RACK: a time-based fast loss detection for TCP
draft-cheng-tcpm-rack-00

Yuchung Cheng <ycheng@google.com>

IETF94 Yokohama, Nov 2015

A quarter-century of counting packets for recovery

RFC5681: DupAck threshold (DupThresh)

RFC6675: Total SACKed > DupThresh

FACK: Highest SACKed > DupThresh

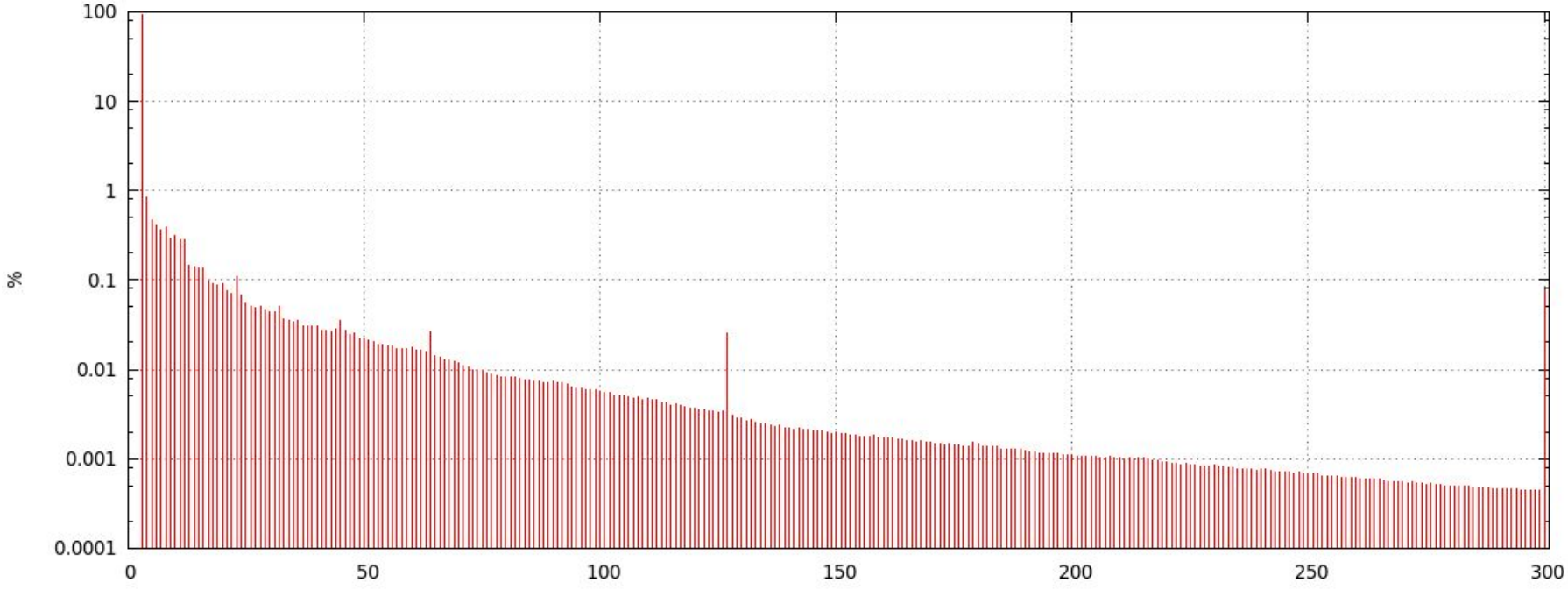
RFC5827: DupThresh = 1 if cwnd < 4

Thin-stream: DupThresh = 1 on thin-stream

RFC4653: DupThresh = FlightSize / 2

Reordering-detection: DupThresh to maximum reordering packet distance

Reordering in packet distance is deceiving



DupThresh on YouTube TCP

Design Rationale of a new loss detection

1. Replace all DupThresh magic with the notion of time

2. Robust to small reordering
   a. Packets traversing on slightly different physical paths
   b. Out-of-order delivery in (wireless) link layer

3. Detect tail drops and lost retransmit well

4. Use every (re)transmission to detect loss, including TLP and RTO probes

5. Decoupled from congestion control

Algorithm

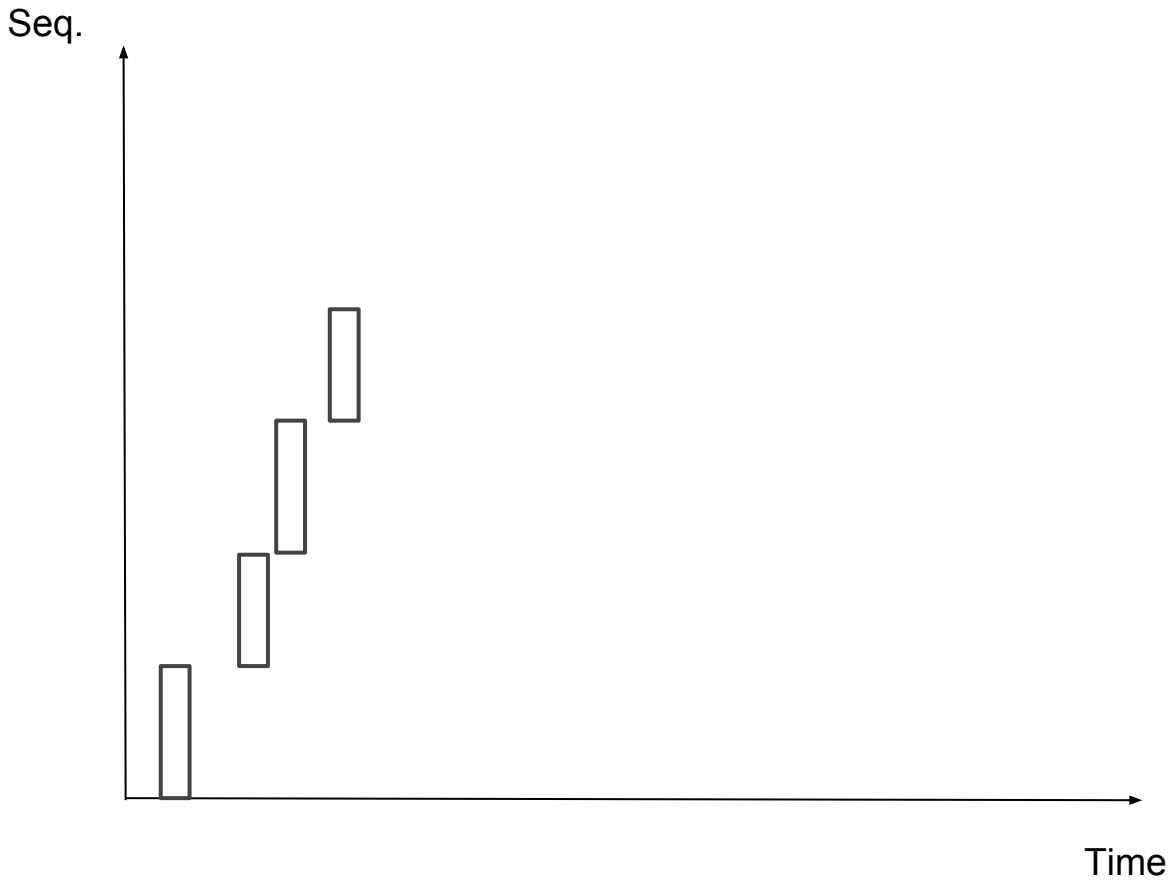
Packet A is lost if some packet B sent sufficiently later is s/acked

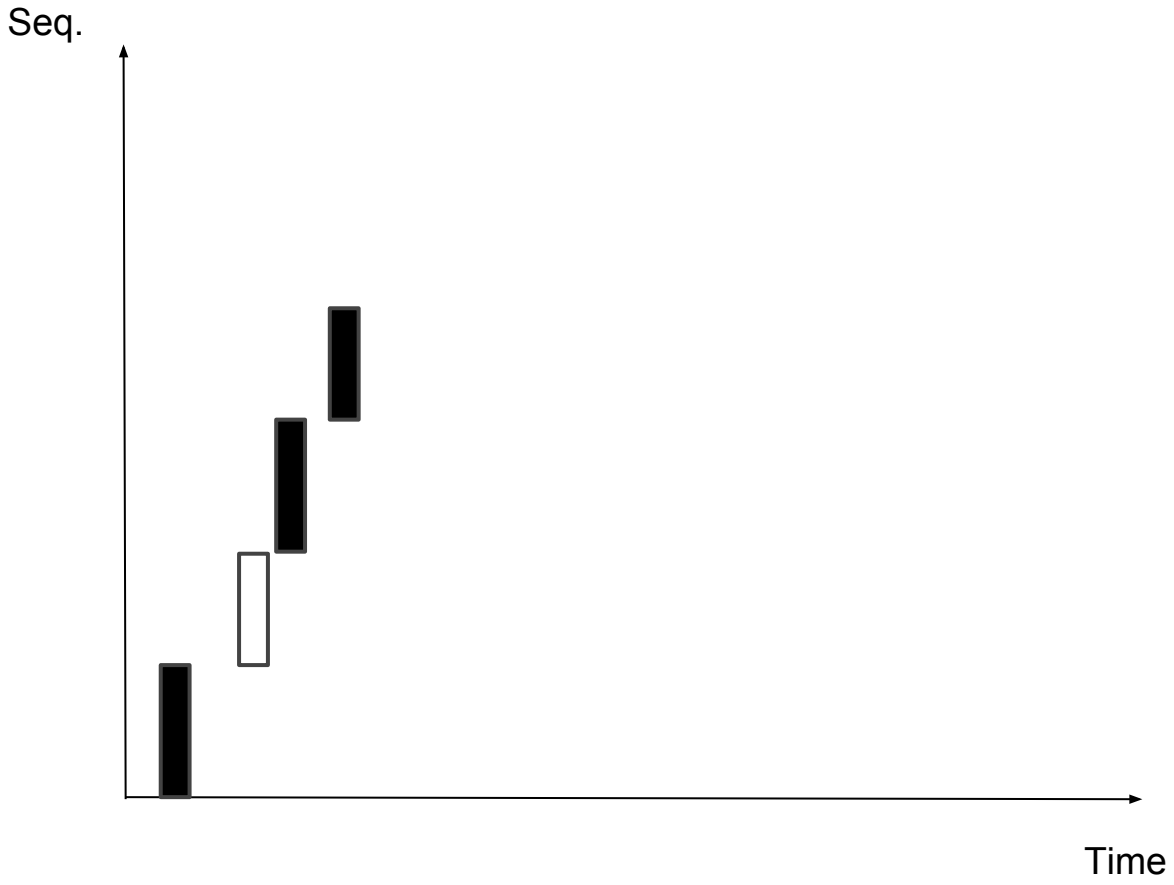
Packet.xmit_time: latest xmit time of a Packet

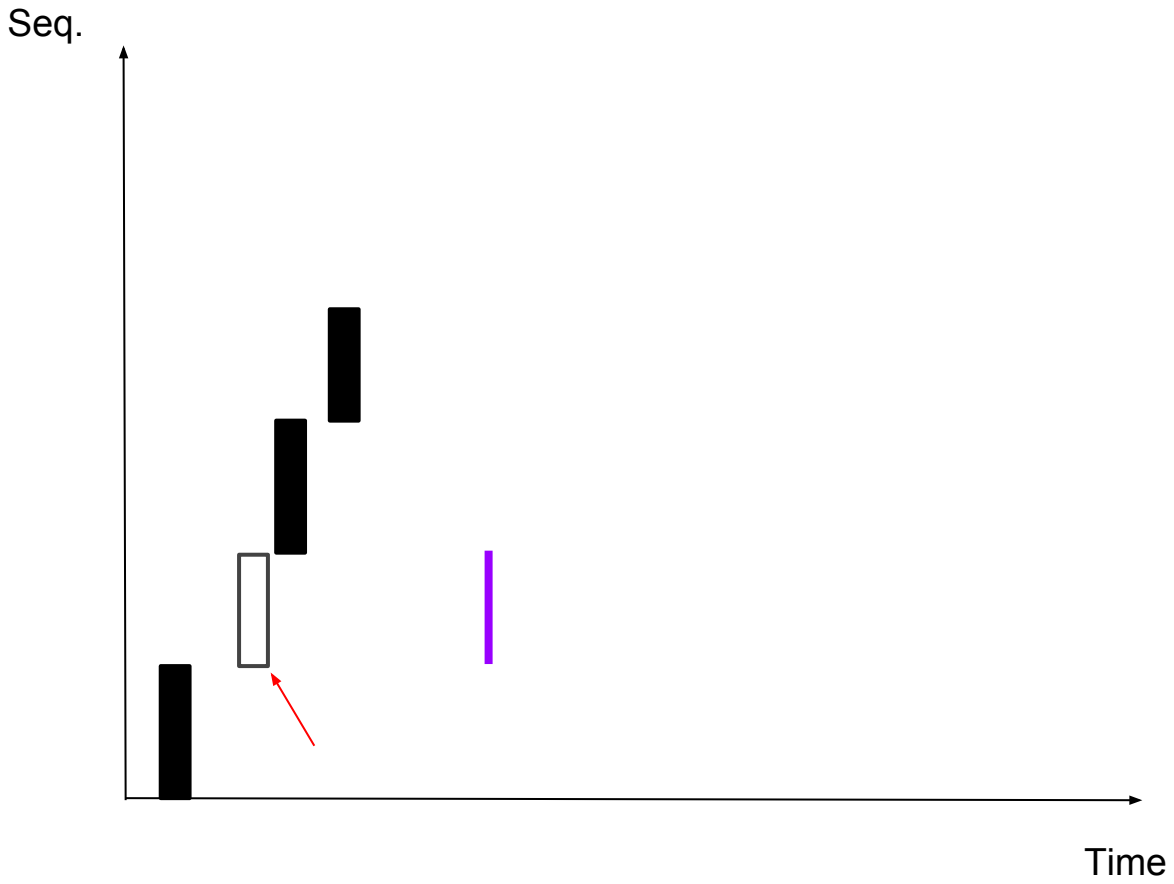
RACK.xmit_time: most recent Packet.xmit_time among SACKed or ACKed packets

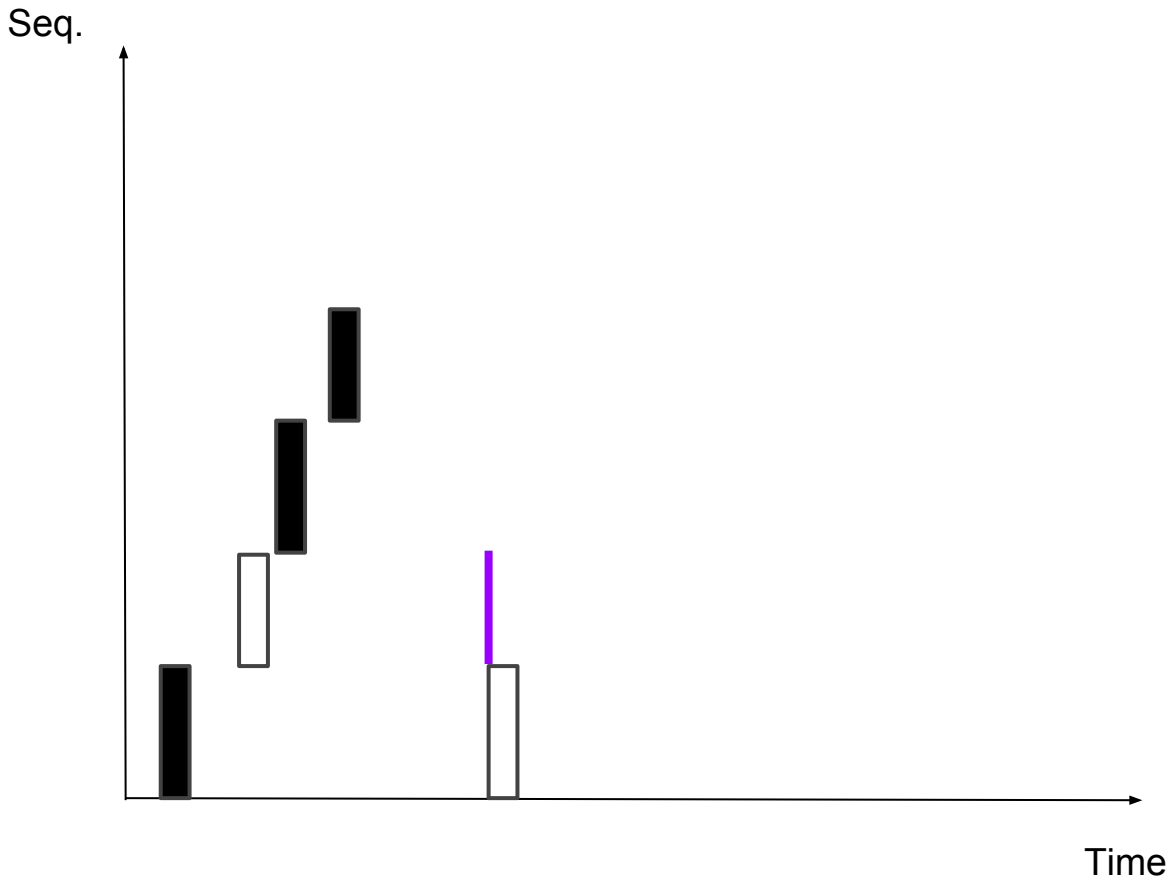
RACK.RTT: associated RTT of RACK.xmit_time

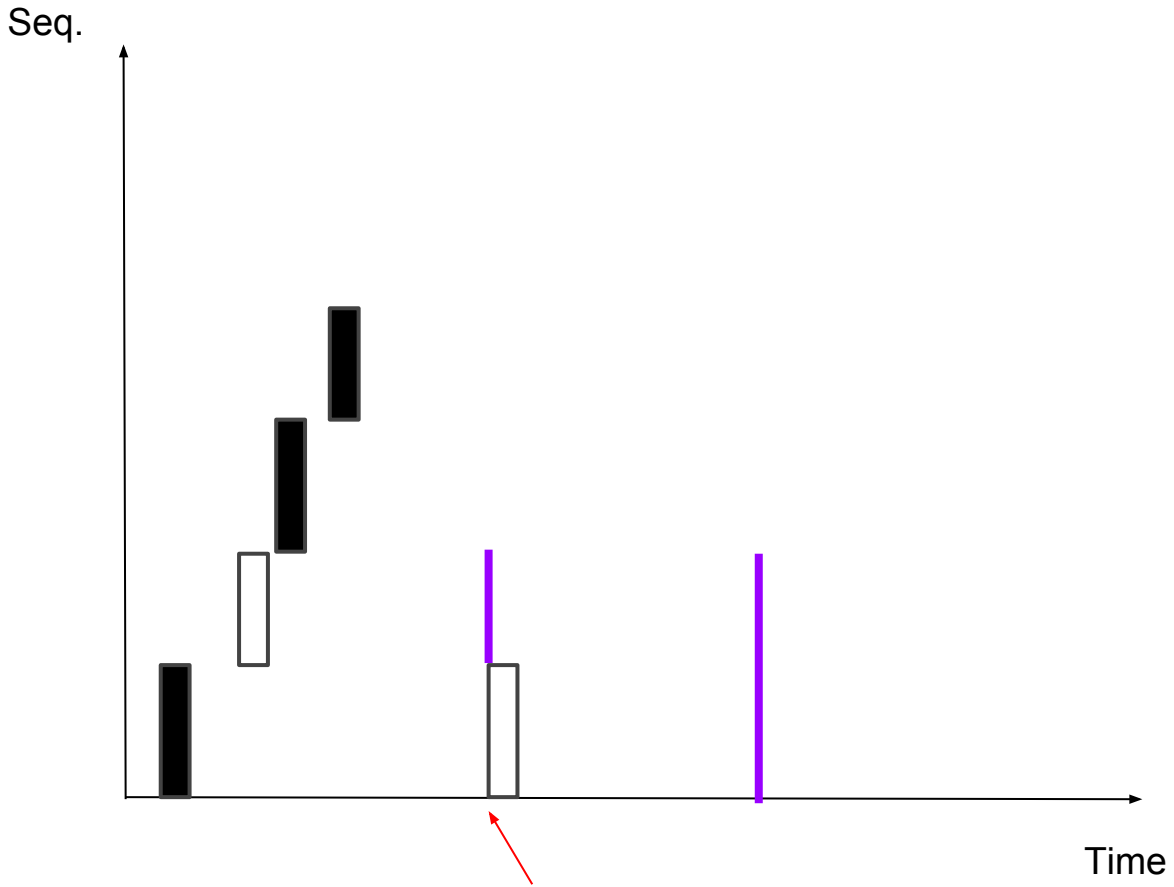
RACK.reo_wnd: reordering window

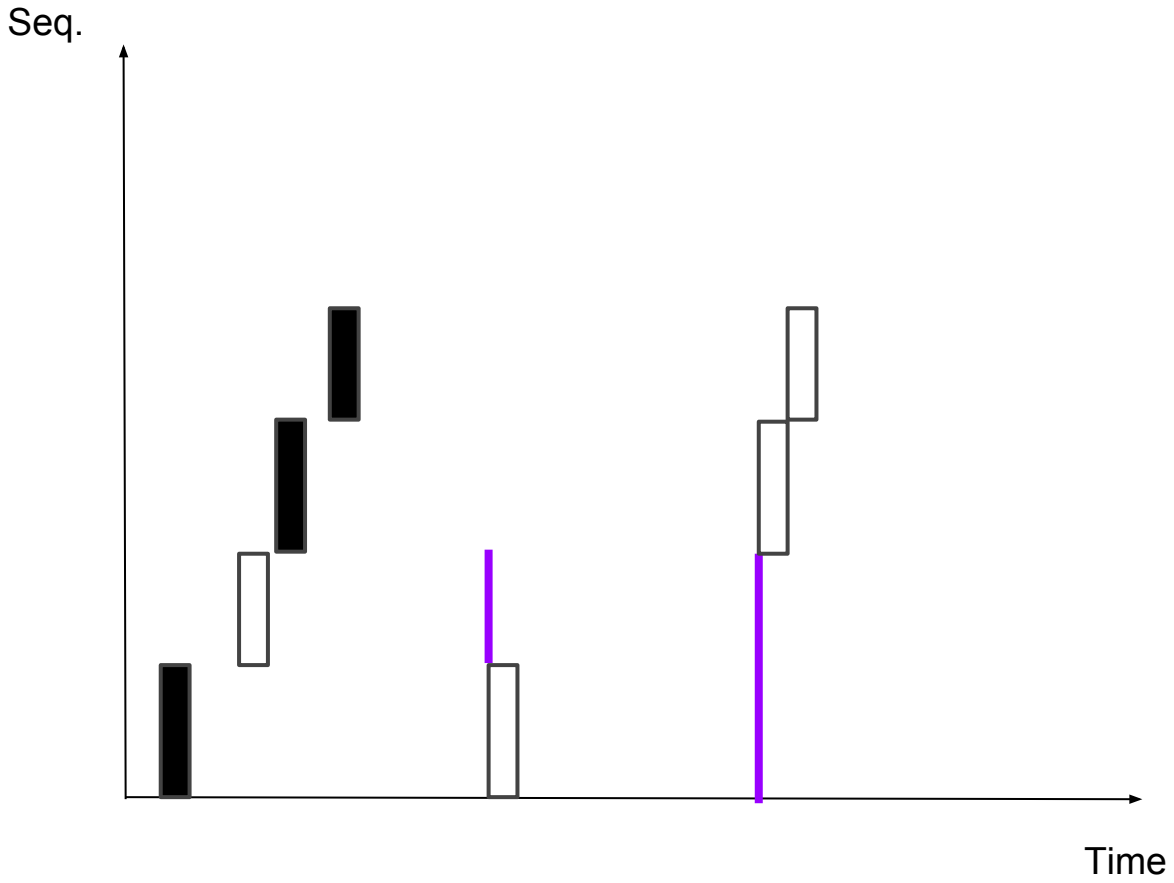

Seq.
Time

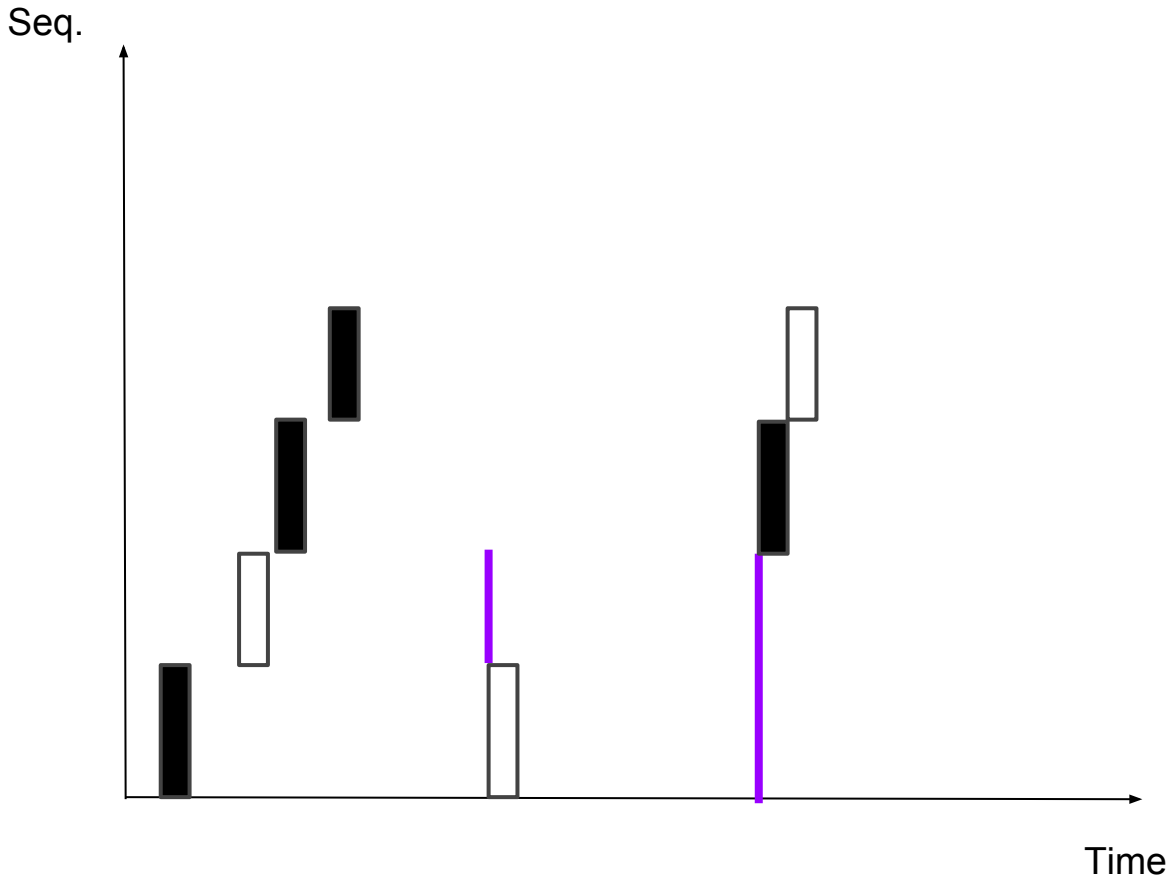

Seq.
Time

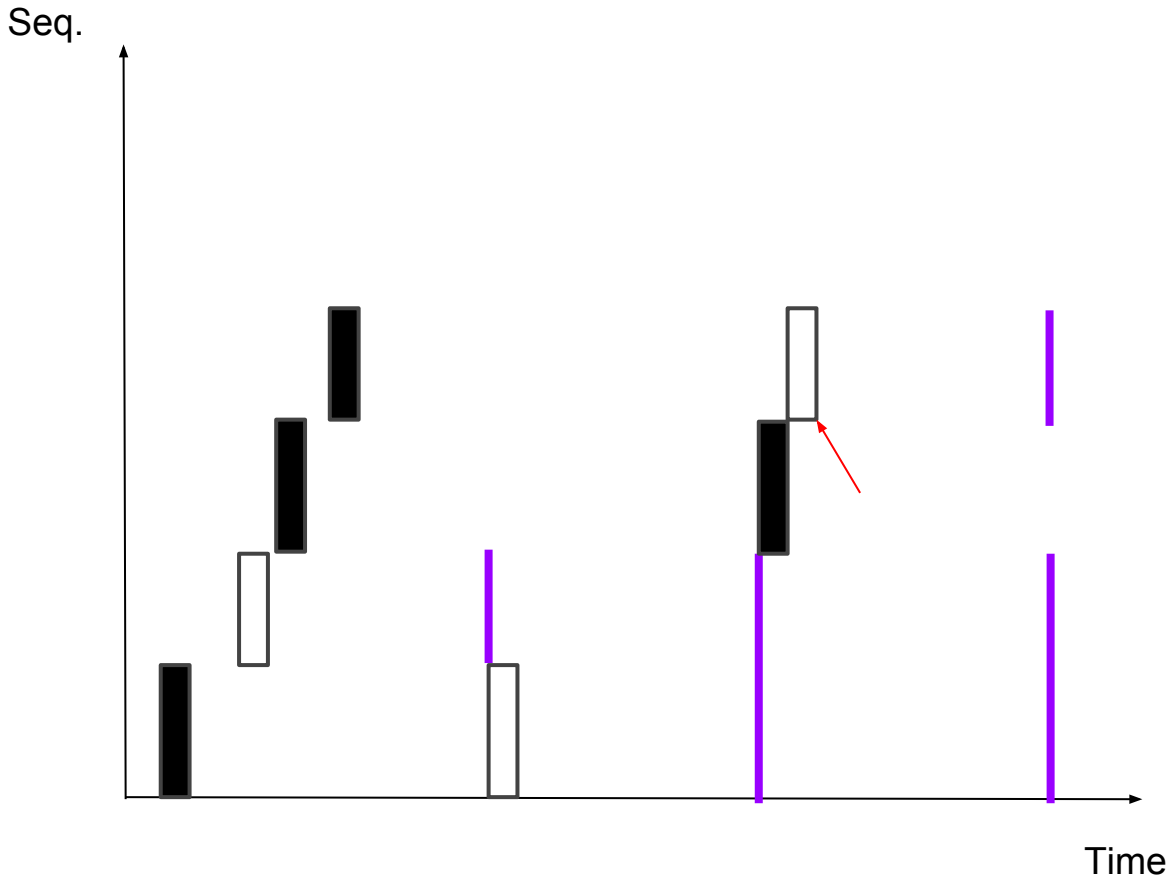

Seq.
Time

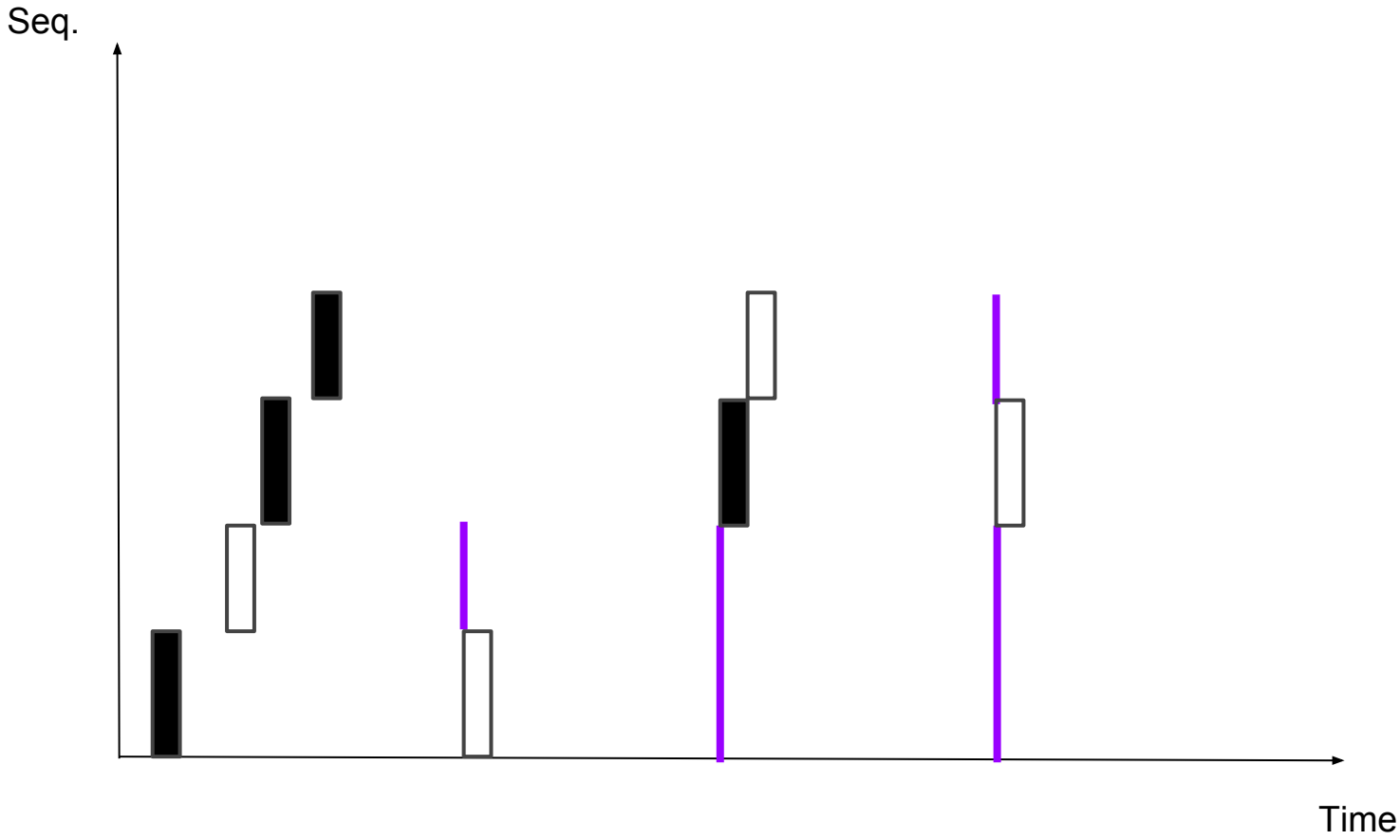

Seq.
Time

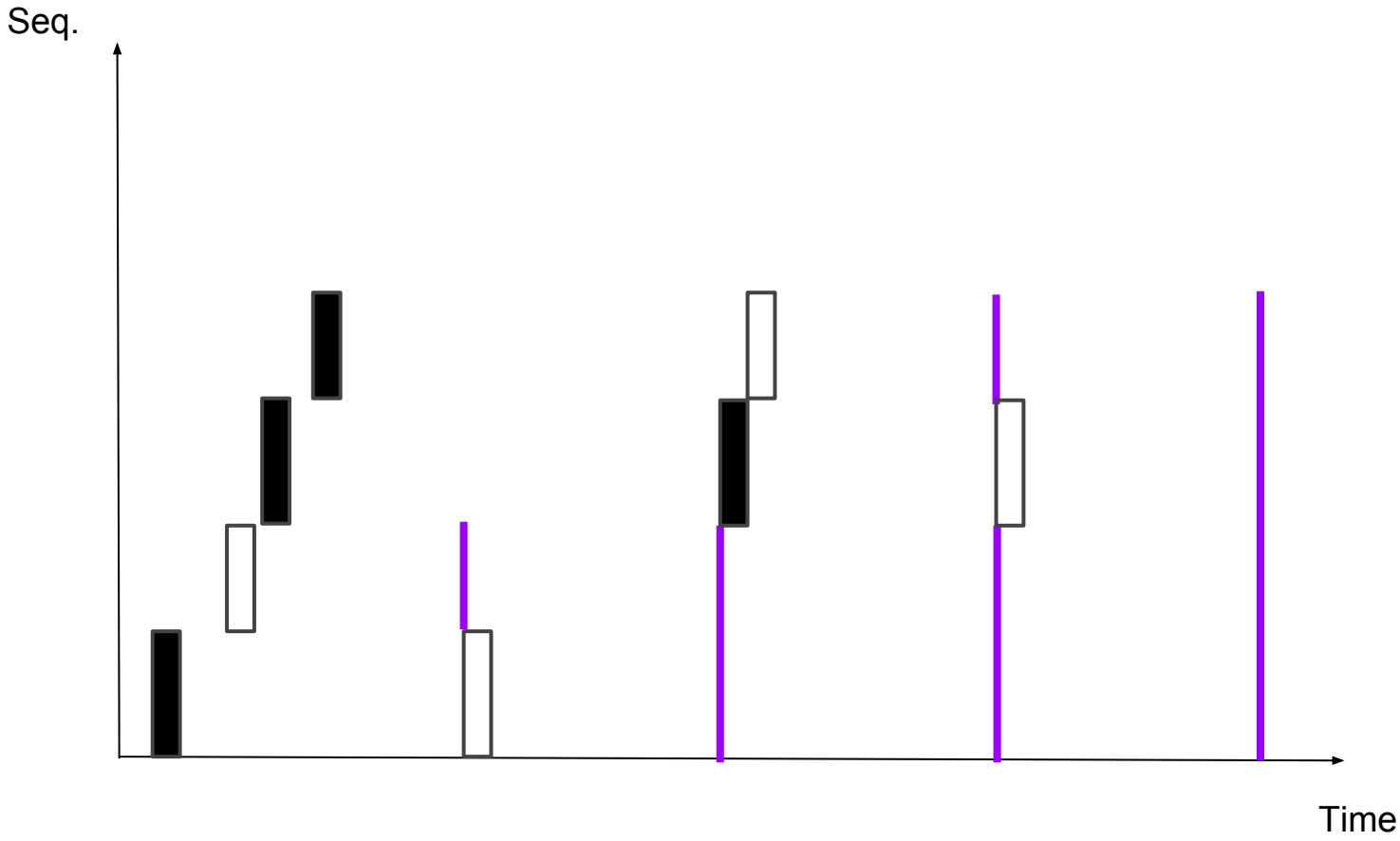

Seq.
Time


Seq.
Time


Seq.
Time


Seq.
Time


Seq.
Time


Seq.
Time

Algorithm

Init: RACK.reo_wnd = 1ms

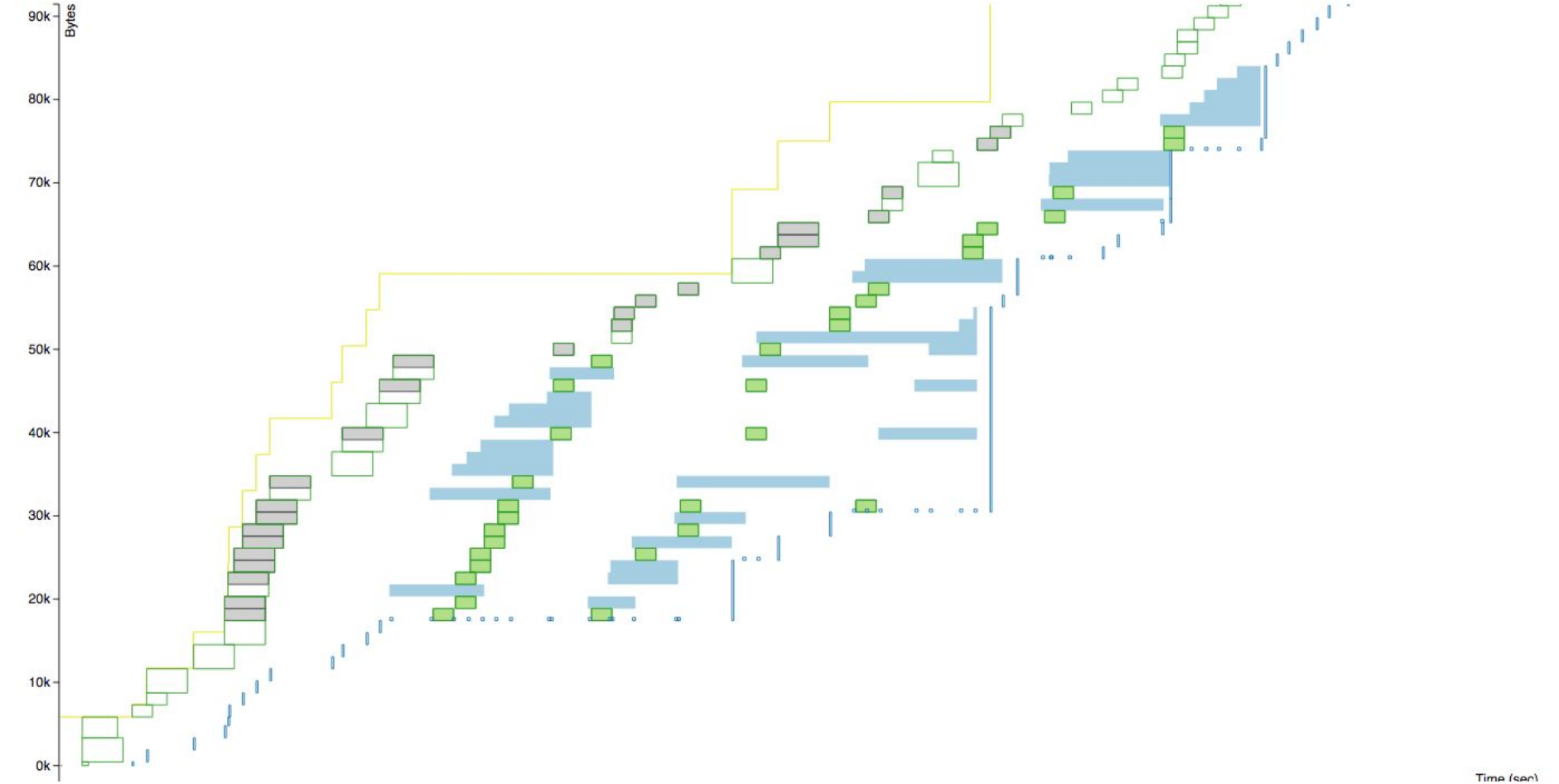
For each (re)transmission record its Packet.xmit_time

For each Packet newly s/acked:

RACK.xmit_time = most recent Packet.xmit_time
RACK.RTT == now - RACK.xmit_time
RACK.reo_wnd = RACK.min_RTT / 4 (if detected reordering)

For each Packet not yet s/acked:

Mark lost if Packet.xmit_time > RACK.xmit_time + RACK.reo_wnd


Bytes
90k
80k
70k
60k
50k
40k
30k
20k
10k
0k
Time (sec)

Status

Deployed on Google since 2014 and upstreamed to Linux 4.4 in Oct 2015

Currently implemented to co-exist with other DupThresh heuristics

Next steps

1. Experiment retiring other heuristics (FACK, Early retransmit, RFC6675, …)
2. Improve for heavy reordering (e.g., packet spray)
3. Merge draft with draft-dukkipati-tcpm-tcp-loss-probe