Robust ECN Signaling with Nonces

David Ely, Neil Spring, David Wetherall, Stefan Savage†, and Tom Anderson
University of Washington and †UC San Diego

IEEE ICNP, November, 2001
ECN gives receivers power

Old congestion signals:

- Receivers claim lost packets for reliable delivery.
- Senders slow down when retransmitting.

ECN congestion signals:

- Behaving receivers return Congestion Experienced (CE) as ECN Congestion Echo (ECE).
- Senders slow down when ECE seen.
- Greedy or buggy receivers have no incentive to set ECE.
Misbehavior is dangerous

Receivers that conceal ECN signals:

- Get better performance:  
  - Behaving: ▶   
  - Misbehaving Victims: ≤

- Reduce performance of competing Victims:  
  - Behaving: ▶   
  - Misbehaving Victims: ≤
The Challenge

Can we have both:

- The benefits of ECN, and
- Protection against misbehaving receivers?

Can we detect misbehavior?

Can we discourage misbehavior?
Outline

TODO: fix at last minute

- ECN-nonce basics
- Header bits: IP and TCP
- Walkthroughs: Normal and with misbehavior
- Endpoint state requirements
- Policy: giving the ECN-nonce teeth
- Short flows
- Conclusions
Nonces revoke receiver power

Sender attaches random “nonce” using ECT field

Sum (parity) of nonces in ACKs

Correct nonce sum depends on unmarked packets.

Receiver’s sum legitimately incorrect when:

- ECT(0) or (1) removed by CE. → expect ECE.
- Retransmission lacks ECT → have retransmitted.

Sender can detect an misbehavior: claimed sum is wrong
### ECN bits to code-points

<table>
<thead>
<tr>
<th>bits</th>
<th>RFC 2481</th>
<th>RFC 3168</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>not ECN capable</td>
<td>same</td>
</tr>
<tr>
<td>0 1</td>
<td>unused</td>
<td>ECT (1)</td>
</tr>
<tr>
<td>1 0</td>
<td>ECN-Capable Transport (ECT)</td>
<td>ECT (0)</td>
</tr>
<tr>
<td>1 1</td>
<td>Congestion Experienced</td>
<td>same</td>
</tr>
</tbody>
</table>

**Transition to CE removes original ECT(0) or ECT(1).**

Use a flag bit in the TCP header to return Nonce Sum.
ECN-Nonces with good receivers

① ECN properly echoed
② Nonce sum (NS) ignored
③ Synch. NS after CWR:
   Assume receiver is right.
ECN-Nonces with a misbehaving receiver

1. CE improperly hidden.
2. Guessed NS is wrong.
3. Sender disables ECN.
State requirements

Senders store:

- Expected nonce sum for ack of each packet in retransmission buffer.
- The sequence number of the last CWR sent.
- A bit set when the expected nonce sum is wrong on Ack-of-CWR.

Receivers store:

- Nonce Sum to send in ACKs.
- Nonces of unacknowledged packets.
Making detection sufficient (policy)

- Encourage nonce support
  - Preferred treatment
  - Non-ECN Optimizations

- Discourage misbehavior when detected:
  - Stop sending ECT
  - Reduce cwnd, ssthresh to 1.
  - Alternatives: RST, limit send window, blacklist
Why cwnd = ssthresh = 1?

Shows long-term behavior: sender continues to set ECT.

cwnd /= 2 left an advantage that cwnd = 1 removed.

Stop sending ECT to be sure
Misbehaving short flows

In-sequence data rate vs. flow length

Misbehavior increasingly effective

ECN converges

ECN-nonce detects misbehavior quickly

ECN disabled

ECN-nonce is effective: \( \leq \), and \( \ll \)
Conclusions

ECN-Nonce provides on-line detection of misbehavior.

- $1 \frac{1}{2}$ header bits
- Cheap per-packet processing

Reacting to misbehavior can remove performance advantage:

- cwnd=1 makes 1-bit nonce sufficient
- Even for short flows.