FH MÜNSTER
University of Applied Sciences

ETI
FB Elektrotechnik und Informatik
Department of Electrical Engineering
and Computer Science

RACK for SCTP

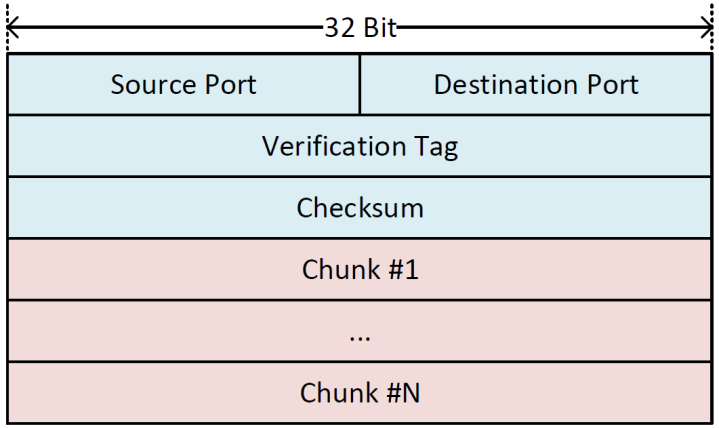
Felix Weinrank
Michael Tüxen
Erwin P. Rathgeb

Agenda

- A brief introdcution to SCTP
- SCTP's default loss recovery
- RACK for TCP
- RACK for SCTP
- Performance evaluation
- Modfications and improvements

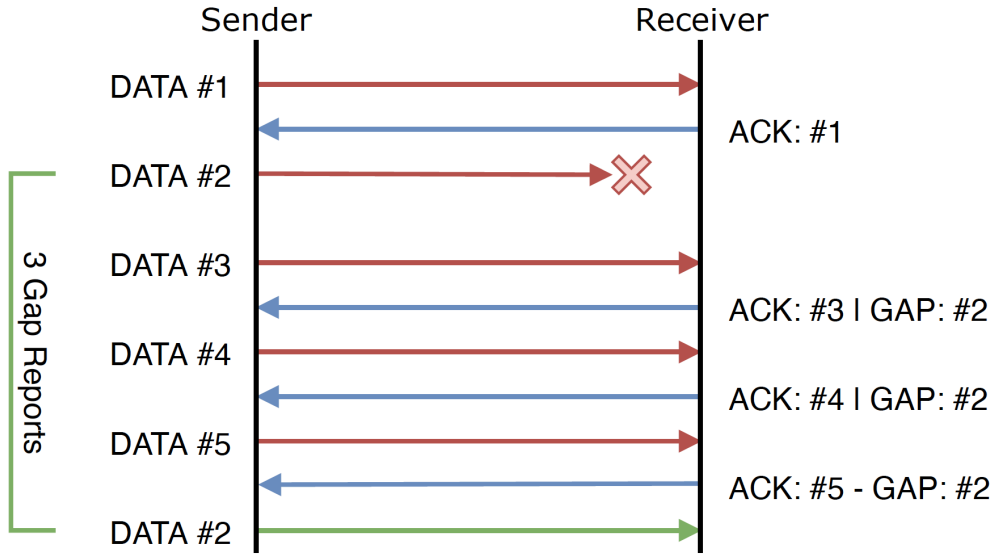
SCTP
Overview

- Stream Control Transmission Protocol
- Connection (“association”) oriented, reliable and message-oriented
- Provides network fault tolerance
  - Support of multihoming
  - Minimisation of head of line blocking
- Originally designed for signalling in telecommunication networks (SS7)
- Now a multi purpose transport protocol, e.g. for WebRTC Data-Channel
- Allows bundling of multiple chunks in a single message



SCTP
Loss recovery

- SCTP uses two loss recovery strategies
  - timer based retransmission (slow! 1s / 200ms)
  - counting loss indications (3 GAP reports → retransmission)



RACK for TCP

Overview

- Originally developed by Google
- Proposed as a full replacement for existing error recovery algorithms
- Currently IETF draft
- Integrated into FreeBSD, Linux and Windows
- RACK ("Recent ACKnowledgment")
  - Fast recovery using time-based inferences
- TLP ("Tail Loss Probe")
  - Leverages RACK and sends a probe packet to trigger ACK feedback
- Sender side only
  - Requires no extensions apart from SACK

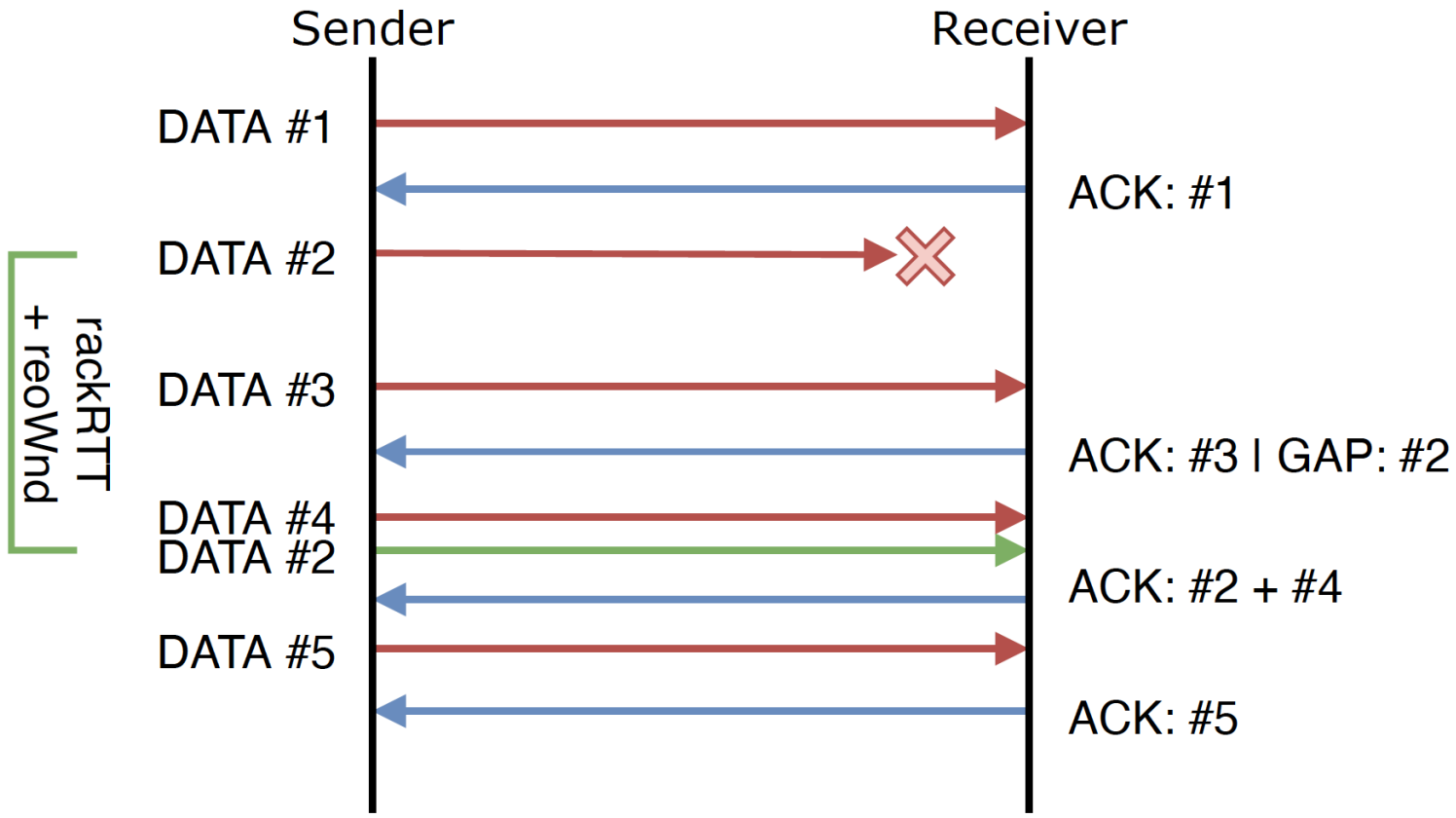
RACK for TCP

How it works

- RACK records the transmission time for every outgoing packet
- If packet has not acknowledged within time and a subsequently sent packet has been acknowledged → retransmission
- RACK considers packet reordering → prevents spurious retransmission
- RackTimeout = rackRTT + 4 x reordering window

RACK for TCP

Operation example



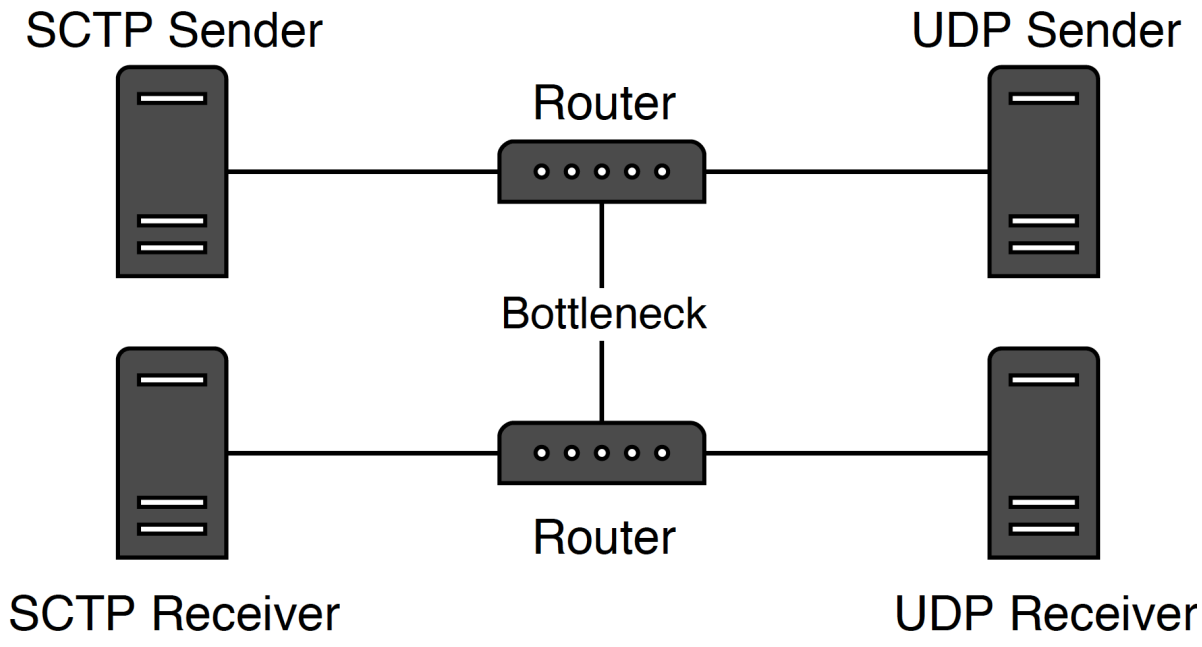
RACK for SCTP

Overview

- SCTP supports all required mechanisms out of the box
  - SACKs always enabled
  - Duplicate packets are always reported to the sender
    - Better reordering window calculation
- Difference: SCTP records transmission time per chunk

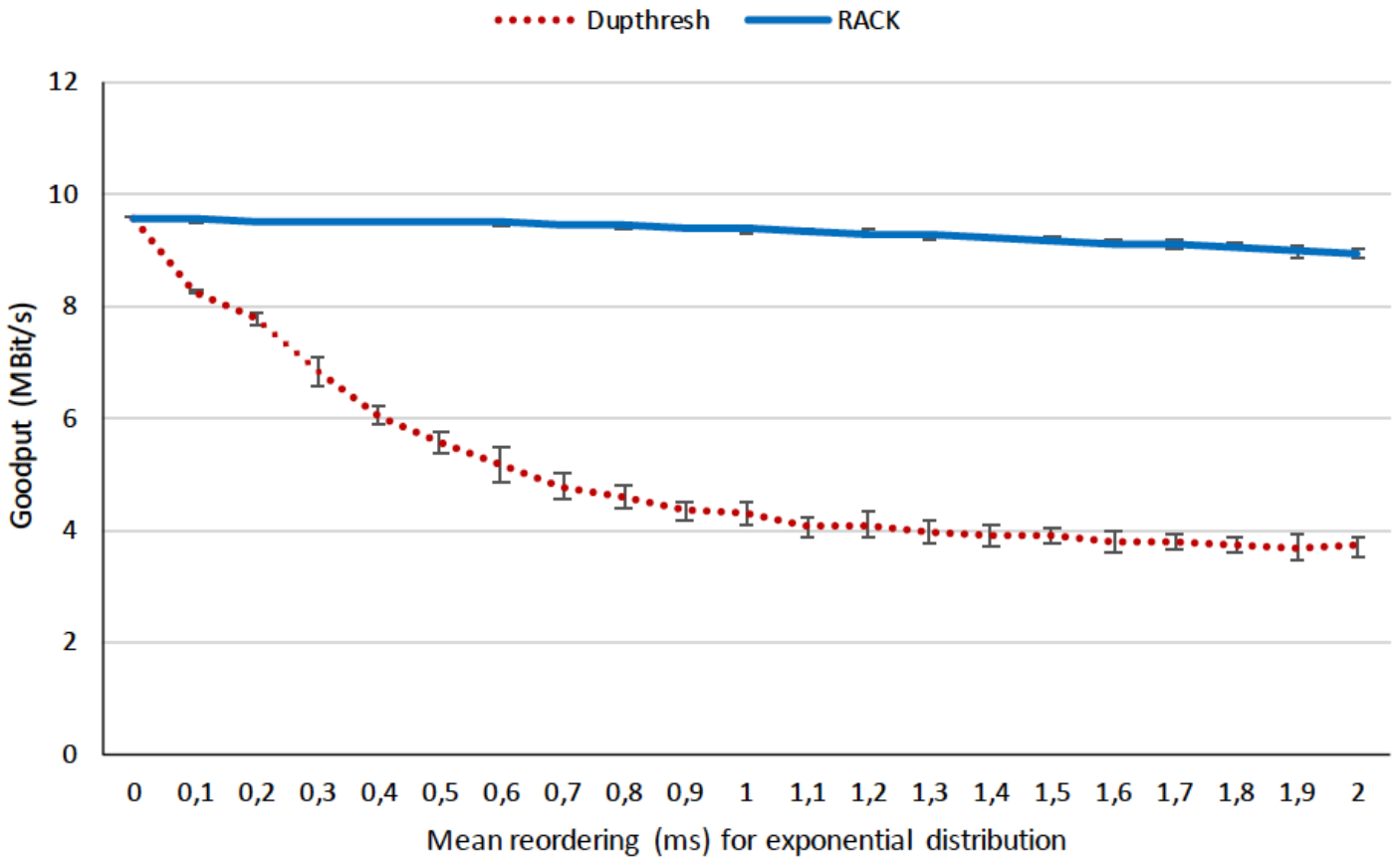
RACK for SCTP

Testbed for simulation



RACK for SCTP

Simulative evaluation



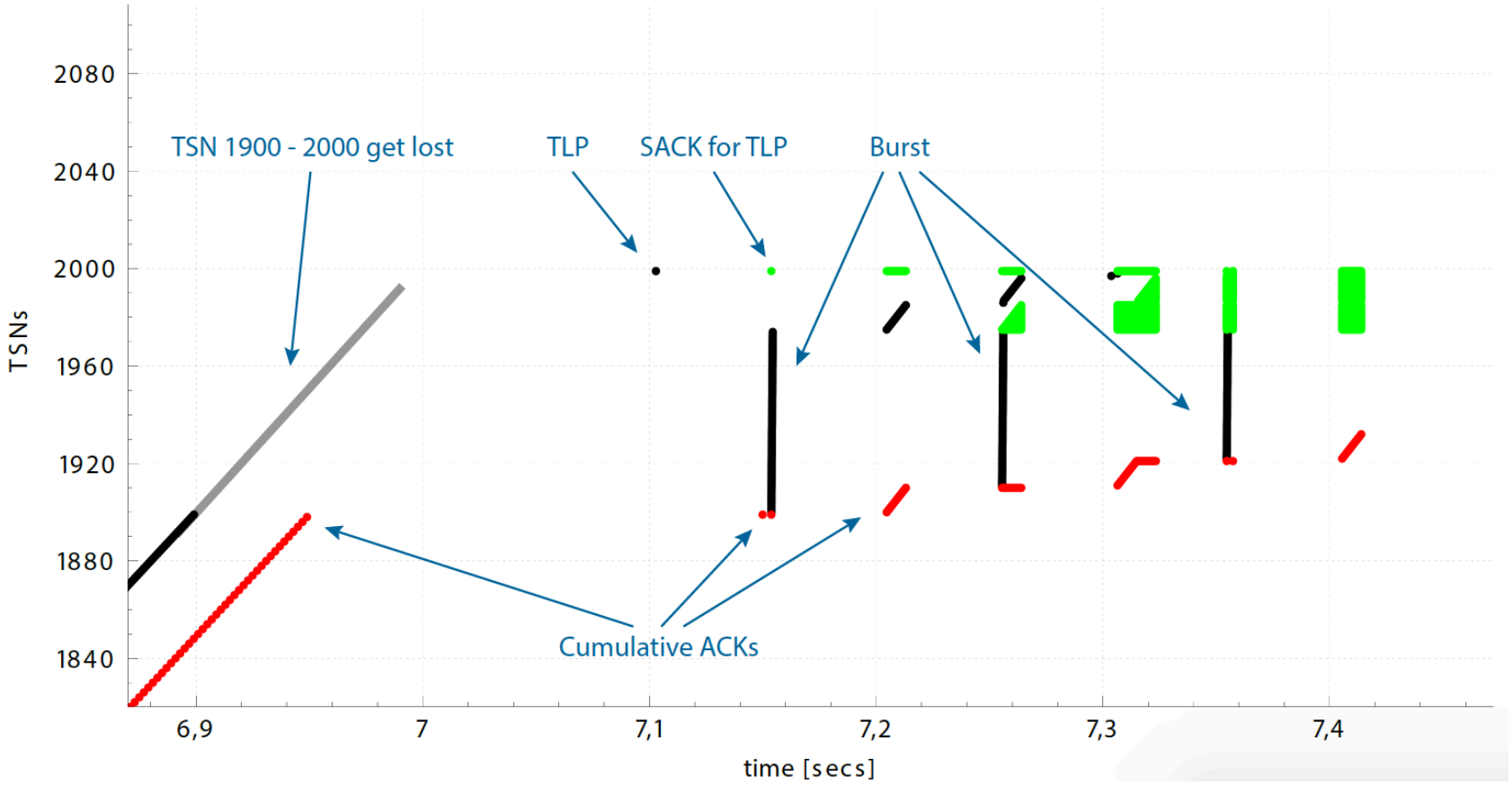
Tail Loss Probing (TLP)

Overview

- Tail Loss: Either the last payload segment(s) or acknowledgements get lost
  - Can not be detected by dupthresh or RACK
  - Are recovered by timer-based retransmissions → slow!
- Common problem for request/response style traffic
  - Google reports that 70% of their losses are recovered by timer-based retransmission
- After every transmission, a probing timer is armed
  - Timeout depends on smoothed RTT and number of packets in flight
- Evaluation shows that the mechanism works well
- But: TLP tends to mark large ranges of packets as lost
  - Burst mitigation needed

Tail Loss Probing (TLP)

Example

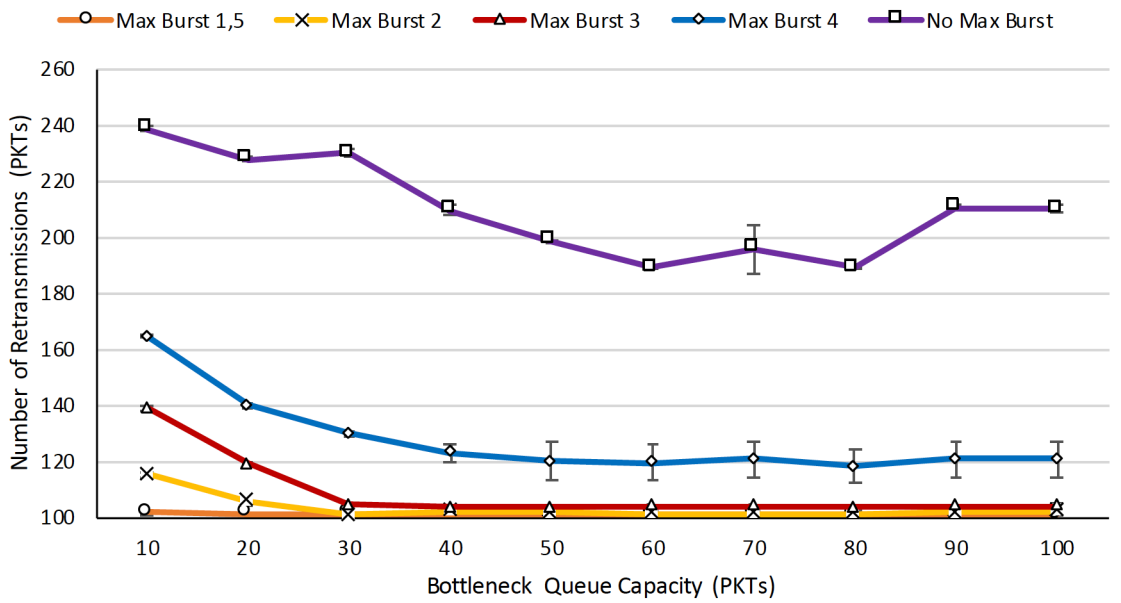


Tail Loss Probing (TLP)

Burst mitigation

FH MÜNSTER
University of Applied Sciences

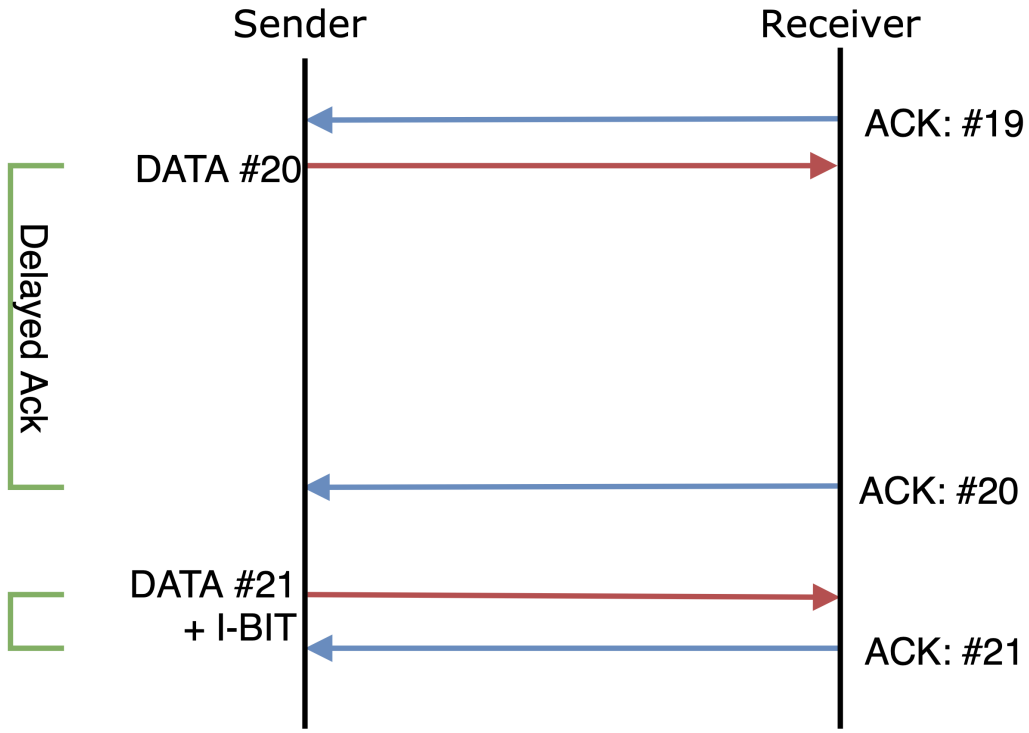
- TLP tends to create large bursts
  - RACK draft suggests Proportional Rate Reduction [RFC6937]
- SCTP already has built-in burst mitigation
  - Limiting the number of packets per acknowledgment (default : 4)
  - Is this mechanism sufficient? It depends!
- We have developed a dynamic burst mitigation algorithm (initial: 2)
  - Max burst reduced by 0.25 if retransmission gets lost
  - Max burst reset to 2 if retransmissions are delivered without loss



Tail Loss Probing (TLP)

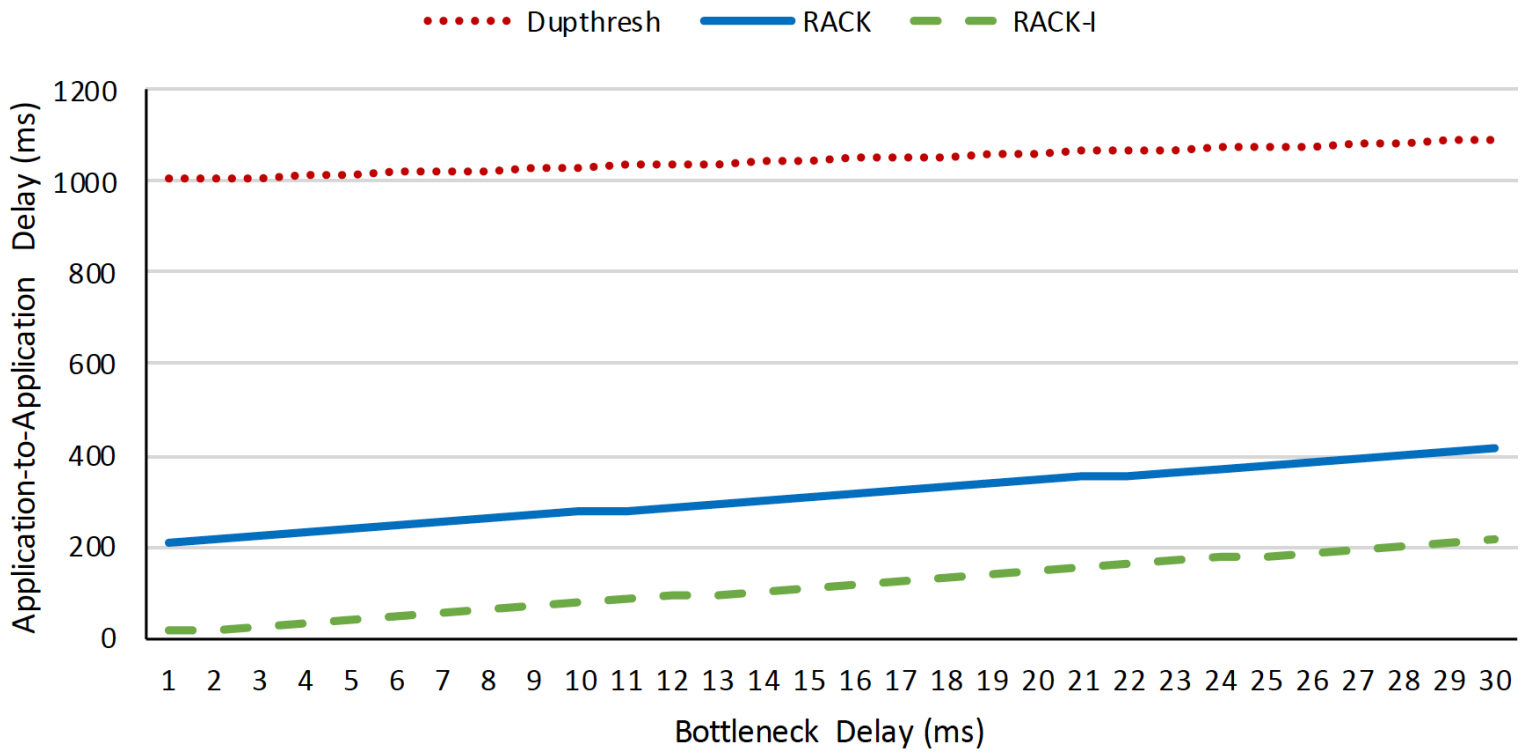
I-Bit

- If only a single packet is in flight, the TLP timer must consider delayed ACKs
  - Delayed ACK reduce the number of ACKs
  - Every second payload carrying packet is acknowledged
  - The worst case delayed ACK timer (WCDelAckT) is 200 milliseconds
- The sender can set an I-Bit to request an ACK without waiting for a subsequent packet



Tail Loss Probing (TLP)

Faster Tail Loss probing



Thank you!