TCP ACK Rate Request (TARR) option

draft-ietf-tcpm-ack-rate-request-04

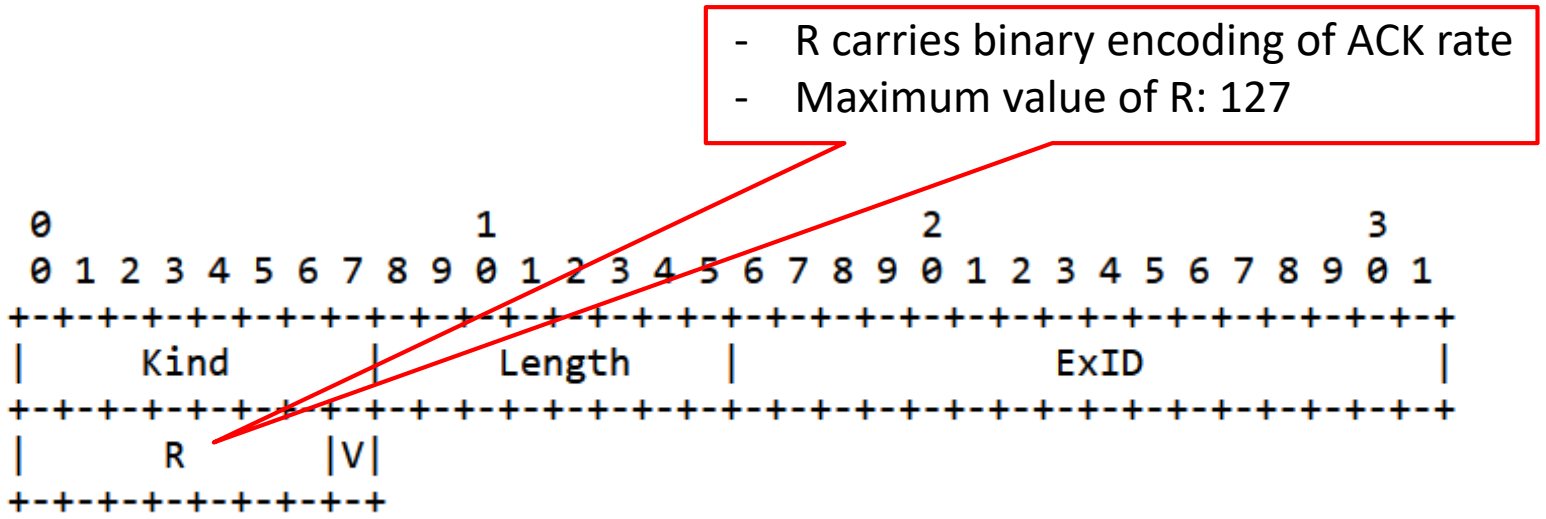
Carles Gomez
Universitat Politècnica de Catalunya
Jon Crowcroft
University of Cambridge

Intro: motivation

- Delayed ACKs
  - Intended to reduce protocol overhead
  - But may also contribute to suboptimal performance

- "Large" cwnd scenarios (i.e. cwnd >> MSS):
  - Saving more than 1 of every 2 ACKs may improve performance
- "Small" cwnd scenarios (i.e. cwnd up to ~1 MSS):
  - Delayed ACKs may incur delay, limit cwnd growth…

Intro: main TARR option format



- "R" is the ACK rate requested by the sender
  - R = 0: request an immediate ACK (but keep steady state R)

Status

- WG adoption
  - draft-ietf-tcpm-ack-rate-request-00
    - Same content as draft-gomez-tcpm-ack-rate-request-06
  - February 2023

- Version -04
  - Aims to address comments from IETF 118

Updates (I/VII)

- Clarifying what overrides TARR and what is overriden by TARR:
  - Introduction, last paragraph:
    - TARR allows to override the Delayed ACKs mechanism (RFC 1122)
      - » While complying with the max ACK delay of 500 ms
    - Standards Track TCP specifications other than RFC 1122 and some Informational specs that recommend or mandate triggering ACKs in special conditions prevail over TARR
      - » See section 3.2 and Appendix B

Updates (II/VII)

- Appendix B: other documents that provide rules on sending ACKs
  - Standards Track documents:
    - RFC 1122:
      - » A receiver MAY send an ACK in response to an out-of-order segment
    - RFC 2018:
      - » “MAY” increased to “SHOULD”, including a SACK option, for every valid segment that contains new data
    - RFC 5681 (already covered in -03, section 3.2):
      - » A receiver SHOULD send a duplicate ACK immediately when an out-of-order segment arrives
      - » A receiver SHOULD send an immediate ACK when the incoming segment fills in all or part of a gap in the sequence space

Updates (III/VII)

- Appendix B: other documents that provide rules on sending ACKs
  - Standards Track documents:
    - RFC 5961 (not covered in -03, section 3.2):
      - » If the RST bit is set and the sequence number does not exactly match the next expected sequence value, yet is within the current receive window, TCP MUST send an ACK (challenge ACK)
    - draft-ietf-tcpm-accurate-ecn (not covered in -03, section 3.2):
      - » Change-Triggered ACKs: An AccECN Data Receiver SHOULD emit an ACK whenever a data packet marked CE arrives after the previous packet was not CE
      - » Increment-Triggered ACKs: An AccECN receiver of a packet MUST emit an ACK if 'n' CE marks have arrived since the previous ACK
        - Rules for 'n' are given

Updates (IV/VII)

- Appendix B: other documents that provide rules on sending ACKs
  - Informational documents:
    - RFC 5690, AckCC:
      - » Already discussed in Appendix A
    - RFC 8257, DCTCP (not covered in -03, section 3.2):
      - » If the CE codepoint is set and DCTCP.CE is false, […] send an immediate ACK
      - » If the CE codepoint is not set and DCTCP.CE is true, […] send an immediate ACK
      - » […] MAY choose to send two ACKs: one for previously unacknowledged packets and another acknowledging the most recently received packet

Updates (V/VII)

- Appendix B: other documents that provide rules on sending ACKs
  - Experimental documents:
    - RFC 4782, Quick-Start:
      - » May produce a sudden increase of pure ACKs
      - » In the absence of ACK CC, the TCP receiver could limit its sending rate for ACKs sent in response to Quick-Start data packets
        - Formula is given
      - » TARR can be used to allow a Quick-Start sender to request the ACK rate to be used by the receiver

Updates (VI/VII)

- Section 3.2: “Receiver behavior”
  - Updated to clearly enforce the rules regarding ACK generation from:
    - RFC 5961
    - RFC 8257
    - draft-ietf-tcpm-accurate-ecn
  - The receivers's count of data segments received from the sender is reset every time that an ACK is sent for any reason

Updates (VII/VII)

- The rwin size may have changed from the last rwin size known by the sender:
  - Section 3.1 (sender):
    - If rwin size has increased, the sender will not request an R value corresponding to an amount of data bytes to be acknowledged at once greater than the current rwin size
    - If rwin size has decreased, a request of an R value corresponding to an amount of data bytes to be acknowledged at once greater than the current rwin size will be ignored by the receiver
  - Section 3.2 (receiver):
    - If the rwin size decreases to a value lower than the amount of data bytes to be acknowledged at once for the latest R requested, the amount of data bytes acknowledged at once by an ACK sent by the receiving TCP MUST NOT exceed its current rwin size.

Thanks!
Questions? Comments?

Carles Gomez
Universitat Politècnica de Catalunya
Jon Crowcroft
University of Cambridge

IETF 119 Brisbane, TCPM WG, March 2024