CS551
Basic TCP Mechanisms

Bill Cheng

http://merlot.usc.edu/cs551-f12

What Does TCP Provide?

- Does TCP Provide...
  - Connection establishment?
  - Connectionless communication?
  - Congestion control?
  - Differentiated services?
  - Duplicate packet detection?
  - Flow control?
  - Loss recovery?
  - Message or record boundaries?
  - Ordered data delivery?
  - Out-of-order data delivery?
  - Quality of service guarantees?
  - Urgent data indication?

What Does TCP Provide?

- Does TCP Provide...
  - Connection establishment? *Y*
  - Connectionless communication? *N*
  - Congestion control? *Y*
  - Differentiated services? *Y (sort of)*
  - Duplicate packet detection? *Y*
  - Flow control? *Y*
  - Loss recovery? *Y*
  - Message or record boundaries? *N*
  - Ordered data delivery? *Y*
  - Out-of-order data delivery? *N*
  - Quality of service guarantees? *N*
  - Urgent data indication? *Y*

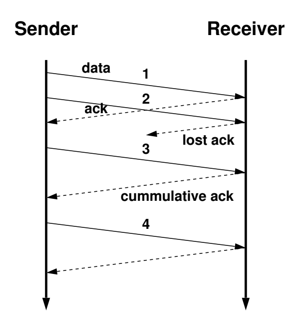
Where And Why Is TCP Used?

- Where: anywhere reliable communication is needed
  - everywhere (60-90% of traffic is TCP)
  - file transfer/ftp, http, p2p, DNS (zone transfers), BGP, SMTP, remote login/telnet
- Why?
  - connection oriented (reliability, ordering, ...)
  - has the right features (congestion control, flow control, ...)
  - widely deployed ⇒ interoperability, understood, exhaustively studies, pretty good implementations
  - doing your own protocol is a lot of work

TCP Summary

- Communication abstraction:
  - Reliable
  - Ordered
  - Point-to-point
  - Byte-stream
- Protocol implemented entirely at the ends
  - Assumes unreliable, non-sequenced delivery
  - Fate sharing
- Operations
  - OPEN/LISTEN, CONNECT, SEND, RECEIVE, ABORT

TCP Reliability Mechanism



- Summary or mechanisms
  - window-based flow control
  - sequence numbers, 3-way handshake
  - reliability (ACK, retransmission policies)
  - congestion control
  - RTT estimation

TCP Header

Flags: SYN, FIN, RESET, PUSH, URG, ACK

Source port		Destination port	
Sequence number			
Acknowledgement			
Hdr len	0	Flags	Advertised window
Checksum		Urgent pointer	
Options (variable)			
Data			

TCP Header

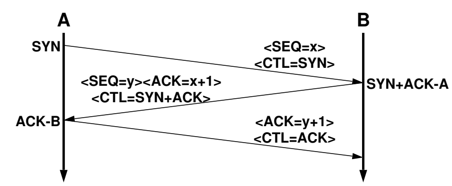
```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                        Sequence Number                        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Acknowledgment Number                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Hdr  |           |U|A|P|R|S|F|                               |
|  Len  | Reserved  |R|C|S|S|Y|I|            Advertised Window  |
|       |           |G|K|H|T|N|N|                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|           Checksum            |         Urgent Pointer        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Options (variable)         |    Padding    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                             Data                              |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```


TCP Mechanisms

- Connection establishment
- Sequence number selection
- Connection tear-down
- Round-trip estimation
- Window flow control

Connection Establishment

- A and B must agree on initial sequence number selection: Use ***3-way handshake***



- Setup initial sequence number
- Make sure other side is listening
- Pass window sizes
- Setup options (such as timestamping)

Connection Setup

```
                              +---------+ ---------\      active OPEN
                              |  CLOSED |            \    -----------
                              +---------+<---------\   \   create TCB
                                |     ^              \   \  snd SYN
                   passive OPEN |     |   CLOSE        \   \
                   ------------ |     | ----------       \   \
                    create TCB  |     | delete TCB         \   \
                                V     |                      \   \
                              +---------+            CLOSE    |    \
                              |  LISTEN |          ---------- |     |
                              +---------+          delete TCB |     |
                   rcv SYN      |     |     SEND              |     |
                  -----------   |     |    -------            |     V
 +---------+      snd SYN,ACK  /       \   snd SYN          +---------+
 |         |<-----------------           ------------------>|         |
 |   SYN   |                    rcv SYN                     |   SYN   |
 |   RCVD  |<-----------------------------------------------|   SENT  |
 |         |                    snd ACK                     |         |
 |         |------------------           -------------------|         |
 +---------+   rcv ACK of SYN  \       /  rcv SYN,ACK       +---------+
   |           --------------   |     |   -----------
   |                  x         |     |     snd ACK
   |                            V     V
   |  CLOSE                   +---------+
   | -------                  |  ESTAB  |
   | snd FIN                  +---------+
```

- Error recovery is not part of the spec

Detecting Half-open Connections

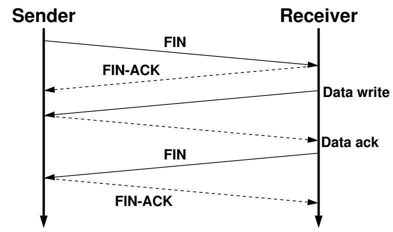
	TCP A			TCP B
1.	(CRASH)			(send 300, receive 100)
2.	CLOSED			ESTABLISHED
3.	SYN-SENT	→	<SEQ=400><CTL=SYN> →	(??)
4.	(!!)	←	<SEQ=300><ACK=100><CTL=ACK> ←	ESTABLISHED
5.	SYN-SENT	→	<SEQ=100><CTL=RST> →	(Abort!!)
6.	SYN-SENT			CLOSED
7.	SYN-SENT	→	<SEQ=400><CTL=SYN> →	

Sequence Number Selection

- Initial sequence number (ISN) selection
  - Why not simply chose 0?
  - Must avoid overlap with earlier incarnation
- Possible solutions
  - Assume non-volatile memory
  - Clock-based solutions
- Can still have sequence number overlap
  - If sequence number space not large enough for high-bandwidth connections

Connection Tear-down

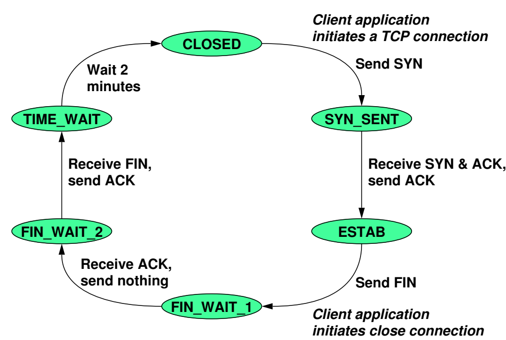
- Either side can close
  - Or one side is closed and the other stays open
- One side must maintain state (TIME_WAIT) for 2 minutes
  - Possible outstanding data in the network
  - Possible ACKs still needed



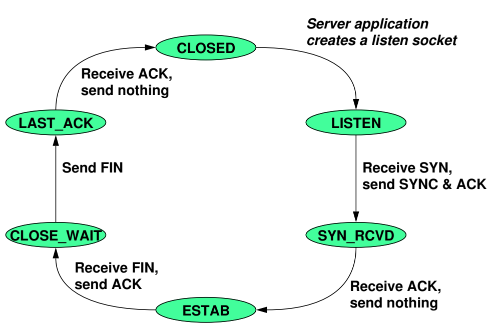
Connection Tear-down

```
  |  CLOSE                    +---------+
  | -------                   |  ESTAB  |
  | snd FIN                   +---------+
  |                   CLOSE     |     |    rcv FIN
  V                  -------    |     |    -------
+---------+          snd FIN  /       \   snd ACK          +---------+
|  FIN    |<-----------------           ------------------>|  CLOSE  |
| WAIT-1  |------------------                              |   WAIT  |
+---------+          rcv FIN  \                            +---------+
  | rcv ACK of FIN   -------   |                            CLOSE  |
  | --------------   snd ACK   |                           ------- |
  V        x                   V                           snd FIN V
+---------+                  +---------+                   +---------+
|FINWAIT-2|                  | CLOSING |                   | LAST-ACK|
+---------+                  +---------+                   +---------+
  |                rcv ACK of FIN |                 rcv ACK of FIN |
  |  rcv FIN       -------------- |    Timeout=2MSL -------------- |
  |  -------              x       V    ------------        x       V
   \ snd ACK                 +---------+delete TCB         +---------+
    ------------------------>|TIME WAIT|------------------>| CLOSED  |
                             +---------+                   +---------+
```


Typical Client TCP State Sequence



Typical Server TCP State Sequence



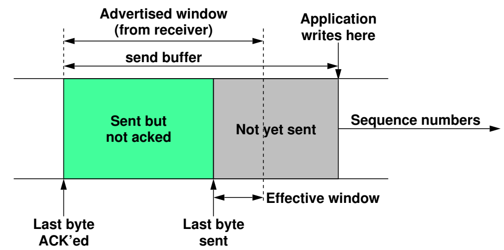
TIME-WAIT Assassination

	TCP A			TCP B
1.	ESTABLISHED			ESTABLISHED
	(Close)			
2.	FIN-WAIT-1	→	<SEQ=100><ACK=300><CTL=FIN,ACK> →	CLOSE-WAIT
3.	FIN-WAIT-2	←	<SEQ=300><ACK=101><CTL=ACK> ←	CLOSE-WAIT (close)
4.	TIME-WAIT	←	<SEQ=300><ACK=101><CTL=FIN,ACK> ←	LAST-ACK
5.	TIME-WAIT	→	<SEQ=101><ACK=301><CTL=ACK> →	CLOSED
5.1.	TIME-WAIT	←	<SEQ=255><ACK=33> ... old duplicate	
5.2.	TIME-WAIT	→	<SEQ=101><ACK=301><CTL=ACK> →	????
5.3.	CLOSED (prematurely)	←	<SEQ=301><CTL=RST> ←	????

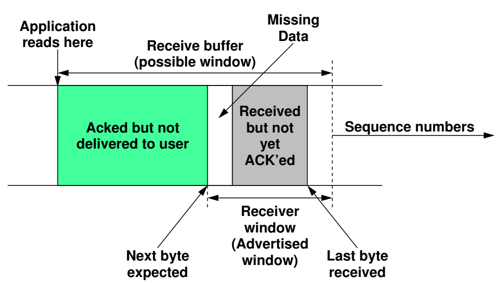
Flow Control

- Problem: Fast sender can overrun receiver
  - Packet loss, unnecessary retransmissions
- TCP's solution:
  - Window sizes are passed in every packet to avoid sender overrunning the receiver
  - Sender transmits at pre-negotiated rate
  - Sender limited to a window's worth of unacknowledged data
- Flow control different from congestion control

Window Flow Control: Sender



Window Flow Control: Receiver



Window Advancement Issues

- Advancing a full window
  - Slow receiver: receive buffer fills up (all data ACK'ed), last advertised window size is 0, sender is not allowed to send, how does it know the window has opened up?
    - solution: sender sends one data byte (probe) when advertised window size is 0
- Silly window syndrome
  - Fast sender, slow receiver: receiver with small buffer advertise it, sender quickly fills it with small amount of data
    - solution: delay ACK at receiver, Nagle's algorithm at sender
    - Nagle's algorithm -- send 1st partial packet, but not more until it is ACK'ed or have a full packet (basically, stop & wait for small packets)

TCP Extensions

- Needed for high-bandwidth delay connections
  - Accurate round-trip time estimation
  - Window size limitations
  - Impact of loss
- Implemented using TCP options
  - Timestamp
  - Protection from sequence number wraparound
  - Large windows

Timestamp Extension

- Used to improve timeout mechanism by more accurate measurement of RTT
- When sending a packet, insert current timestamp into option
- Receiver echoes timestamp in ACK

Protection From Wraparound

- Wraparound time vs. link speed:
  - 1.5Mbps (T1): 6.4 hours
  - 10Mbps (Ethernet): 57 minutes
  - 45Mbps (T3): 13 minutes
  - 100Mbps (FDDI): 6 minutes
  - 155Mbps (STS-3): 4 minutes
  - 622Mbps (STS-12): 55 seconds
  - 1.2Gbps (STS-24): 28 seconds
- Recall that MSL is 2 minutes in TCP
- Use timestamp (32-bit) to distinguish sequence number wraparound

Large Windows

- Recall that the AdvertisedWindow field is 16-bit long
- Delay × Bandwidth vs. link speed, assuming 100ms RTT
  - 1.5Mbps (T1): 18KB
  - 10Mbps (Ethernet): 122KB
  - 45Mbps (T3): 549KB
  - 100Mbps (FDDI): 1.2MB
  - 155Mbps (STS-3): 1.8MB
  - 622Mbps (STS-12): 7.4MB
  - 1.2Gbps (STS-24): 14.8MB
- Apply scaling factor to advertised window
  - Specifies how many bits window must be shifted to the left
- Scaling factor exchanged during connection setup