Some Congestion Experienced

draft-morton-tsvwg-sce-00

Jonathan Morton
Pete Heist
Rodney W Grimes

Some Congestion Experienced

```
Binary  Keyword                                  References
------  -------                                  ----------
  00     Not-ECT (Not ECN-Capable Transport)     [RFC3168]
  01     SCE (Some Congestion Experienced)       [This Internet-draft]
  10     ECT (ECN-Capable Transport)             [RFC3168]
  11     CE (Congestion Experienced)             [RFC3168]

+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
|               |           | E | C | E | U | A | P | R | S | F |
| Header Length | Reserved  | S | W | C | R | C | S | S | Y | I |
|               |           | C | R | E | G | K | H | T | N | N |
|               |           | E |   |   |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

- Redefines ECT(1) as SCE, a high-fidelity congestion signal.
- Retains all other RFC-3168 details, for full backwards compatibility.
- Uses the former NS bit as ESCE, for TCP feedback.
- Multiple instances of running code available!

SCE Design Philosophy

"First, do no harm."

Hippocrates

"Heterogeneity is inevitable and must be supported by design."

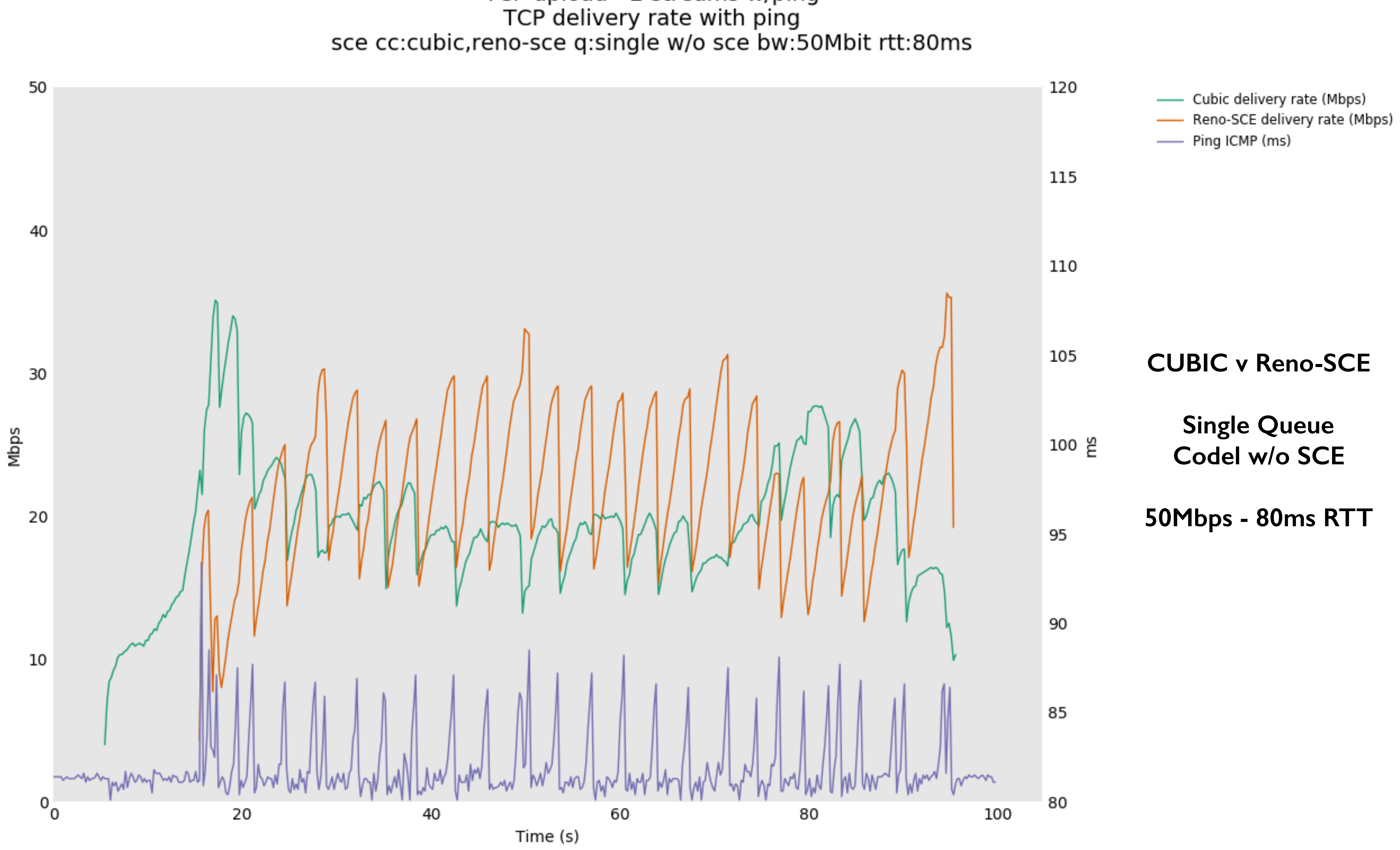
RFC-1958 § 3.1

"Effective congestion control is REQUIRED."

RFC-8311 § 5.2.1

SCE Backwards Compatibility

- Normal AIMD principles apply
  - Normal cwnd growth rate (à la Reno, CUBIC).
  - Response to loss is TCP friendly.
  - Response to CE marks is RFC-3168 compliant.
- Existing RFC-3168 middlebox AQMs treat SCE marks as ECT
  - Can still mark them with CE.
- Existing endpoints ignore SCE marks and NS bit
  - Transparent fallback to RFC-3168.
- Meaning of CE preserved
  - SCE can be a soft "cruise control" signal.
  - Advantage over DCTCP.



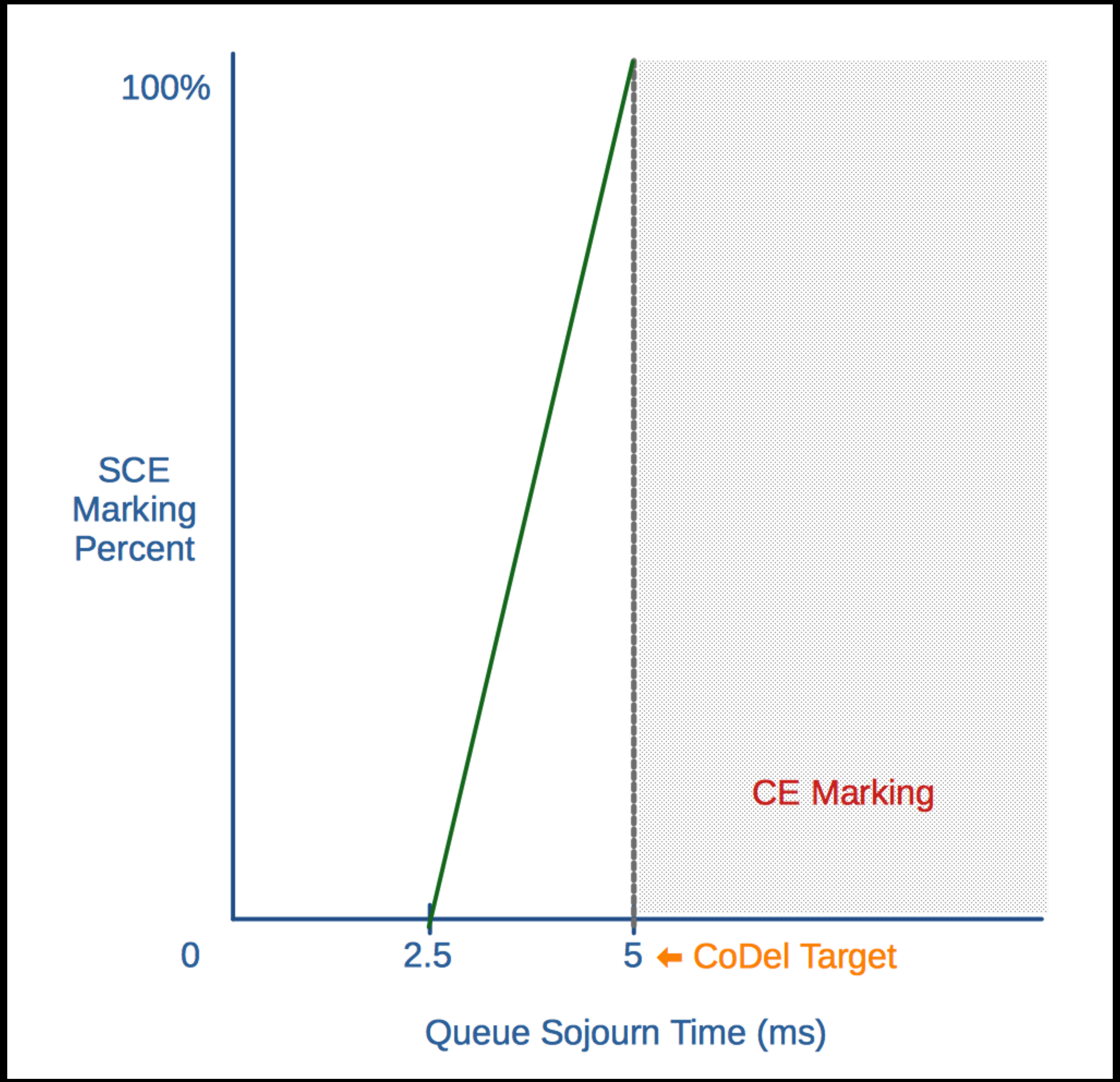
CUBIC v Reno-SCE

Single Queue
Codel w/o SCE

50Mbps - 80ms RTT

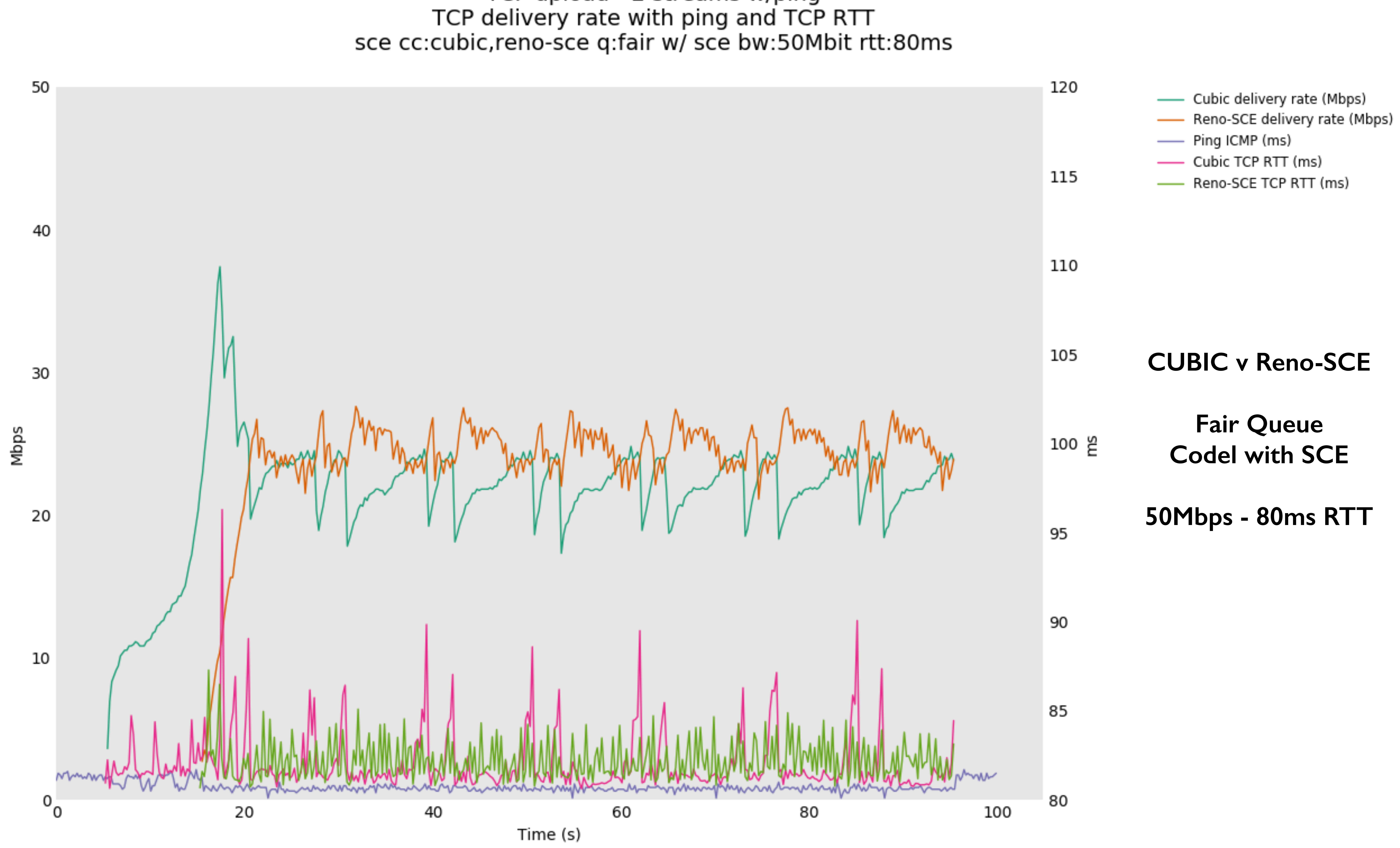
SCE: Signal Network ⇛ Endpoints

- High-fidelity congestion control signal
  - Many marks per RTT, versus many RTTs per mark, in steady state.
- Easily added to existing AQM algorithms
  - Easiest if FQ is also implemented.
  - Threshold function is valid.
  - Ramp functions perform better.
- RFC-3168 CE marks still relevant
  - Large reductions in path capacity.
  - Backwards compatibility.



SCE / ESCE Response

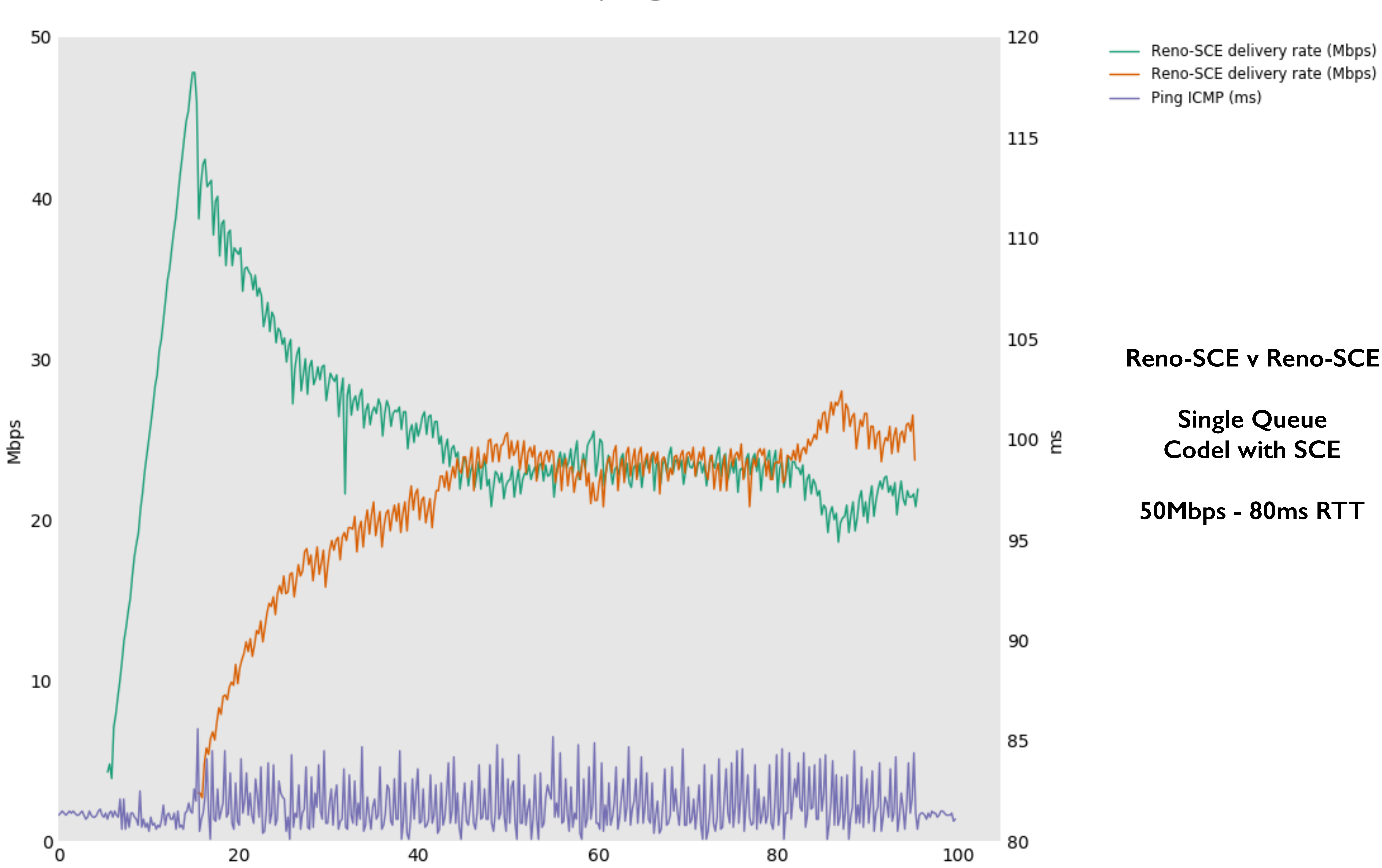
- Former NS bit redefined as ESCE
  - Orthogonal to RFC-3168 ECE & CWR bits.
  - When set, indicates currently acked segment(s) carried SCE mark.
  - Receiver logic is very simple, immediate and almost stateless.
- Sender responds:
  - MAY ignore (backwards compatibility).
  - DCTCP-SCE reduces cwnd by ½ segment per marked segment.
  - Reno-SCE reduces cwnd by $^{1}/_{\sqrt{cwnd}}$ segments per marked segment.
  - CUBIC-SCE also reduces the growth rate if in cubic growth phase.
    - (TODO: fix bugs in CUBIC-SCE.)
- We exit slow-start on a single SCE mark
  - Proceed with congestion avoidance.



CUBIC v Reno-SCE

Fair Queue
Codel with SCE

50Mbps - 80ms RTT



Reno-SCE v Reno-SCE

Single Queue
Codel with SCE

50Mbps - 80ms RTT

SCE: Dual Queues

We prefer FQ. However…

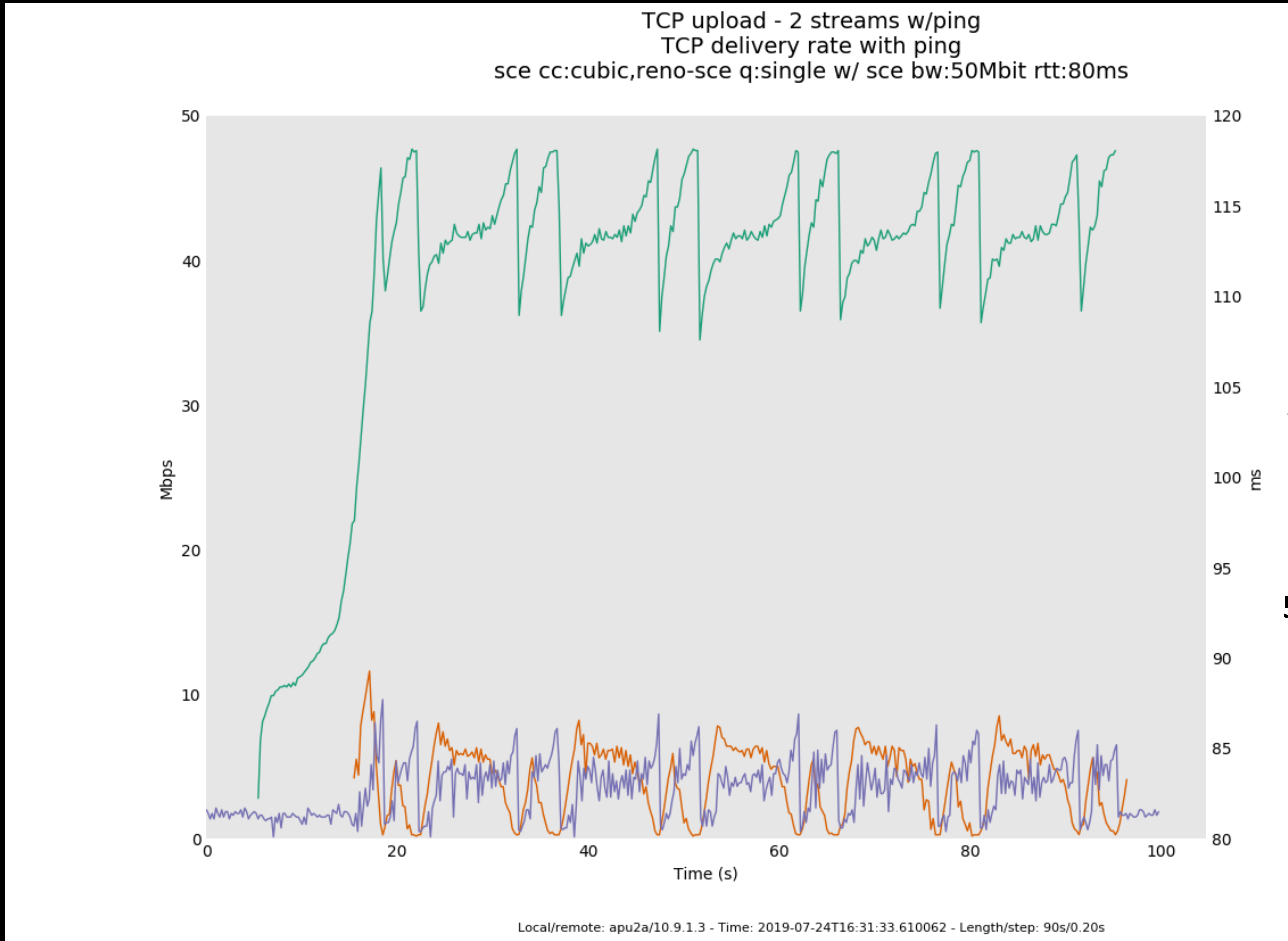
- Any traffic classifier may be used to direct SCE traffic into a second, SCE-specialised queue.
  - Such as a DSCP.
  - "Conventional" traffic defaults to the first queue.
- Robustness against misclassification:
  - (eg. DSCPs bleached en route)
  - "Conventional" queue SHOULD NOT mark with SCE.
  - Misclassified SCE traffic adopts RFC-3168 behaviour & coexists naturally.
- Major benefit of unambiguous signalling via the extra codepoint!

SCE: Single Queue

We prefer FQ. However…

As standard, SCE yields very politely to conventional traffic.

This may actually be useful to some people wanting a "scavenging" transport.



CUBIC v Reno-SCE

Single Queue
Codel with SCE

50Mbps - 80ms RTT

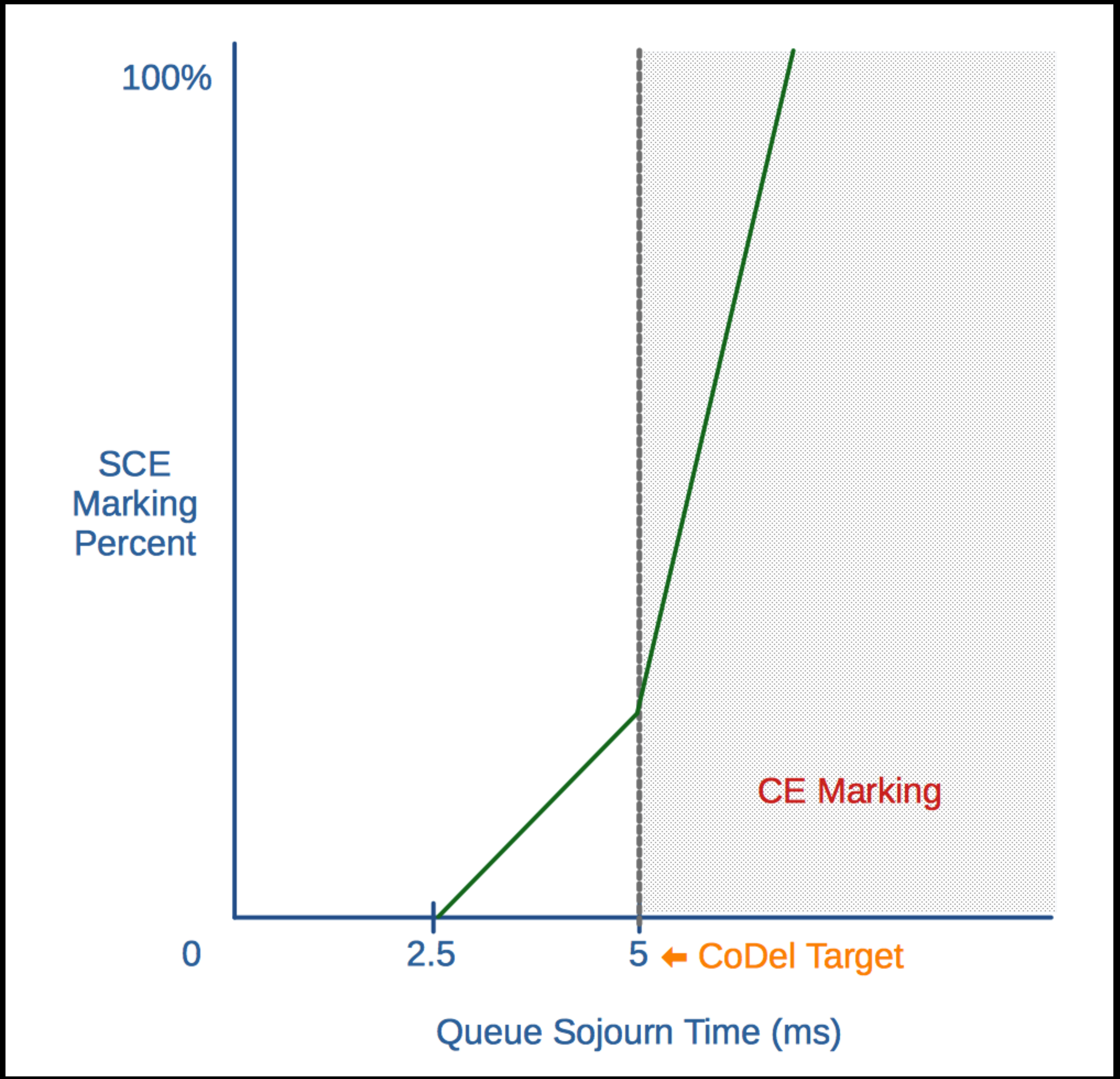
SCE: Single Queue

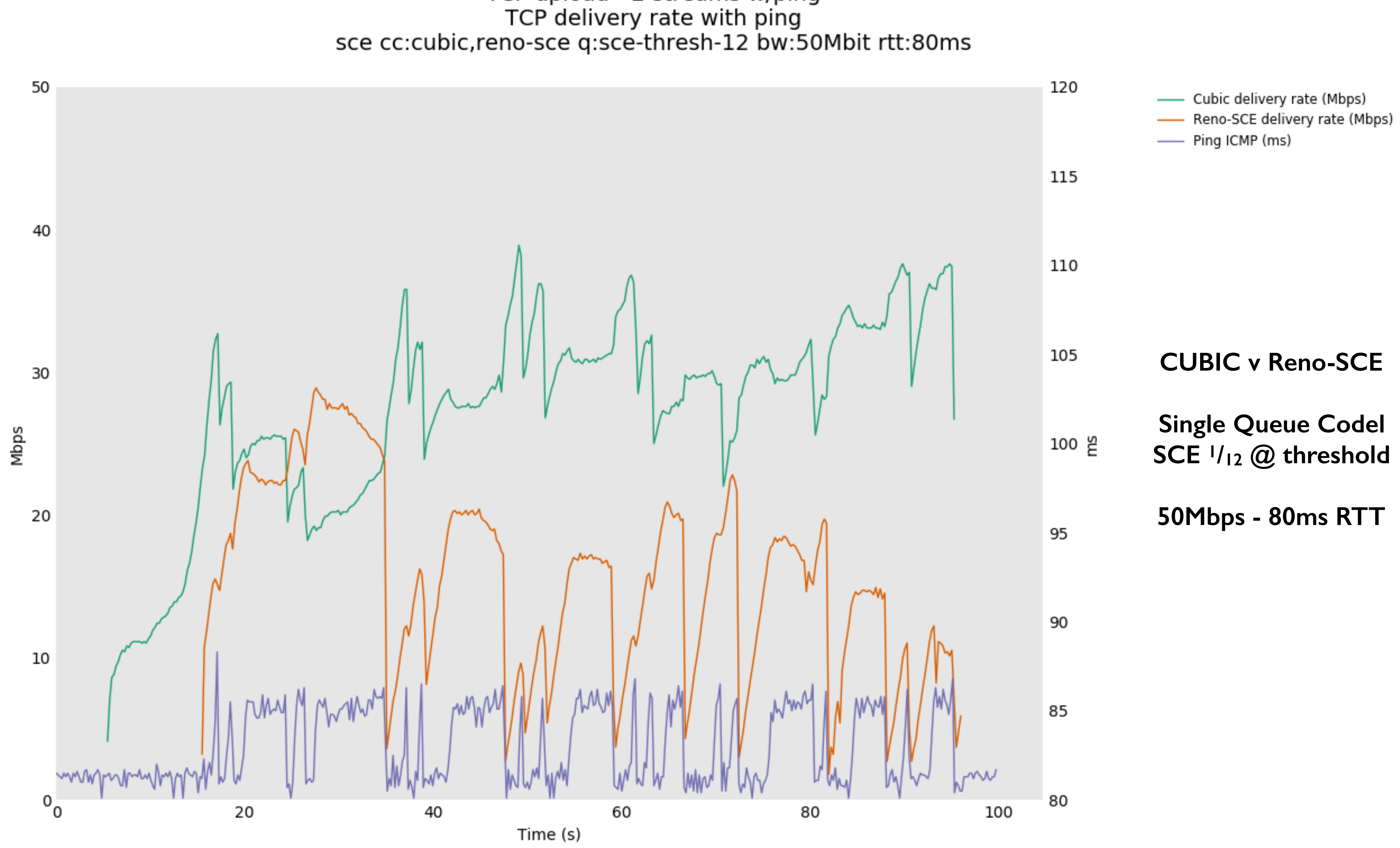
We prefer FQ. However…

By modifying the SCE marking probability ramp, some of SCE's benefits can be realised without requiring multiple queues.

There are some compromises in performance, and the current implementation is not knob-free, but SCE can coexist fairly with conventional traffic and run smoothly by itself.

The ramp parameters are chosen at the single-queue bottleneck node in question.

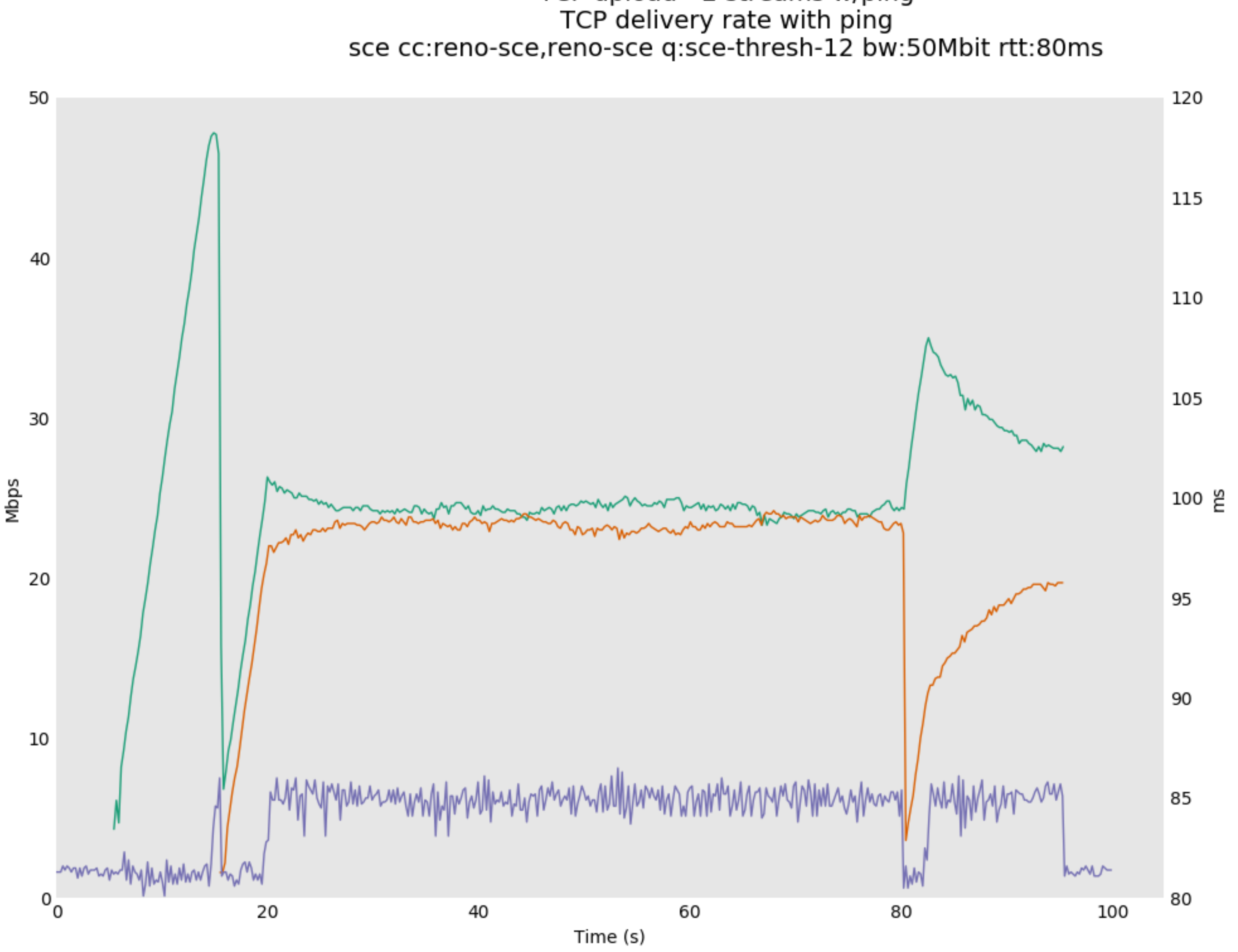




CUBIC v Reno-SCE

Single Queue Codel
SCE $^1/_{12}$ @ threshold

50Mbps - 80ms RTT



Reno-SCE v Reno-SCE

Single Queue Codel
SCE 1/12 @ threshold

50Mbps - 80ms RTT

SCE: Running Code

- At IETF-104 (Prague), we had:
  - FreeBSD sender (DCTCP-SCE)
  - FreeBSD receiver (rudimentary SCE ⇛ ESCE echo)
  - Linux middlebox AQMs (fq_codel, Cake)
- At IETF-105 (Montreal), we have:
  - Linux senders (DCTCP-SCE, Reno-SCE)
  - Linux receiver (accurate SCE ⇛ ESCE transform)
  - FreeBSD middlebox AQM (thanks to Loganaden Velvindron)
- Technically, this is <u>multiple</u> instances of running code…
- Work continues to mature, diversify, and characterise.

Notes on TCP Pacing

- High fidelity congestion signalling is <u>very</u> sensitive to burstiness.
  - Bursts collect transiently in queue and cause SCE signalling.
  - Ack clocking is not sufficient, especially with delayed acks.
  - Pacing is effectively <u>mandatory</u>.
- Linux exempts first 10 packets from pacing.
  - Prevents slow-start from working with SCE.
  - We patched that out of our kernel - one-line fix.
- Transition from SS to CA phase needs halving of send rate.
  - SS doubles per RTT - first response delayed by one RTT.
  - Traditionally provided by first loss/CE response.
  - We use modified pacing scale factors: SS=100% CA=40%.

Some Congestion Experienced

Any questions?