Announcements

- Reading
  - Today: Chapter 6 (6.1 & 6.2)
- Midterms : Requests for re-grade due by 10/19
- HW #1 (Due Tuesday 10/19)
  - Tanenbaum: 3-5, 3-25, 5-11, 5-16, 5-24

Transport Layer

- Goal: provide error free end-to-end delivery of data
  - provide in-order delivery over unreliable network layer
- Issues:
  - checking packet integrity
  - re-transmission of lost of corrupt packets
  - connection establishment and management
  - addresses
    - need to define a host plus process
    - typical abstraction is <host, port>
  - byte vs. packet transport service
    - byte service
      - bytes are in order, but packet boundaries are lost
      - used by TCP
    - packet service
      - preserve packet boundaries

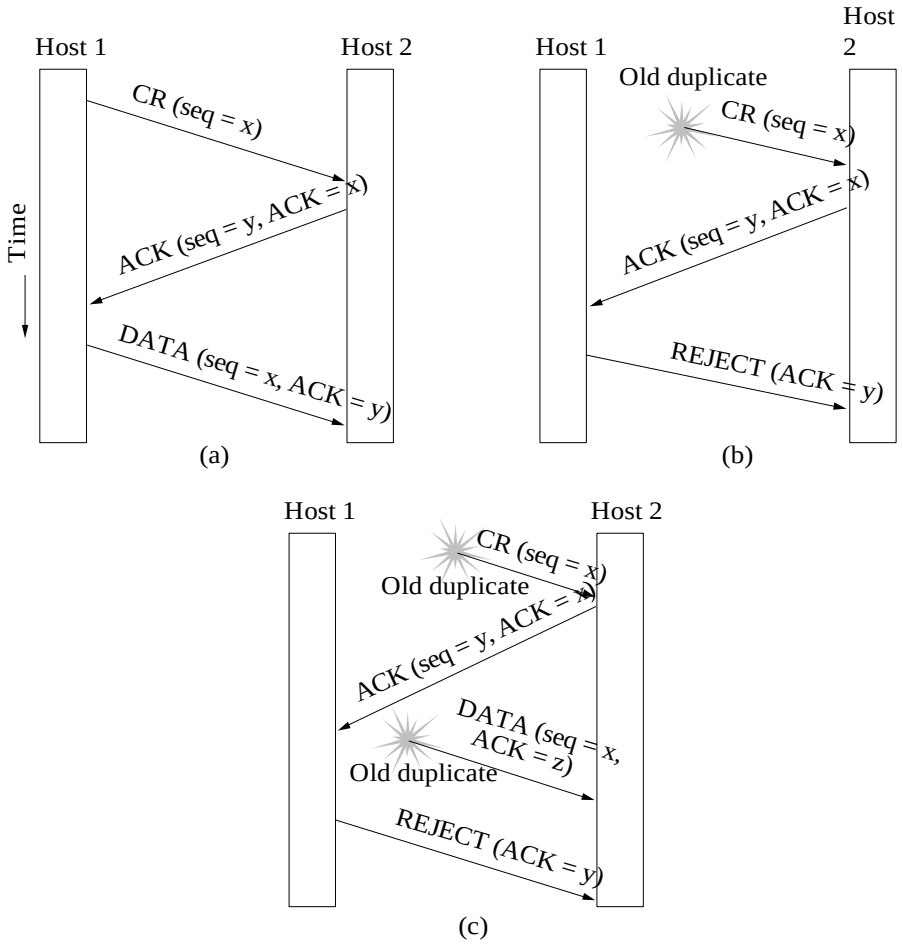
Duplicate Packets

- Issue: packets can be lost or duplicated
  - need to detect duplicates
  - need to re-send lost packets
    - but how do we know they are not just delayed?
- Solution 1
  - use a sequence number
    - each new packet uses a new sequence number
    - can detect arrival of stale packets
  - problem: when node crashes, sequence number resets
- Solution 2
  - use a clock for the sequence number
    - clocks don't reset on reboot, so we never lose sequence #
  - use a max lifetime for a packet
    - permits clocks to roll over
  - can get into **forbidden** region

Three-way Handshake

- Use different sequence number spaces for each direction
- Three messages used
  - Connection Request
    - send initial sequence number from caller to callee
  - Connection Request Acknowledgment
    - send ACK of initial sequence number from caller to callee
    - send initial sequence number from callee to caller
  - First Data TPDU
    - send ACK of initial sequence number from callee to caller
- Each Side Selects an initial number
  - it knows that the number is not currently valid
    - uses time of day
    - limits number of connects per unit time, but not data!

Example of Three-way Handshake

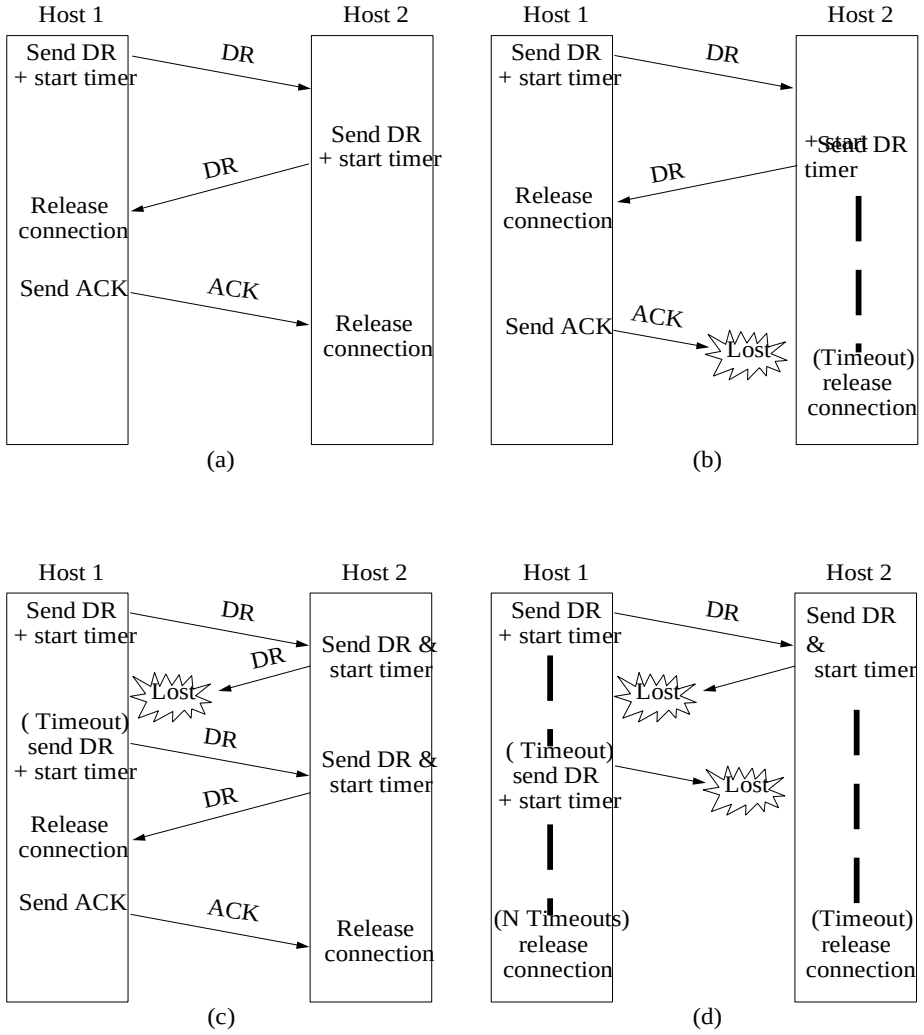


From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

Closing a Connection

- To prevent data loss,
  - both sides must agree they are done
- Problem: how to agree
  - possible that “I am done” messages will get lost
  - possible that “I ACK you are done” messages will get lost
- Solution:
  - initiator sends Disconnect Request, start DR timer
  - when initiated party receives DR, send DR and start DR timer
  - when initiator gets DR back, send ACK and release connection
  - when initiated gets ACK, release connection
  - if initiator times out, send new DR
  - if initiated times out, release connection

Connection Close Example



From: *Computer Networks*, 3rd Ed. by Andrew S. Tanenbaum, (c)1996 Prentice Hall.

Lingering Half-Duplex Connections

- If a party (or a link) dies
  - can be left with dead connections
- Solution: use keep-alive packets
  - every n seconds, send a packet
  - if no packet is received after n * m seconds, cleanup

Buffer Management

- Unreliable Network
  - sender must buffer all un-acked packets
  - receiver can buffer if space is available
    - if not, drop packet and wait to re-transmission
- Buffer Size
  - does one size fit all?
    - are TPDUs of uniform size?
  - might use a fixed size buffer smaller than max TPDU
    - requires support for multiple buffers per TPDU
- Possible to decouple buffer allocation from window
  - ACKs contain both buffer credits and ACKSs
- Buffer Copies
  - possible for each layer to copy the buffer, but this is slow
  - handoff pointers to data, but requires coordination between layers

Multiplexing in the Transport Layer

- Upward multiplexing
  - putting multiple transport connections onto one network connection
  - used to accommodate pricing strategies that charge for connections
- Downward multiplexing
  - using several network connections per transport connection
  - permits use of multiple copies of network resources
    - if the network layer uses sliding windows
      - a high latency network may under utilize the link
      - multiple connections each get a window
    - per connection buffer allocation
      - get more buffers
    - round-robin scheduling
      - get a larger share of link bandwidth

Crash Recovery

- Router or Link Crashes
  - Data in transit can be lost.
  - End nodes have sufficient state to recover lost data.
  - Transport protocol can hide network failures from the application.
- Host Crashes
  - Transport level state will be lost at one end.
  - Does the transport layer have sufficient info to recover?, **No!.**
    - Information must flow down to network and up to transport user
      - ACKs go down, and data goes up.
      - It is not possible to make these two operations atomic.
    - lack of stable storage causes this problem
  - Result, higher up layer must deal with host crashes