Chapter 5
TCP Control Flow

Networking
CS 3470, Section 1
TCP Control Flow

- TCP connections include
  - 3-way handshake to start connection
  - Transfer of data with ACKs
  - Connection tear-down

- **SequenceNum** field contains the sequence number for the first byte of data carried in segment
TCP Connection Establishment

- Both the client and server have to open a connection
- Server must start first to be able to accept incoming connections
  - Performs a *passive open*
- Client can then connect to the server
  - Performs an *active open*
TCP Connection Establishment

- Both parties (client and server) must agree on the starting sequence numbers
  - Each side has its own set of sequence numbers that start at some random number

- Sequence numbers are exchanged during the TCP three-way handshake algorithm, which establishes a TCP connection between two hosts
TCP Three-Way Handshake Algorithm

- Acknowledgment field identifies the “next sequence number expected”, implicitly acknowledging all earlier sequence numbers.
TCP State-Transition Diagram

- Some arcs are not shown
  - Most of states that involve sending segments also schedule a timeout for the ACK response
  - If ACK not received, retransmit
  - If after several tries the ACK does not arrive, TCP gives up and returns to CLOSED state
When closing connection

- Connection in TIME_WAIT state cannot move to CLOSED until it has waited for two times max time of IP TTL (120 seconds)
- Prevents delayed FINs from accidentally tearing down new connections
Introduction to Project 4

- Model TCP state-transition diagram in your own program
- Will help you make sense of network traffic for last project
Closing a Connection

- Reaching agreement: two approaches
  - Abort: send close msg to peer, delete state info
    - What if close() message lost?
  - Graceful: send close msg, but before deleting state
    - Wait for peer to acknowledge close()

- Problem solved?
  - Can I decide to close, knowing that
    - Other entity also agreed to close and knows that I will close

- Can two armies coordinate their attacks
  - If communication is unreliable?
The Byzantine Generals’ Problem

- Two generals are on the tops of two mountain.
The Byzantine Generals’ Problem

- They communicate only through messengers (pigeon network?)…
The Byzantine Generals’ Problem

- They need to coordinate the attack…
The Byzantine Generals’ Problem

- If they attack at the same time, they win…
The Byzantine Generals’ Problem

- If they attack at different times, they will die
The Byzantine Generals’ Problem

- Question: can they guarantee a synchronized attack?
The Byzantine Generals’ Problem

11 am OK?
The Byzantine Generals’ Problem

Sounds good.
The Byzantine Generals’ Problem

I got your “Sounds good.”
The Byzantine Generals’ Problem

- What if the last message is lost?

great!
The Byzantine Generals’ Problem

- What if the last message is lost?

Does the other general know I got his “Sounds good” message?
great!
The Byzantine Generals’ Problem

- Over an unreliable network, we cannot guarantee network synchronization
The Byzantine Generals’ Problem

- It can be proven that no protocol exists that works to solve this problem.

- Three-way handshake will not work, so does not four-way handshake.

- Three-way handshake is normally used to release connections.
Connection Release

Four protocol scenarios for releasing a connection. (a) Normal case of a three-way handshake. (b) final ACK lost.
Connection Release

(c) Response lost. (d) Response lost and subsequent DRs lost.