Chapter 5 End-to-End Protocols

The Transmission Control Protocol (TCP)

- TCP offers a reliable, connection-oriented, byte-stream service
  - Reliable, in-order delivery of a stream of bytes
  - Full duplex operation
  - Includes a flow control mechanism that keeps the sender from over-running the receiver
  - Implements a congestion control mechanism that keeps the sender from overloading the network
- TCP uses the sliding window algorithm on an end-to-end basis to provide reliable and ordered delivery. However, because TCP runs over the Internet rather than a point-to-point link, there are many important differences.
- End-to-end issues
  - TCP supports logical connections between processes running on any two computers in the Internet
    - Need explicit connection establishment and teardown
  - TCP connections may have widely different RTTs, and RTT may vary during a single TCP connection
    - Need adaptive timeout mechanism
  - Potentially long delay in the network
    - Need to be prepared for very old packets to suddenly show up at the receiver, potentially confusing the sliding window algorithm
  - Potentially different capacity at destination host
    - Each side needs to learn how much buffer space the other side can allocate to the connection (i.e., flow control)
  - Network is shared by many hosts
    - Need to be prepared for network congestion
TCP is a byte-oriented protocol: the sender writes bytes into a TCP connection and the receiver reads bytes out of the TCP connection
  o TCP on the source host buffers enough bytes from the sending process to fill a reasonably sized packet and then sends this packet to its peer on the destination host
  o TCP on the destination host then empties the contents of the packet into a receive buffer, and the receiving process reads from this buffer at its leisure
  o The packets exchanged between TCP peers are called *segments*

TCP segment format
  o SrcPort/DstPort identify the source/destination port
    ▪ A TCP connection is uniquely identified by the 4-tuple <srcPort, SrcIPAddr, DstPort, DstIPAddr>
  o SequenceNum: the sequence number for the first byte of data carried in the segment
    ▪ Each byte of data has a sequence number
  o Acknowledgement: the next sequence number expected
  o AdvertisedWindow: number of bytes, beginning with the sequence number indicated in the Acknowledgement field, that the receiver is able to accept
  o HdrLen: length of the header in 32-bit words
  o Flags
    ▪ SYN: used in connection establishment
    ▪ FIN: used in connection termination
    ▪ RESET: used when one side wants to abort the connection
    ▪ ACK: set when the Acknowledgement field is valid
    ▪ URG: indicate that this segment contains urgent data
      ▪ Urgent data is contained at the front of segment body, before the nonurgent data
      ▪ UrgPtr indicates the number of bytes in urgent data
    ▪ PUSH: indicates that the sending process wants TCP to send whatever bytes it had collected to its peer
o Checksum: computed over the entire TCP segment and the pseudoheader
  ▪ The pseudoheader consists of source IP address, destination IP address, and protocol fields from the IP header plus a TCP length field (length of the TCP header and data measured in bytes)
  ▪ Required in both IPv4 and IPv6.
o Options: up to 40 bytes, attached after the mandatory fields