The TCP Project

Important Handouts:

- **Minet**
  - The Minet Technical Report
  - The Minet Socket Interface
- **TCP Guides**
  - RFC 793 (in convenient HTML format), RFC 1122
  - A very useful summary and picture of the TCP State diagram
  - TCP, UDP and IP pocket guide includes header details
  - Brief overview of TCP contains a nice summary of the essentials
  - Here is a page with nice TCP animations. They explain TCP connection startup, termination, data flow and flow control and cumulative ack concepts. Please see animations 20_1 to 20_5.
TCP Lingo

- When a client requests a connection, it sends a “SYN” segment (a special TCP segment) to the server port.
- SYN stands for synchronize. The SYN message includes the client’s ISN.
- ISN is Initial Sequence Number.
• Every TCP segment includes a Sequence Number that refers to the first byte of data included in the segment.
• Every TCP segment includes a Request Number (Acknowledgement Number) that indicates the byte number of the next data that is expected to be received.
  – All bytes up through this number have already been received.
And more...

• There are a bunch of control flags:
  – URG: urgent data included.
  – ACK: this segment is (among other things) an acknowledgement.
  – RST: error - abort the session.
  – SYN: synchronize Sequence Numbers (setup)
  – FIN: polite connection termination.
And more...

- MSS: Maximum segment size (A TCP option)
- Window: Every ACK includes a Window field that tells the sender how many bytes it can send before the receiver will have to toss it away (due to fixed buffer size).
TCP Connection Creation

• A server accepts a connection.
  – Must be looking for new connections!
• A client requests a connection.
  – Must know where the server is!
Client Starts

- A client starts by sending a SYN segment with the following information:
  - Client’s ISN (generated pseudo-randomly)
  - Maximum Receive Window for client.
  - Optionally (but usually) MSS (largest datagram accepted).
  - No payload! (Only TCP headers)
Server Response

• When a waiting server sees a new connection request, the server sends back a SYN segment with:
  – Server’s ISN (generated pseudo-randomly)
  – Request Number is Client ISN+1
  – Maximum Receive Window for server.
  – Optionally (but usually) MSS
  – No payload! (Only TCP headers)
Finally

• When the Server’s SYN is received, the client sends back an ACK with:
  – Request Number is Server’s ISN+1
Why 3-Way?

• Why is the third message necessary?

• HINTS:
  – TCP is a reliable service.
  – IP delivers each TCP segment.
  – IP is not reliable.
TCP Data and ACK

• Once the connection is established, data can be sent.
• Each data segment includes a sequence number identifying the first byte in the segment.
• Each segment (data or empty) includes a request number indicating what data has been received.
Buffering

• Keep in mind that TCP is (usually) part of the Operating System. It takes care of all these details *asynchronously*.
• The TCP layer doesn’t know when the application will ask for any received data.
• TCP buffers incoming data so it’s ready when we ask for it.
TCP Buffers

• Both the client and server allocate buffers to hold incoming and outgoing data
  – The TCP layer does this.
• Both the client and server announce with every ACK how much buffer space remains (the Window field in a TCP segment).
Send Buffers

• The application gives the TCP layer some data to send.
• The data is put in a send buffer, where it stays until the data is ACK’d.
  – it has to stay, as it might need to be sent again!
• The TCP layer won’t accept data from the application unless (or until) there is buffer space.
ACKs

- A receiver doesn’t have to ACK every segment (it can ACK many segments with a single ACK segment).
- Each ACK can also contain outgoing data (piggybacking).
- If a sender doesn’t get an ACK after some time limit (MSL) it resends the data.
TCP Segment Order

• Most TCP implementations will accept out-of-order segments (if there is room in the buffer).
• Once the missing segments arrive, a single ACK can be sent for the whole thing.
• Remember: IP delivers TCP segments, and IP in not reliable - IP datagrams can be lost or arrive out of order.
Termination

• The TCP layer can send a RST segment that terminates a connection if something is wrong.
• Usually the application tells TCP to terminate the connection politely with a FIN segment.
FIN

• Either end of the connection can initiate termination.
• A FIN is sent, which means the application is done sending data.
• The FIN is ACK’d.
• The other end must now send a FIN.
• That FIN must be ACK’d.
App1

1. FIN
   SN=X
2. ACK=X+1
3. FIN
   SN=Y
4. ACK=Y+1

App2

Netprog: TCP Details