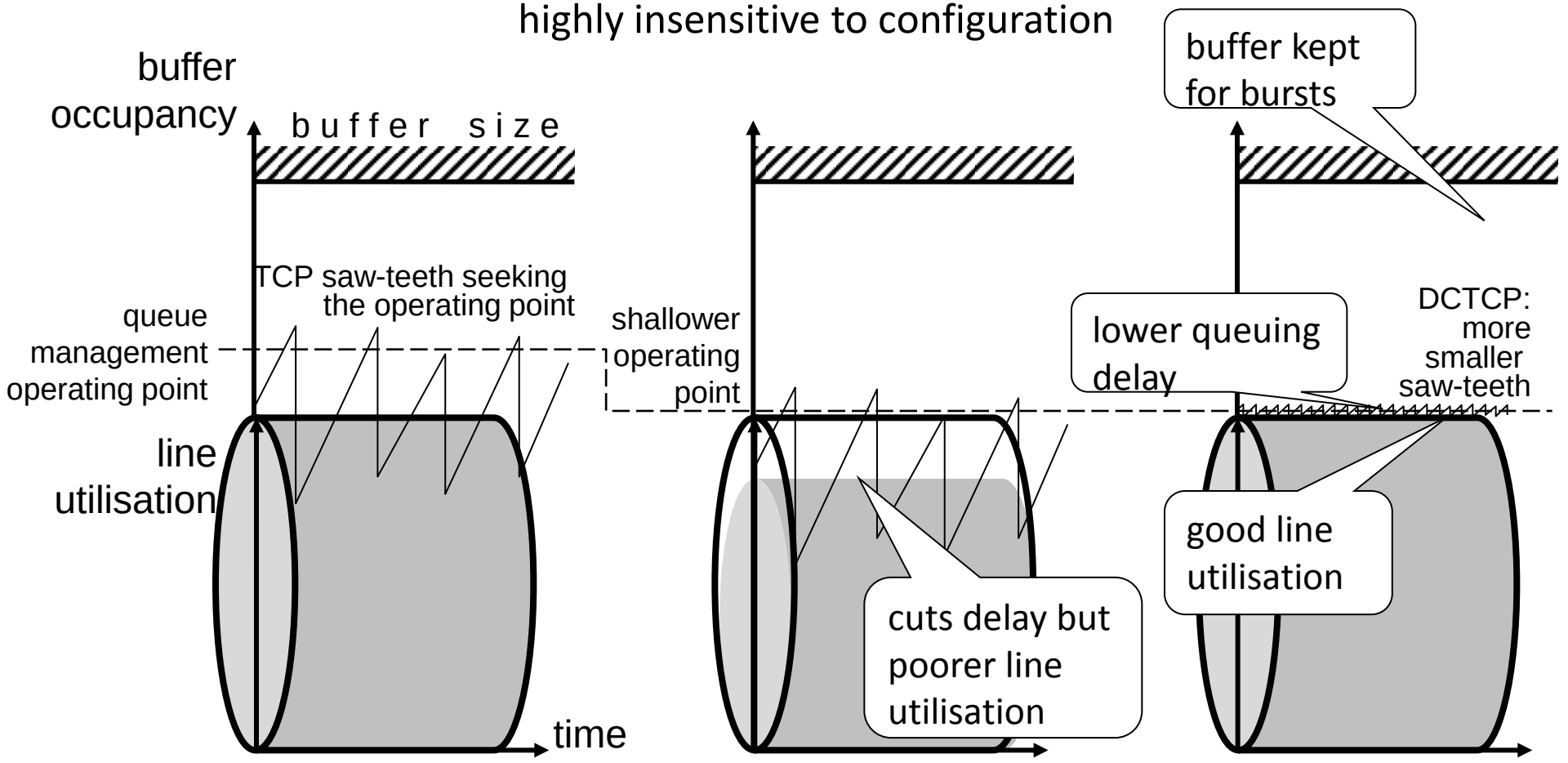


* Modified DCTCP is only shown separate from DCTCP, because we improved the original DCTCP slightly

Data Centre TCP (DCTCP)

high utilisation in steady state still leaves room for bursts

highly insensitive to configuration



Today (at best)
TCP on end-systems
RED in queues

if solely change queues

change queues and end-systems

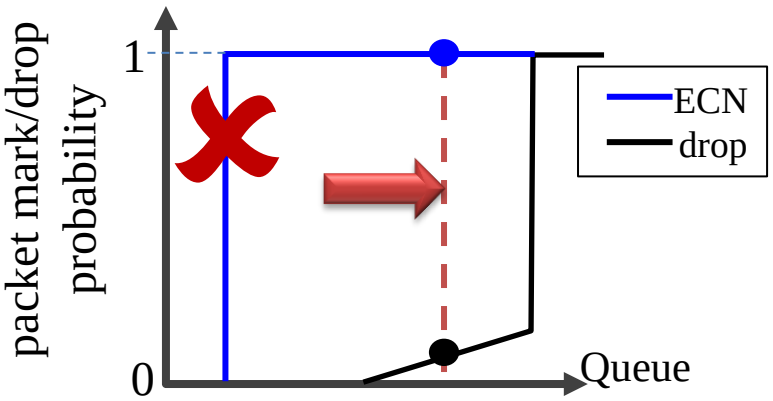
aim: real performance gain

- classic ECN
  - cannot justify deployment pain
    for a questionable performance gain
- immediate ECN
  - addresses *predictability* and low queuing delay
  - including self-delay for short flows
  - avoiding RTT-sensitive config

problem II

co-existence of DCTCP with existing Internet traffic

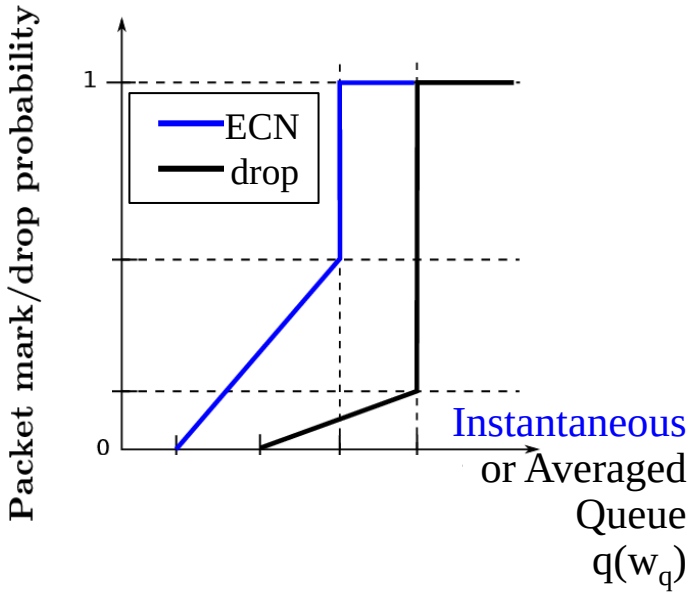
- data centre TCP was so-called only because it couldn't co-exist with Internet traffic
- can't have a low delay threshold for ECN and a deep threshold for drop in one FIFO queue
- drop traffic would push the queue to its own balance point
- causing 100% marking of ECN packets



- then ECN traffic would starve itself

co-existence solution
can use existing network hardware

- use weighted RED (WRED) implementation
- in an unusual configuration
  - one FIFO queue with two instances of RED algo
    - smoothed queue for drop (EWMA-constant = 9 say)*
    - current queue for ECN (EWMA-constant = 0)
- as share of DCTCP grows
  - more insensitive to config



* if exponential-weighting-constant = B,
then RED smooths the queue over 2^B packets
if B = 9, RED smooths over 2^9 = 512 packets
if B = 0, RED smooths over 2^0 = 1 packet (i.e. it doesn't smooth)

a similar coexistence approach should be applicable to other AQMs

- ultimately, want to auto-tune against line-rate *and* RTT
  - use a modern AQM that uses queuing delay as its metric
  - *and* separate drop and ECN algos

AQM	smoothing parameter	non-ECN packets	ECN packets
ARED	ewma-const	9	0
PIE	max_burst	100ms	0
CoDel	interval	100ms	0

- message for implementers in silicon
  - ensure parameters can be configured separately for ECN

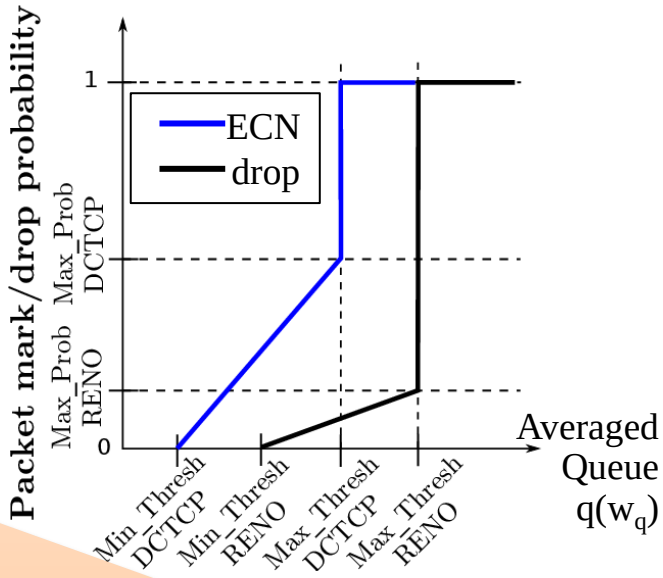
co-existence

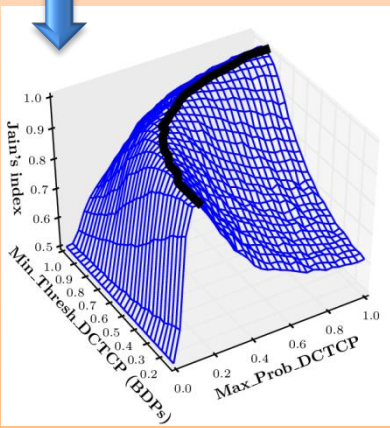
results of ‘gating tests’

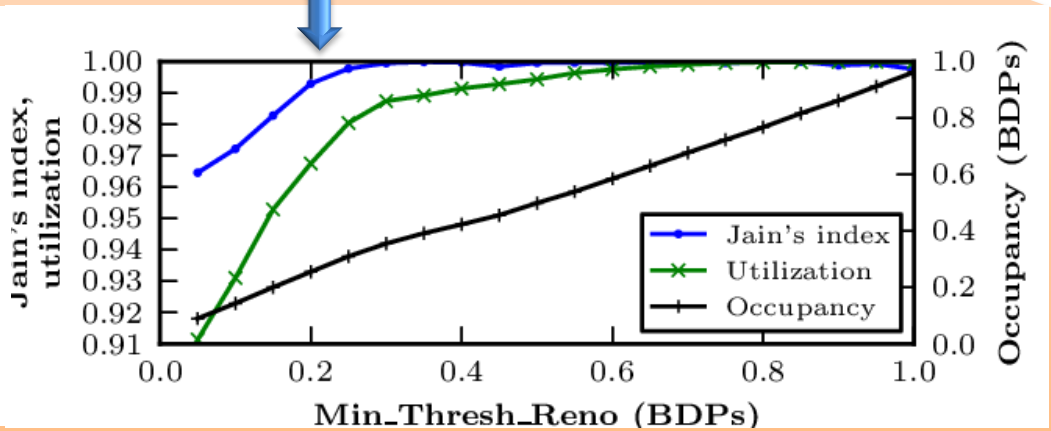
- explored large part of the much larger parameter space
  - implemented in Linux 3.2.18; simulated in IKR simlib
  - ‘gating tests’: long-running flows only
  - paper under submission, available on request

- robust against starvation

- formula to derive ECN config from drop config to maintain rate fairness
- can then find sweet spot for the drop config

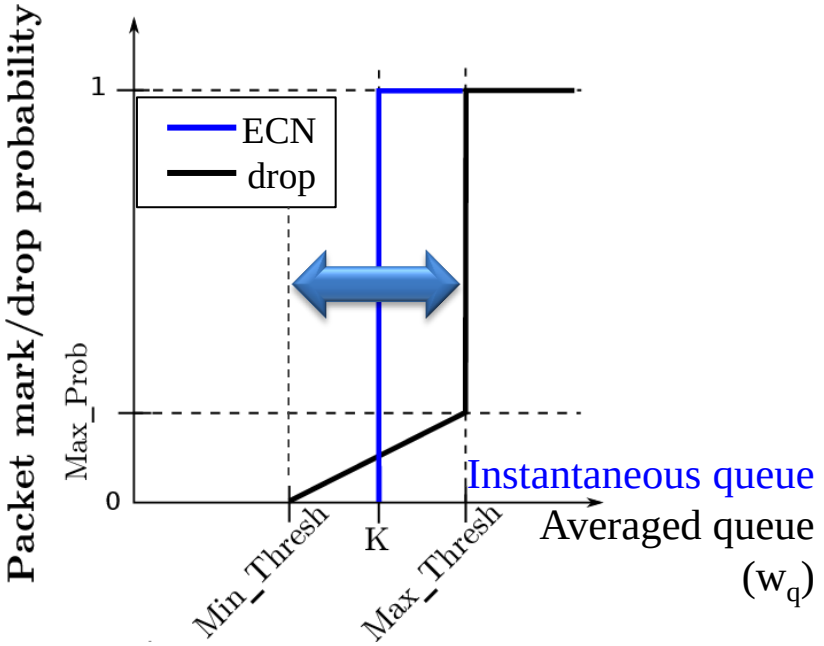


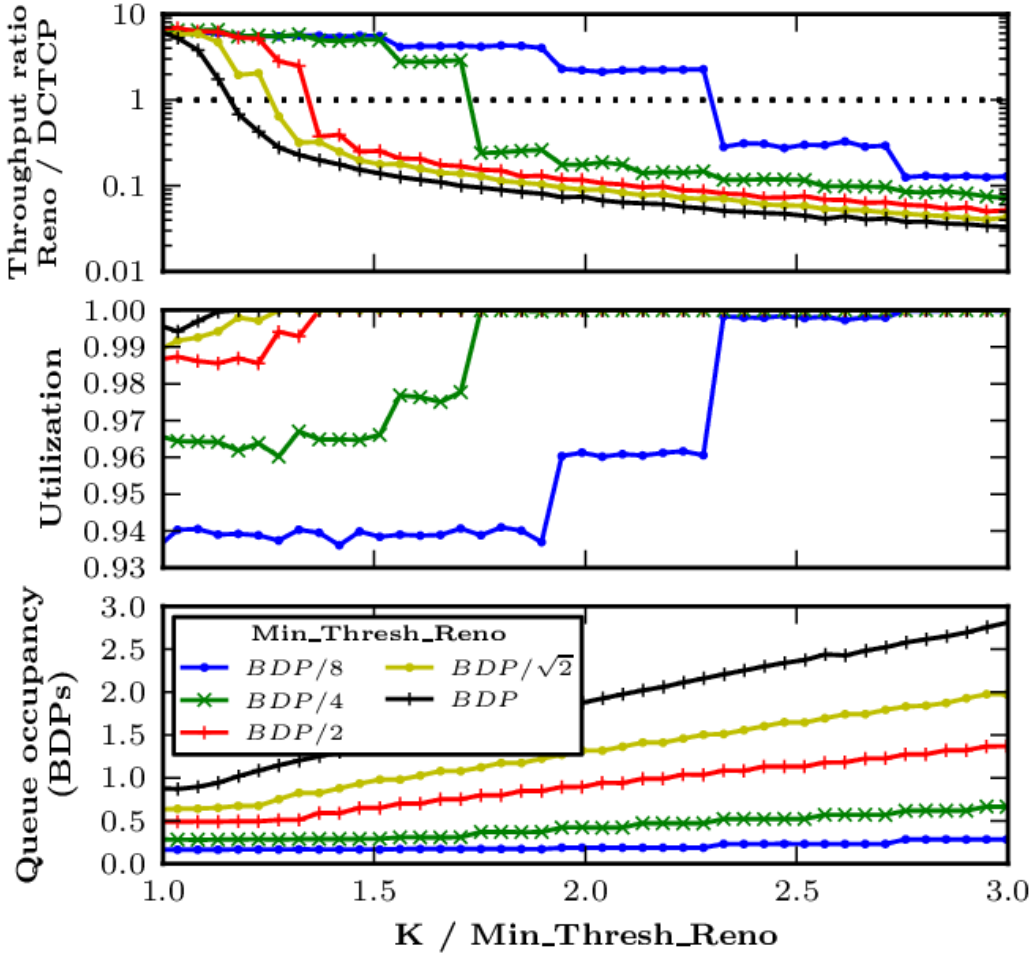




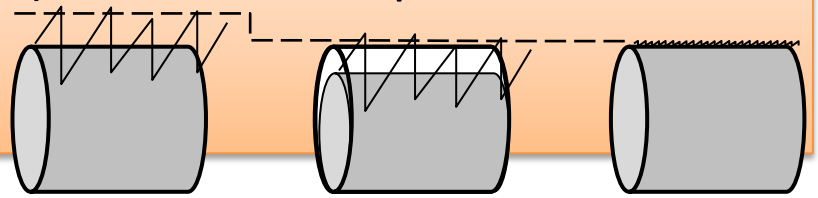
a sample of the

results so far





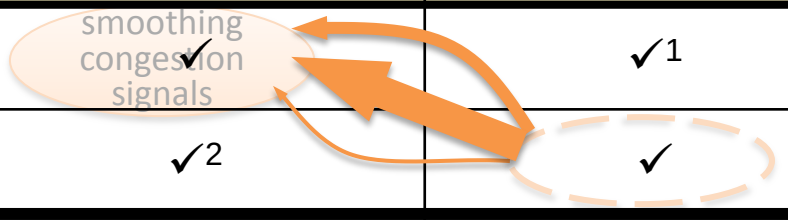
- early deployment, when traffic mostly drop-based
  have to set drop (and therefore ECN) threshold deep
- as more flows shift to DCTCP,
  can set both thresholds shallower



problem III
incremental deployment
interop between classic and immediate ECN

- ECN widely implemented on hosts:
  - on by default at TCP servers
  - off by default at TCP clients
- turn clients on by default when deploy:
  - accurate ECN feedback & ECN fall-back

buffers \ host	small smoothed responses to each ECN	one big instant response to ECN per RTT
immediate ECN	✓	✓[1]
smoothed ECN	✓[2]	✓



[1] don't get full gain in latency until host upgrades as well

[2] doubly delayed response to congestion

these two ticks are based on conjecture, not experimental evidence (yet)

cross-layer / cross-wg impact on IETF

	component	IETF wg	document
1	redefine meaning of ECN CE	tsvwg	Expt update to RFC3168
2	specify ECN behaviour in AQM algos	aqm	CoDel, PIE, (RED++?)
3	specify change to TCP feedback	tcpm	draft-ietf-accurate-ecn-reqs
4	specify change to TCP sender algo	tcpm	Expt update to RFC5861

1. RFC 3168 may not need to be updated (see spare slide)
2. urgent given pace of AQM development
3. wire protocol: the main standards track change
4. algorithm experimentation expected

concluding messages

- research in progress
- promise of *predictably* low delay during dynamics
  - an unnecessary queue is not a 'good' queue
- adds RTT auto-tuning to AQM
  - by shifting smoothing from network to host
- can use existing network hardware
- if you're implementing a new AQM
  - at least ensure parameters can be configured separately for ECN
- question: if subsequent experiments are as promising as these, would there be an appetite in the transport area to tweak the meaning of ECN?

Immediate ECN

Q&A

spare slides

which codepoint for immediate ECN?

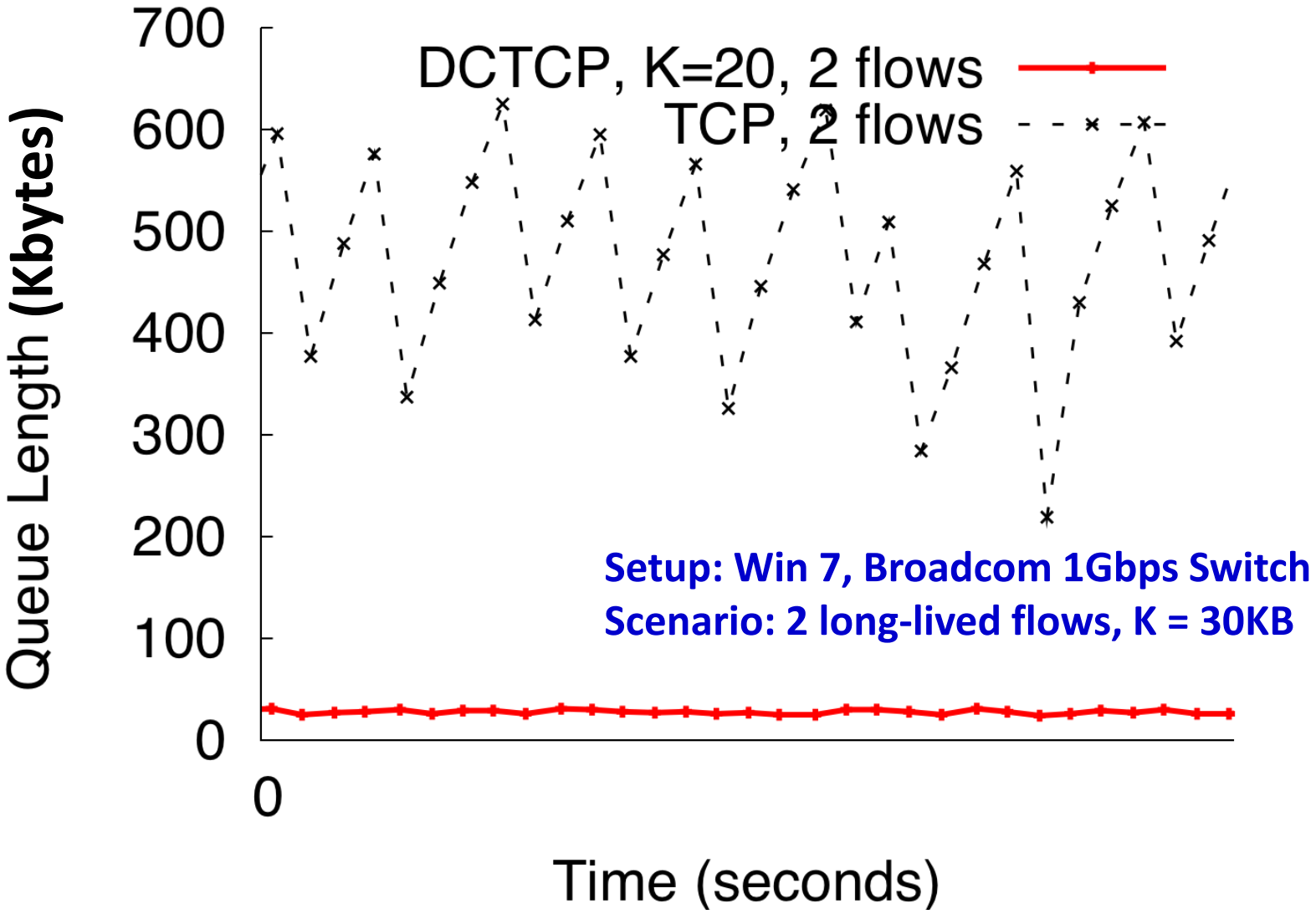
- To use CE for immediate ECN,
  may not need to update RFC3168 (Addition of ECN to IP):

  ...if the ECT codepoint is set in that packet's IP header ... then instead of dropping the packet, the router MAY instead set the CE codepoint in the IP header.

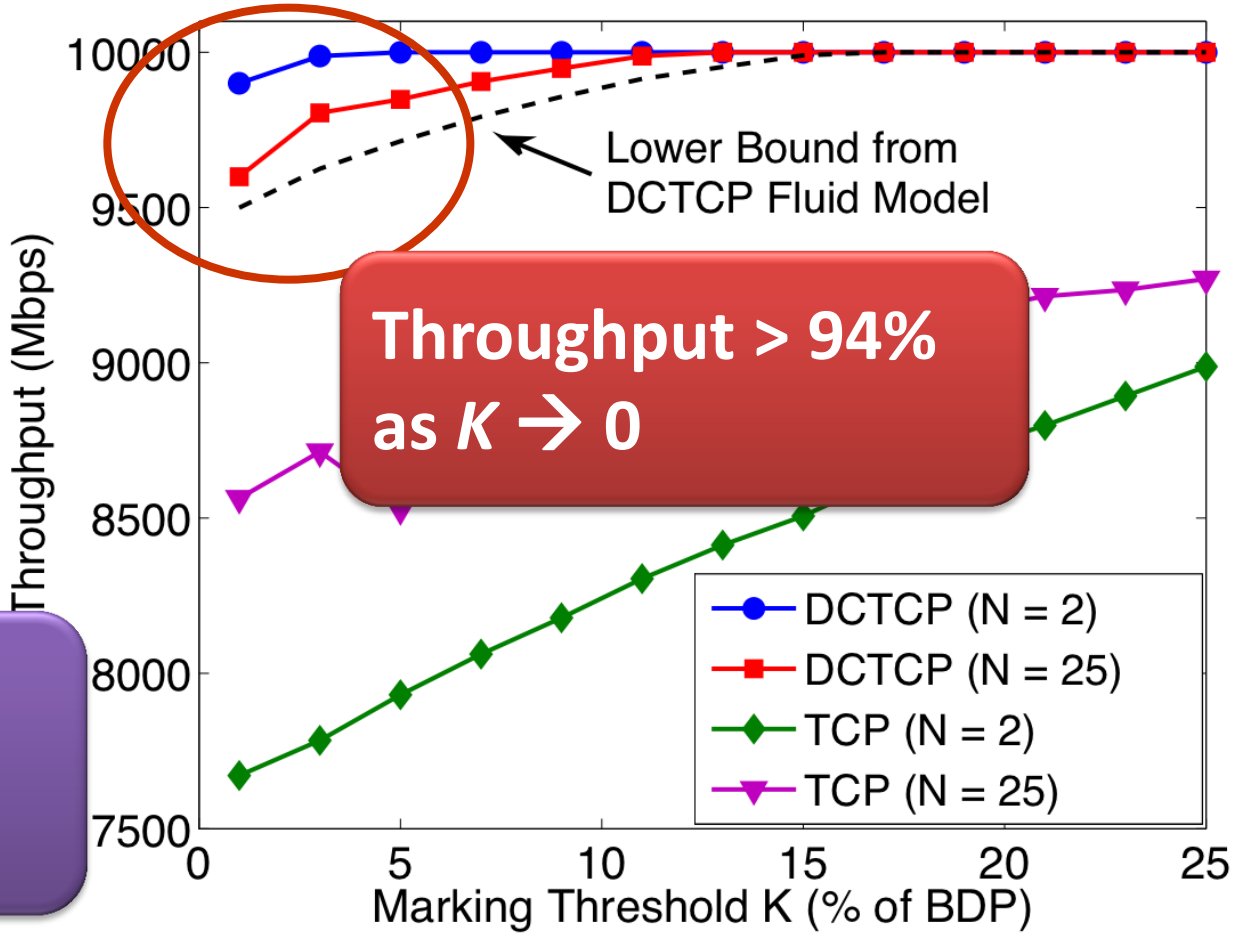
  An environment where all end nodes were ECN-Capable could allow new criteria to be developed for setting the CE codepoint, and new congestion control mechanisms for end-node reaction to CE packets. However, this is a research issue, and as such is not addressed in this document.

- Could use ECT(1) for immediate ECN
  - but this unnecessarily wastes the CE codepoint
    (who would want ‘sluggish ECN’?)

DCTCP in Action



Throughput-Latency Tradeoff



Throughput > 94% as $K \rightarrow 0$

For TCP: Throughput → 75%

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

DCTCP activity

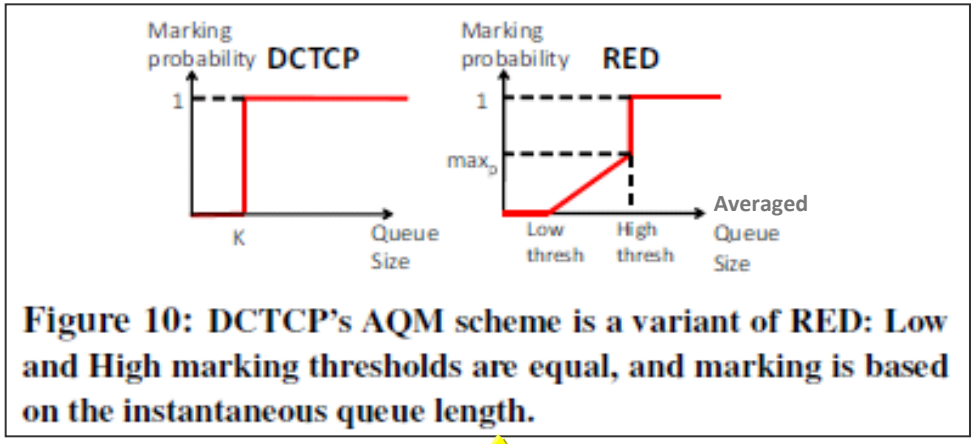


Figure 10: DCTCP's AQM scheme is a variant of RED: Low and High marking thresholds are equal, and marking is based on the instantaneous queue length.

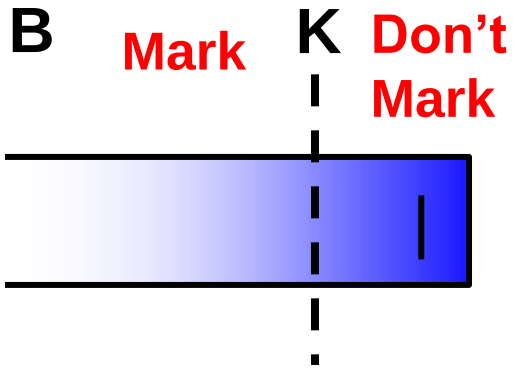
Figures courtesy of Alizadeh et al

- E2e Transport
  - In Windows 8 Server data center template
  - I-D for DCTCP feedback (intended EXP) [draft-kuehlewind-tcpm-accurate-ecn-01]
- AQM
  - Existing kit: Just a degenerate config of RED
  - Can be implemented as just a step at K packets (single 'if' command)
  - For zero-delay can use a virtual queue [RC5670]
    - hardware implementations ["How to Build a Virtual Queue from Two Leaky Buckets"]
    - see HULL for specifics with DCTCP
- Analysis, papers, Linux & ns2 implementation, etc
  - <http://www.stanford.edu/~alizade/Site/DCTCP.html>
  - SIGCOMM paper gives entry point

Data Center TCP Algorithm

Switch side:

- Mark packets when **Queue Length > K**



Sender side:

- Maintain ***moving average*** of ***fraction*** of marked packets **(α)**

$$\text{each RTT}: \; F = \frac{\#\text{ of marked ACKs}}{\text{Total \# of ACKs}} \quad \Rightarrow \quad \alpha \leftarrow (1-g)\alpha + gF$$

- Adaptive congestion window decrease:

$$W \leftarrow (1-\frac{\alpha}{2})W$$