TCP

Matthew Roughan
AT&T Labs - Research
roughan@research.att.com

Ashok Erramilli
Qnetworx
Erramilli@qnetworx.com

Darryl Veitch
EMUlab – Ericsson and the University of Melbourne
d.veitch@ee.mu.oz.au
TCP Congestion Control

- Has four main parts
  - Slow Start (SS)
  - Congestion Avoidance (CA)
  - Fast Retransmit
  - Fast Recovery

- *ssthresh*: slow start threshold determines whether to use SS or CA

- Assume packet losses are caused by congestion
Slow Start

- Start with \( cwnd = 1 \) (slow start)
- On each successful ACK increment \( cwnd \)
  \[ cwnd \leftarrow cwnd + 1 \]
- Exponential growth of \( cwnd \)
  each RTT: \( cwnd \leftarrow 2 \times cwnd \)
- Enter \( CA \) when \( cwnd \geq ssthresh \)
Slow Start

cwnd ← cwnd + 1 (for each ACK)
Congestion Avoidance

- Starts when \( cwnd \geq ssthresh \)
- On each successful ACK:
  \[ cwnd \leftarrow cwnd + \frac{1}{cwnd} \]
- Linear growth of \( cwnd \)
  each RTT: \( cwnd \leftarrow cwnd + 1 \)
Congestion Avoidance

\[ \text{cwnd} \leftarrow \text{cwnd} + \frac{1}{\text{cwnd}} \text{ (for each ACK)} \]
Packet Loss

- **Assumption**: loss indicates congestion
- Packet loss detected by
  - Retransmission TimeOuts (RTO timer)
  - Duplicate ACKs (at least 3)

<table>
<thead>
<tr>
<th>Packets</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Acknowledgements</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>
Timeout

\[ \text{ssthresh} \leftarrow \frac{\text{cwnd}}{2} \]

\[ \text{cwnd} = 1 \]
Fast Retransmit

- Wait for a timeout is quite long
- Immediately retransmits after 3 dupACKs without waiting for timeout
- Adjusts ssthresh
  \[ \text{flightsize} = \min(\text{awnd, cwnd}) \]
  \[ \text{ssthresh} \leftarrow \max(\text{flightsize}/2, 2) \]
- Enter Slow Start \((\text{cwnd} = 1)\)
Successive Timeouts

- When there is a timeout, double the RTO
- Keep doing so for each lost retransmission
  - Exponential back-off
  - Max 64 seconds\(^1\)
  - Max 12 retransmits\(^1\)

1 - Net/3 BSD
Fast recovery

- Motivation: prevent `pipe' from emptying after fast retransmit
- Idea: each dupACK represents a packet having left the pipe (successfully received)
- Enter FR/FR after 3 dupACKs
  - Set $\text{ssthresh} \leftarrow \max(\text{flightsize}/2, 2)$
  - Retransmit lost packet
  - Set $\text{cwnd} \leftarrow \text{ssthresh} + \text{ndup}$ (window inflation)
  - Wait till $W = \min(\text{awnd}, \text{cwnd})$ is large enough;
    transmit new packet(s)
  - On non-dup ACK (1 RTT later), set $\text{cwnd} \leftarrow \text{ssthresh}$
    (window deflation)
- Enter CA
TCP Reno

Fast retransmission/fast recovery