CS118 Discussion 1D, Week 4

Zengwen Yuan
Boelter Hall 2760, Friday 4:00—5:50 p.m.
Outline

• Lecture review: Transport layer

• Project Questions?
Stop and Wait Protocol

• **Main Issue:** limited performance

• Consider two hosts that are directly connected by a 50 Kbit/sec satellite link that has a 250 milliseconds propagation delay. If these hosts send 1000 bits segments, what is the maximum throughput in stop-and-wait protocol if we ignore the transmission time of ACK?
Stop and Wait Protocol

• **Main Issue**: limited performance

• Consider two hosts that are directly connected by a 50 Kbit/sec satellite link that has a 250 milliseconds propagation delay. If these hosts send 1000 bits segments, what is the maximum throughput in stop-and-wait protocol if we ignore the transmission time of ACK?

  • \( \frac{1000}{1000/50+250+250} = 2 \text{ Kbit/sec}! \)
Pipelined Protocols

• Go-back-N: receiver only sends cumulative ACKs
  • Drop out-of-order segments
  • reACK packet with highest in-order sequence number
  • Timer for oldest unACKed packet only, retransmit all unACKed packets

• Selective repeat: receiver ACKs individual packets
  • Buffer out of order segments
  • Timer for each individual unACKed packet, retransmit any unACKed packet
Demo: Selective Repeat/Go Back N

http://www.ccs-labs.org/teaching/rn/animations/gbn_sr/
TCP

- **Point-to-point, byte-stream reliable transport protocol**
- **Multiplexing/de-multiplexing:** Source/Dest port
- **Reliable data transfer:** sequence number, ack, checksum, RTT estimation
- **Connection setup:** sequence number, SYN, receive window
- **Connection teardown:** sequence number, FIN
## Comparison of Reliable Transport Protocol

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Buffer at sender</th>
<th>Buffer at receiver</th>
<th>ACK</th>
<th>Timeout/Retransmission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stop &amp; Wait</td>
<td>No</td>
<td>No</td>
<td>No out-of-order</td>
<td>Retransmit timeout packet</td>
</tr>
<tr>
<td>Go-Back-N</td>
<td>Yes</td>
<td>No</td>
<td>Accumulative Seq#</td>
<td>Retransmit all packets in window</td>
</tr>
<tr>
<td>Selective Repeat</td>
<td>Yes</td>
<td>Yes</td>
<td>Received Seq#</td>
<td>Retransmit timeout packet</td>
</tr>
<tr>
<td>TCP</td>
<td>Yes</td>
<td>Yes</td>
<td>Next expected Seq#</td>
<td>Retransmit timeout packet</td>
</tr>
</tbody>
</table>
TCP: overview

- The Transmission Control Protocol (TCP), documented in RFC 793, makes up for IP's deficiencies by providing reliable, stream-oriented connections that hide most of IP's shortcomings. The protocol suite gets its name because most TCP/IP protocols are based on TCP, which is in turn based on IP. TCP and IP are the twin pillars of TCP/IP.
TCP: key functionalities

- TCP adds a great deal of functionality to the IP service it is layered over:
  - **Streams.** TCP data is organized as a stream of bytes, much like a file. The datagram nature of the network is concealed. A mechanism (the Urgent Pointer) exists to let out-of-band data be specially flagged.
  
  - **Reliable delivery.** Sequence numbers are used to coordinate which data has been transmitted and received. TCP will arrange for retransmission if it determines that data has been lost.
  
  - **Network adaptation.** TCP will dynamically learn the delay characteristics of a network and adjust its operation to maximize throughput without overloading the network.
  
  - **Flow control.** TCP manages data buffers, and coordinates traffic so its buffers will never overflow. Fast senders will be stopped periodically to keep up with slower receivers.
TCP: header format

Note that one tick mark represents one bit position.
TCP: connection setup

- Connection setup: three-way handshaking
  - 1st round: SYN+initial sequence number
  - 2nd round: SYN+SYNACK+server’s initial sequence number
  - 3rd round: SYNACK+(optional) data
TCP: connection setup

**Figure 3.39** TCP three-way handshake: segment exchange
TCP: connection teardown

- Normal termination
  - allow unilateral close
  - avoid sequence number overlapping
- TCP must continue to receive data even after closing
  - “Half-open” connection: cannot close connection immediately: what if a new connection restarts and uses same sequence number?
TCP: connection teardown


diagram showing TCP connection teardown process

Two generals' problem

Figure 3.40 ♦ Closing a TCP connection
TCP: flow control

• Limits the rate a sender transfers data

• Avoid having the sender send data too fast

• Avoid exceeding the capacity of the receiver to process data

• Receiver specify the receive window

• The window size announce the number of bytes still free in the receiver buffer
TCP: timeout

- Estimated RTT = \((1 - \alpha) \times SRTT + \alpha \times \text{SampleRTT}\)

- DevRTT = \((1 - \beta) \times \text{DevRTT} + \beta \times |\text{SampleRTT} - \text{Estimated RTT}|\)

- Retransmission Timer (RTO) = Estimated RTT + 4 \times \text{DevRTT}
TCP: timeout

• Karn’s algorithm — in case of retransmission
  • do not take the RTT sample (i.e. do not update SRTT or DevRTT)
  • double the retransmission timer value (RTO) after each timeout
  • Take RTT measure again upon next data transmission (that did not get retransmitted)
References

• MIT 18.996: Topic in TCS: Internet Research Problems

• Princeton ELE539A: Optimization of Communication Systems

• http://www.freesoft.org/CIE/Course/Section4/