More Accurate ECN Feedback in TCP

draft-ietf-tcpm-accurate-ecn-15

Bob Briscoe, Independent
Mirja Kühlewind, Ericsson
Richard Scheffenegger, NetApp

IETF-111 Jul 2021
Solution (recap)
Congestion extent, not just existence

- AccECN: Change to TCP wire protocol
  - Repeated count of CE packets (ACE) - essential
  - and CE bytes (AccECN Option) – supplementary

- Key to congestion control for low queuing delay
  - 0.5 ms (vs. 5-15 ms) over public Internet
Implementation & Testing

- Linux v5.10 AccECN implementation [Ilpo]
  - [https://github.com/L4STeam/linux](https://github.com/L4STeam/linux)
  - now works with all Congestion Control (CC) modules (Cubic, BBRv2, Prague, DCTCP, Reno, …)
  - enables A-B testing of CCs with consistent feedback code

- Also minimalist FreeBSD implementation (w/o TCP Option) of draft-09++ [RScheff]
AccECN and ACK Filtering

• Extensive testbed evaluation [Ilpo]. Ongoing
  • Wide area testing also needed

• Interim results compare the 4 degrees of feedback support shown

• Various traffic scenarios and AQMs
  • on-off (step) and spaced (probabilistic) ECN markings

• Built/building models of ACK filters
  • focus on worst-cases – up to 1/34 packets ACK’d, and not 'TCP-smart'
  • modelling closed source boxes is challenging

• Built/building simple heuristics for AccECN to fill the gaps

<table>
<thead>
<tr>
<th></th>
<th>AccECN TCP Option (3*24b)</th>
<th>ACE (3b)</th>
<th>DCTCP f/b (1b)</th>
<th>With ACK filtering</th>
</tr>
</thead>
<tbody>
<tr>
<td>alwaysopt</td>
<td>Y</td>
<td>Y</td>
<td>-</td>
<td>As good as without filtering</td>
</tr>
<tr>
<td>minopt</td>
<td>Minimum</td>
<td>Y</td>
<td>-</td>
<td>As good as without filtering</td>
</tr>
<tr>
<td>noopt</td>
<td>-</td>
<td>Y</td>
<td>-</td>
<td>Usually good, some poor scenarios</td>
</tr>
<tr>
<td>dctcpfb</td>
<td>-</td>
<td>-</td>
<td>Y</td>
<td>Sometimes good, unpredictable</td>
</tr>
</tbody>
</table>
How often to ACK ACKs?

An AccECN Data Receiver:
- SHOULD emit an ACK whenever a data packet marked CE arrives after the previous packet was not CE.
- MUST emit an ACK once 'n' CE marks have arrived since the previous ACK.

<table>
<thead>
<tr>
<th>n</th>
<th>min</th>
<th>SHOULD</th>
<th>max</th>
</tr>
</thead>
<tbody>
<tr>
<td>if newly delivered data to ack</td>
<td>-</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>if no newly delivered data to ack</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
</tbody>
</table>

• Intentions:
  - rapid feedback at congestion onset
  - reduce risk of double wrap of 3-bit ACE counter

• 2nd bullet could lead to ACKs of ACKs (first bullet deliberately doesn't)
  - 'OK in principle': ACKing new information (new CE marks)
  - to maintain cwnd during idles, or ready for adding ACK CC
  - 'n' no less than 3 to strongly damp potential ACK ping-pong
Distinguishing ACKs of ACKs from DupACKs

- ACKs of ACKs could be DupACKs [Yoshi]
  - if ACK stream CE marked
  - and data volleys take turns

- SHOULD implement extra DupACK test
  - not a DupACK if either is true:
    - SACK negotiated, but no SACK on the ACK
    - or using timestamps,
      and ACK sent before oldest unack'd data

- If extra test is not implemented
  - SHOULD NOT send ECN-capable pure ACKs
How often to ACK during a burst?

• If arrivals are processed as one burst (e.g. LRO/GRO)

• Does Receiver emit back-to-back ACKs? [Neal]
  1) every 'n' CE marks,
  2) every transition to CE?

• Guideline added:
  • both rules SHOULD be interpreted as requiring multiple ACKs to be emitted back-to-back (v similar to DCTCP)
  • If performance-critical, can emit one ACK at the end
AccECN TCP Option
Simplified Usage Rules

<table>
<thead>
<tr>
<th>kind0</th>
<th>length</th>
<th>EE0B</th>
<th>[ECEB</th>
<th>[EE1B]</th>
</tr>
</thead>
<tbody>
<tr>
<td>kind1</td>
<td>length</td>
<td>EE1B</td>
<td>[ECEB</td>
<td>[EE0B]</td>
</tr>
</tbody>
</table>

• When to include an Option (if using the Option at all)?

• SHOULD include option on every ack of new data
  • SHOULD include any counter field that ever changed

• Previously just guidance; now 2 SHOULDs, because:
  • You don't know which ACKs will survive ACK filtering
  • So, accurate and simple to include the Option on them all

• Removed requirements:
  • to beacon all fields
  • to emit an ACK when any byte counter changes (left in by accident after earlier edits)
**Status & Next Steps**
draft-ietf-tcpm-accurate-ecn-15

- Recent text tweaking
  - e.g. Transparent Middleboxes / TCP Normalizers [Gorry]
  - checked MUSTs with the eyes of a minimalist implementation

- Ready for WGLC, except
  - Waiting for SECDIR review
  - ACK filter testing ongoing

- draft-ietf-tcpm-generalized-ecn (EXP) dependent on this

- April’20 tcpm interim:
  - WG resolved to wait a while for L4S, but go ahead soon if still waiting
AccECN

Q&A

spare slides
Problem (Recap)
Congestion Existence, not Extent

- Explicit Congestion Notification (ECN)
  - routers/switches mark more packets as load grows
  - RFC3168 added ECN to IP and TCP

<table>
<thead>
<tr>
<th>IP-ECN</th>
<th>Codepoint</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>00</td>
<td>not-ECT</td>
<td>No ECN</td>
</tr>
<tr>
<td>10</td>
<td>ECT(0)</td>
<td>ECN-Capable Transport</td>
</tr>
<tr>
<td>01</td>
<td>ECT(1)</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>CE</td>
<td>Congestion Experienced</td>
</tr>
</tbody>
</table>

- Problem with RFC3168 ECN feedback:
  - only one TCP feedback per RTT
  - rcvr repeats ECE flag for reliability, until sender's CWR flag acks it
  - suited TCP at the time – one congestion response per RTT