Sender Control of Delayed ACKs in TCP: Problem Statement, Requirements and Analysis of Potential Solutions

draft-gomez-tcpm-delack-suppr-reqs-01

Carles Gomez

Universitat Politècnica de Catalunya

Jon Crowcroft

University of Cambridge

Interim TCPM WG meeting, 29 April 2020

Status

- Related document: draft-gomez-tcpm-ack-pull
  - Avoid Delayed ACKs issues
  - Enable a sender to trigger immediate ACKs from a receiver
  - Defines a new AKP flag
  - -00 presented at IETF 105 (Montreal)
  - -01 presented at IETF 106 (Singapore)
- Discussion led to working first on **requirements**, rather than solutions
- draft-gomez-tcpm-delack-suppr-reqs-00
  - 7th March
  - Title: "Delayed ACKs suppression: problem statement, requirements and analysis of potential solutions"
- Revision -01
  - 24th March
  - Title: "Sender control of Delayed ACKs: problem statement, requirements and analysis of potential solutions"

Introduction

- Delayed ACKs: intended to reduce protocol overhead
- Delayed ACKs may be detrimental
  - Segment carrying a message of up to 1 MSS, no app-layer response, 2nd data segment not sent earlier than Delayed ACK timer
  - ACK unnecessarily delayed, negative consequences
- A sender may want to override/restore use of Delayed ACKs at a receiver
- This document:
  - Issues due to Delayed ACKs
  - Requirements for a potential solution
  - Analysis of potential solutions (based on the requirements)

Issues due to Delayed ACKs (I)

- Slow Start
  - cwnd grows by up to SMSS per ACK covering new data
  - Delayed ACKs reduces number of ACKs received by the sender, reducing the rate of cwnd growth
    - Transfer time increase, throughput decrease
    - ABC (RFC 3465) not fully included in RFC 5681
  - Delayed ACKs precludes sender behaviors for fast, non-intrusive capacity probing (e.g. chirping)
- High bit rate environments, short segments
  - A sender that uses Nagle, may be prevented from sending more data while awaiting a delayed ACK
    - High underperformance in high bit rate environments (e.g. DNS stateful operations, RFC 8490)

Issues due to Delayed ACKs (II)

- IoT scenarios
  - Memory resources cannot be released until ACK arrival
  - Increased energy consumption
  - Delay might be exacerbated (in some L2 technologies)
- Beyond classic ACK transmission behavior
  - E.g. congestion control for ACKs (RFC 5690)
  - Path asymmetric capacity: ACK arrival rate limits forward path performance
    - Some technologies (DOCSIS, mobile cellular…) apply ACK thinning

Requirements for sender control of Delayed ACKs (I)

- Sender-triggered mechanism
  - Assumption: the sender knows when Del. ACKs should be overriden
    - Sender's own traffic pattern
    - Expectation of application-layer responses
- Per-segment granularity
  - Instead of per-device or per-connection granularity
- Header/Message overhead
  - As the identified problems are about low performance
- Support for enabling generic ACK ratios
  - Would allow to address all the identified issues

Requirements for sender control of Delayed ACKs (II)

- Middlebox traversal
- Safe return to normal Delayed ACKs operation
- Impact on existing TCP functionality
- Impact on future TCP development
- Avoidance of 'hacks'
  - Workarounds may be suboptimal regarding implementation cleanliness
  - May entail other performance issues
- Who is in control?
  - Range of possibilities if the receiver cannot honor the behavior desired by the sender

Potential solutions (I)

- ACK CC (RFC 5690)
  - The sender tells the receiver the ACK ratio R to be used
  - 2-byte “TCP ACK Congestion Control Permitted” option
  - 3-byte "ACK ratio TCP” option
  - Middlebox traversal of new TCP options often regarded as ‘bad’
- TLP
  - Additional ACKs by sending a segment after Probe TimeOut (PTO)
  - Significant overhead
- ACK Pull (AKP) flag
  - No overhead, but uses reserved TCP header bits

Potential solutions (II)

- New ‘ACK Pull’ option
  - Same semantics as the AKP flag
  - Middlebox traversal of new TCP options often regarded as ‘bad’
- Reuse of existing TCP header fields
  - E.g. use 3 of the URG pointer bits as an ACK ratio exponent (for URG=0)
  - Semantics become overloaded
    - Both original field functionality and Sender Control of Delayed ACKs not always fully available
- ‘Hacks’
  - Sending a previously ACKed byte, ‘split hack’ (Contiki OS)…

Summary: solutions vs requirements

	Per-segment	Over-head	Generic ACK rat	Middle-box tr.	Impact current	Impact future	Hack Avoid.
ACKcc	Yes	Low	Yes	Bad?	No	Low	Yes
TLP	No	High	No	Good	No	No	Yes
AKP flag	Yes	No	Yes	?	No	Med/Hi	Yes
AKP option	Yes	Low	Yes	Bad?	No	Low	Yes
Reuse fields	Yes	No	Yes	Good?	Yes	?	Yes
Hacks	?	Med/Hig	No	Good?	No	No	No

- No ideal solution appears to exist…

Ready for WG adoption?

Carles Gomez
Universitat Politècnica de Catalunya
Jon Crowcroft
University of Cambridge

Interim TCPM WG meeting, 29 April 2020