A Survey of Recent Developments of TCP

Sally Floyd
ACIRI
(AT&T Center for Internet Research at ICSI)
October 17, 2001

IEEE Annual Computer Communications Workshop

An overview of this session:

- This talk: A Survey of Recent Developments of TCP.
  - And an introduction to DCP, a newly-proposed transport protocol.
- Neil Spring: Robust ECN Signaling with Nonces.
  - Adding ECN Nonces to TCP.
- Srini Seshan: The Congestion Manager.
  - Sharing congestion control state below the transport layer.
- Randall Stewart: An Overview of SCTP.
  - A reliable transport protocol accommodating multiple streams.

An outline of this talk:

Current and future developments for TCP:

- The Limited Transmit mechanism in TCP.

- Using D-SACK to detect unnecessary Fast Retransmits,
or unnecessary Retransmit Timeouts.

- The addition of ECN (Explicit Congestion Notification) to TCP.

- Corruption Notification?

- Congestion control mechanisms with very large congestion windows?

Other developments:

- DCP: Datagram Control Protocol.
  - A proposed unreliable transport protocol.

Themes:

- Bringing TCP closer to a model of pure AIMD (Additive Increase, Multiplicative Decrease) and exponential backoff of the retransmit timer.

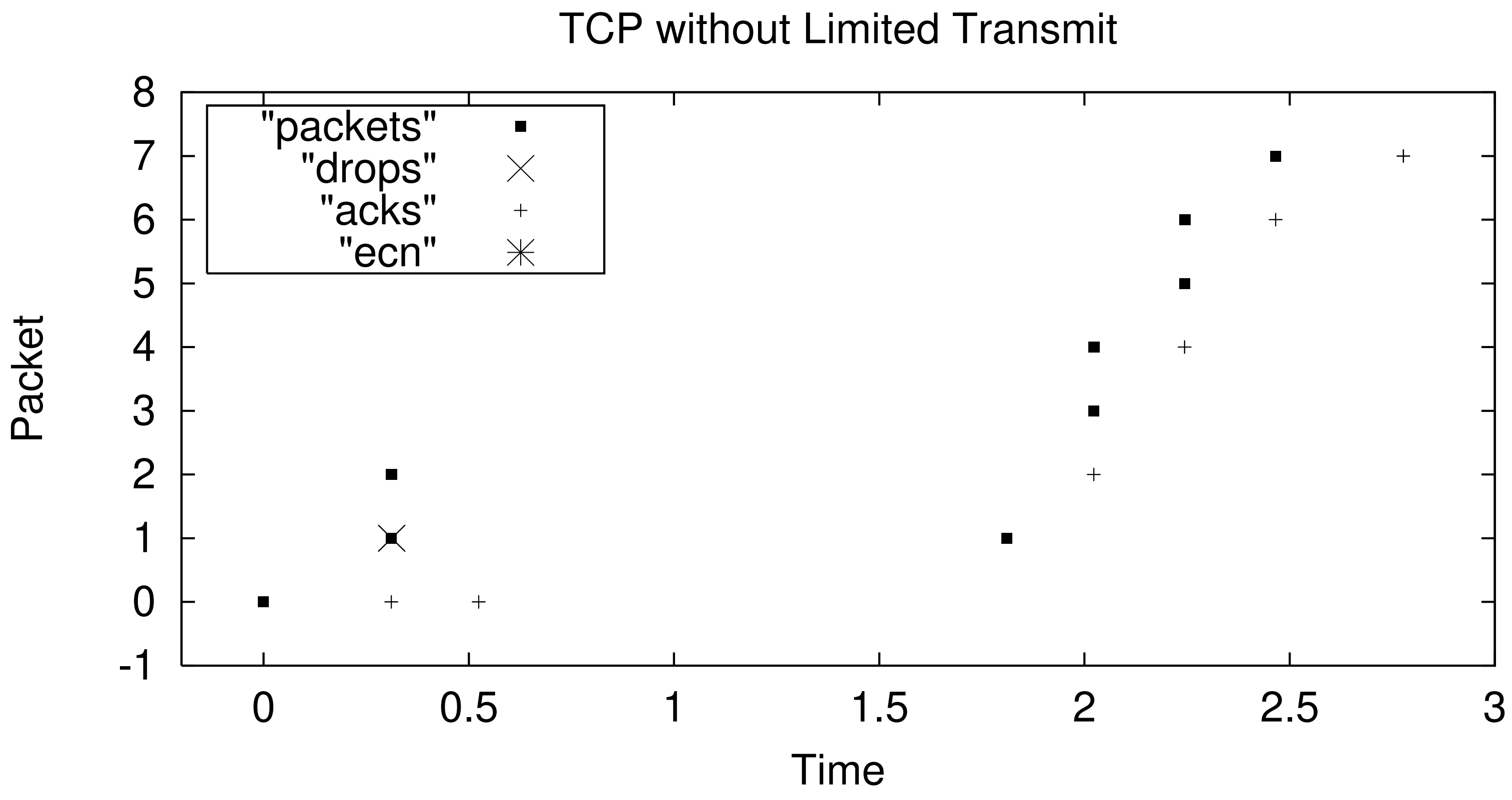
- Making the indication of congestion more explicit.

- Modifying TCP's AIMD mechanisms at very high congestion windows?

- Other transport protocols for other purposes:
  - Other than reliable, in-order delivery.
  - Other than AIMD congestion control.
  - (For this talk, we only consider unicast transport protocols.)

The Limited Transmit mechanism in TCP:

- The problem: unnecessary retransmit timeouts after a loss.

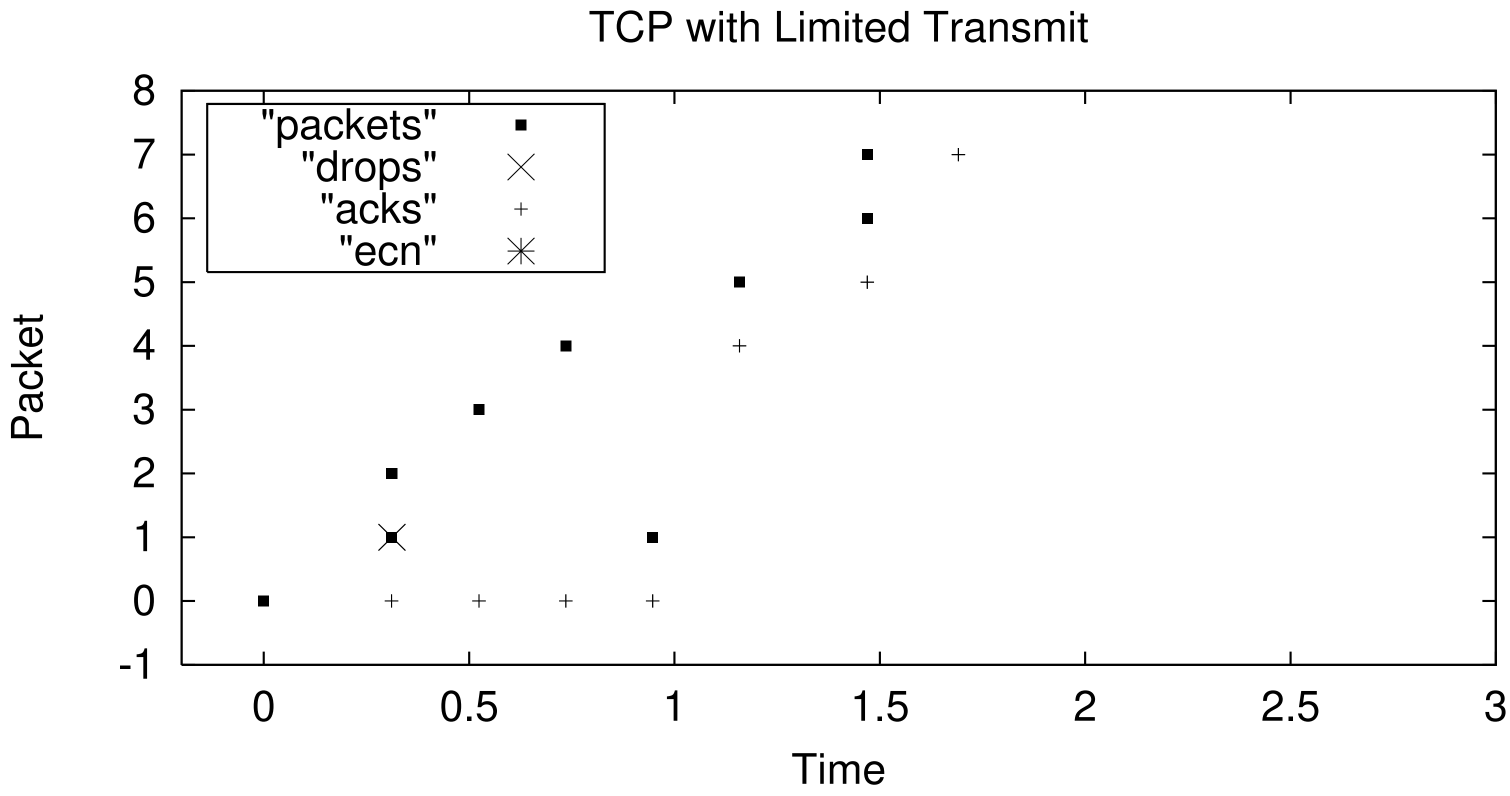


- There is no response to the duplicate acknowledgement (dup ack) at time 0.5.

The Limited Transmit mechanism in TCP:

- The solution: Limited Transmit, RFC 3042.

Send a new packet after the first and second dup acks.



- A Fast Retransmit can be more effective than a Retransmit Timeout.
  - Limited Transmit is useful for TCPs with small congestion windows.

DSACK: Detecting unnecessary Fast Retransmits or Retransmit Timeouts

- The problem: TCP is not robust to reordered or delayed packets.
  - More that three reordered packets are interpreted as a loss.
  - A long delay is interpreted as a loss.
  - After a loss, a packet is retransmitted, and the window is halved.

- The solution: make TCP robust to reordered or delayed packets.
  - Part 1: Detect that the packet was delayed or reordered, not lost.
  - Part 2: “Undo” the unnecessary halving of the window.
  - Part 3: Modify the Dup Ack Threshold or Timeout value as appropriate.

Part 1: Detecting that the packet was not lost.

- Use the D-SACK extension to SACK TCP for the receiver to inform the sender of all duplicate packets received.
  - If the sender retransmitted a packet, and the receiver received two copies of that packet, the sender can infer that the original packet was not lost. [RFC 2883], [Blanton01]

- Or, use timestamps.

- Or, define a new “RTX” bit in TCP packet headers to indicate retransmitted packets, and to report the receipt of retransmitted packets. [Ludwig01]

Part 2: Undoing the unnecessary halving of the congestion window.

- Set the slow start threshold (ssthresh) to the window's value before the halving.

- Slow-start back up to the old window.

Part 3: Modify the Dup Ack Threshold or Timeout value as appropriate.

Goal: Most of the Fast Retransmits and Retransmit Timeouts should be for legitimate lost packets.
– Without waiting unnecessarily long to retransmit a lost packet.

- Adjust the Dup Ack Threshold after each Fast Retransmit. [Blanton01a]
  – Or, keep track of the history of reordering in the connection, and use this to determine the Dup Ack Threshold. [Zhang, Karp, Floyd]

- Similar approaches could be used for adjusting the algorithm for setting Retransmit Timeout values.

- This is still under investigation. [Blanton01a]

The Addition of ECN to IP and to TCP:

- The problem:

Packet losses are an expensive form of congestion notification,

  - For the receiver (the packet is not received);
  - For the sender (uncertainties in inferring the congestion signal).

- The solution: Explicit Corruption Notification (ECN).
  - The router can mark a packet as an indication of congestion.
  - RFC 3168, Proposed Standard.

- Issue: Why trust the receiver to report the congestion indication?
  - Use the ECN nonce, described in the next talk.

Future Work: Corruption Notification

- The problem:

TCP assumes that losses are from congestion, not corruption.

- Even with ECN, packets can be lost from buffer overflow.
  – Losses can't be assumed to be corruption instead of congestion.

- The solution: Some form of Corruption Notification?

"The packet with this header was lost in the network due to corruption, not congestion."

- Issues:
  – Only the link-level sender knows for sure the packet header of the corrupted packet.
  – The Corruption Notification could arrive at the TCP sender after the packet was already assumed lost.
  – The Corruption Notification message could be spoofed.

Future Work: Modified congestion control for large-window TCP

- The problem:

For high windows, TCP requires a *very* low packet drop rate.
To fill a 10 Gbps pipe, a TCP with 1500-byte packets requires:

– A congestion window of 83,333 packets.
  (Assuming a 100-ms round-trip time.)
– At most one in 5,000,000,000 packets lost.

- The solution:

– Use parallel TCP connections;
– Or, modify TCP's increase and decrease parameters for high windows.

- This is research in progress. [Ratnasamy, Floyd, Shenker]

For high congestion windows, TCP's increase and decrease parameters could depend on the congestion window:

– Increasing by more than one packet per round-trip time;
– Decreasing less than halving the congestion window.

DCP: Datagram Control Protocol

- DCP: a congestion-controlled, unreliable transport protocol.
- Without in-order, reliable delivery.
- The application can specify either AIMD-based (like TCP) or equation-based (TFRC) congestion control.
- Designed to be as minimal as possible, for applications that now use UDP without end-to-end congestion control.
- This is in the initial stages [Kohler, Handley, Floyd, Padhye].

Why add DCP instead of using an unreliable version of SCTP?

- For some applications, the AIMD congestion control used by TCP and SCTP is not appropriate.

- For some applications, unreliable versions of SCTP would involve more overhead that is needed (compared to DCP).

Why add DCP when applications could use the Congestion Manager instead?

- The initial versions of the Congestion Manager (CM) require applications to have their own end-to-end feedback about packet drops and marks.

- We want to allow applications to choose between AIMD and TFRC congestion control.

- We don't want the use of end-to-end congestion control for unreliable applications to be contingent on the deployment of CM.

References:

[Blanton01] Ethan Blanton, Mark Allman, Using TCP DSACKs and SCTP Duplicate TSNs to Detect Spurious Retransmissions, draft-blanton-dsack-use-01.txt, internet-draft, work in progress, August, 2001.

[Blanton01a] Ethan Blanton, Mark Allman, Adjusting the Duplicate ACK Threshold to Avoid Spurious Retransmits, draft-blanton-dupack-thresh-adjust-00.txt, internet-draft, work in progress, July, 2001.

[DCP] DCP Web Page, http://www.aciri.org/kohler/dcp/

[ECN] Ramakrishnan, K.K., Floyd, S., and Black, D., The Addition of Explicit Congestion Notification (ECN) to IP., RFC 3168, Proposed Standard, September 2001.

[Floyd01] Floyd, S., A Report on Some Recent Developments in TCP Congestion Control. IEEE Communications Magazine, April 2001.

[Ludwig01] Reiner Ludwig, TCP Retransmit (RXT) Flag, draft-ludwig-tsvwg-tcp-rxt-flag-01.txt, internet-draft, work in progress, July, 2001.

[RFC 2883] Floyd, S., Mahdavi, J., Mathis, M., and Podolsky, M.. , An Extension to the Selective Acknowledgement (SACK) Option for TCP. RFC 2883, Proposed Standard, July 2000.

[RFC 3042] Allman, M., Balakrishnan, H., and Floyd, S., Enhancing TCP's Loss Recovery Using Limited Transmit. RFC 3042, Proposed Standard, January 2001.

Papers by Floyd are available from: http://www.aciri.org/floyd/