CS118 Discussion 1B, Week 4

Boyan Ding

Outline

- Quiz 1 Logistics
- Lecture review: Transport layer

Quiz 1 - logistics

Suggestions from Prof. Lu

1. Quiz 1 will cover Chapters 1 and 2, and part of Chapter 3 up to 3.4. The materials to study are the lecture notes, homeworks, programming project 1 (see the sample quiz regarding how socket programming from Project 1 can be tested), and textbooks.

2. You do not need to memorize a lot of acronyms as shown in the textbook or slides. Wherever needed, we will give the full name (such as SMTP (simple mail transfer protocol).

3. Do show your steps in your answer to receive partial credit, so that we know how you think to maximize your received credit.

Quiz 1 - logistics

- Time (tentative): 4-10pm (PST), Friday, Apr 30
  - Choose 2.5h within to finish the exam
  - Email me if you cannot make it

- Format: sample quiz posted on line

1. Which of the following statement is true?
   - Your answer ____ (A) HTTP is a transport-layer protocol. (B) Modularity through protocol layering makes it easier to update system components. (C) POP3 is not an application-layer protocol. (D) SMTP uses UDP protocol at its transport layer. (E) DNS is not needed for the Internet.
   - Justification:

Questions on Piazza

- In the discussion, there is an example for "dig google.com A". TA said AUTHORITY SECTION has longer TTL than ANSWER SECTION because the former does not change that much. Does that mean nsx.google.com's IP is more permanent than google.com's IP? Doesn't nsx.google.com server have lower hierachy than google.com and higher hierachy servers should change less often.

Principles of Reliable Data Transfer

- How to deal with bit errors?
  - Error detection (e.g. checksum)
  - Receiver feedback
  - Retransmission
  - Why not error correction?
- How to deal with duplicate packets due to retransmission?
- How can the sender detect that ACK or data is lost?

Version	Channel	Mechanism
rdt1.0	No error/loss	nothing
rdt2.0	bit errors (no loss)	(1)error detection via checksum (2)receiver feedback (ACK/NAK) (3)retransmission
rdt2.1	Same as 2.0	(4)Seq# (1 bit)
rdt2.2	Same as 2.0	(no NAK): Unexpected ACK = NAK
Rdt3.0	errors + loss	(5) Timer; ACK-only
Performance issue: low utilization		
Go-back-N	Same as 3.0	N sliding window (pipeline) Discard out-of-order pkts (recovery)
Selective Repeat	Same as 3.0	N sliding window, selective recovery

Stop and Wait Protocol

- Main Issue: **limited performance**

- Consider two hosts that are directly connected by a 50 Kbps satellite link that has a 250 milliseconds propagation delay. If these hosts send 1000 bits segments, what is the maximum throughput in stop-and-wait protocol if we ignore the transmission time of ACK?

Stop and Wait Protocol

- Main Issue: **limited performance**

- Consider two hosts that are directly connected by a 50 Kbps satellite link that has a 250 milliseconds propagation delay. If these hosts send 1000 bits segments, what is the maximum throughput in stop-and-wait protocol if we ignore the transmission time of ACK?

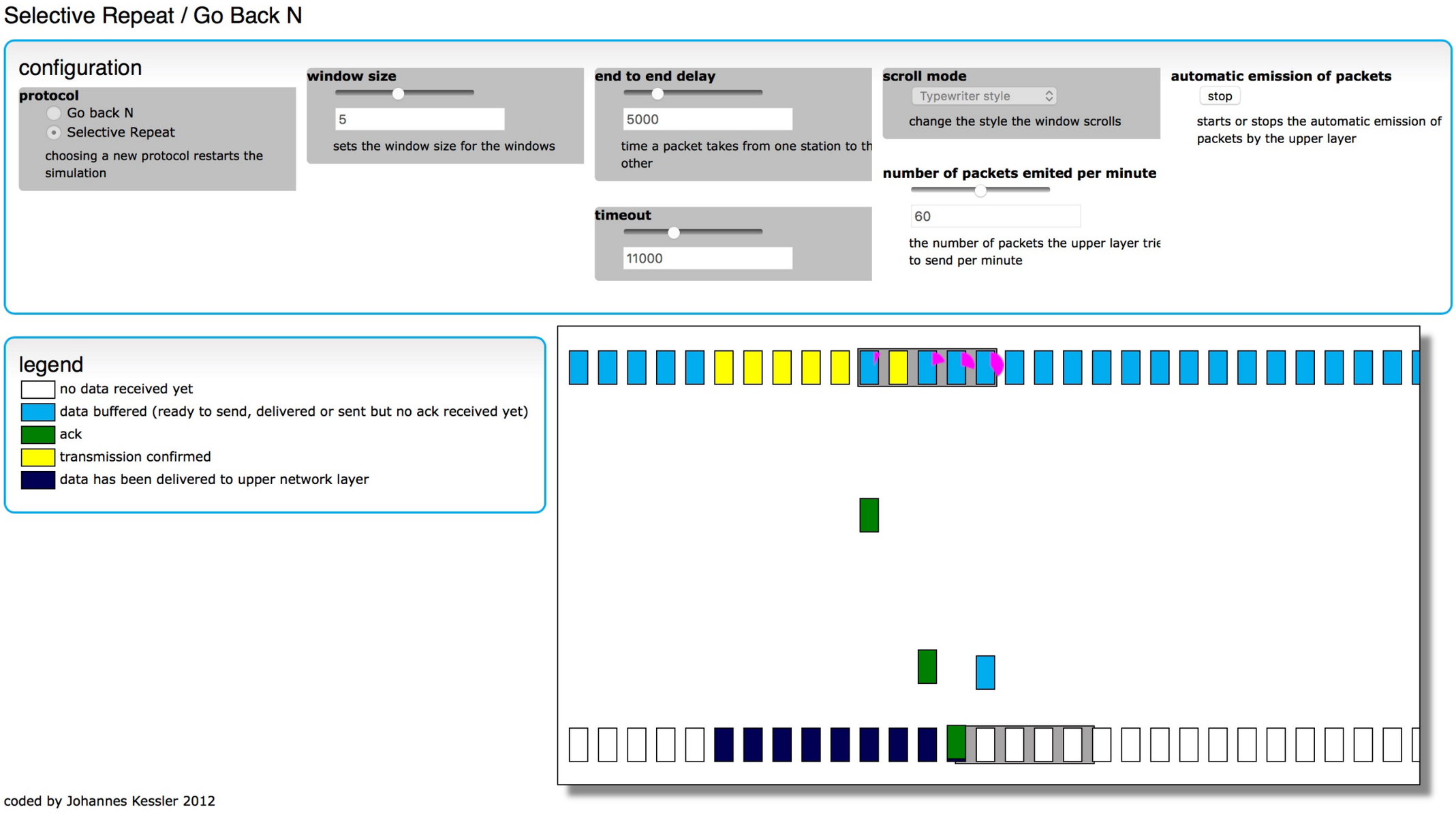
- 1000/(1000/50000+0.25+0.25) = 1923 bps < **2 Kbps << 50 Kbps!**

Pipelined Protocols

- Go-back-N: receiver only sends **cumulative** ACKs
  - Drop out-of-order segments
  - reACK packet with highest in-order sequence number
  - Timer for oldest unACKed packet only, retransmit all unACKed packets
- Selective repeat: receiver ACKs **individual** packets
  - Buffer out of order segments
  - Timer for each individual unACKed packet, retransmit any unACKed packet

Demo: Selective Repeat/Go Back N

- http://www.ccs-labs.org/teaching/rn/animations/gbn_sr/



Comparison of Reliable Transport Protocol

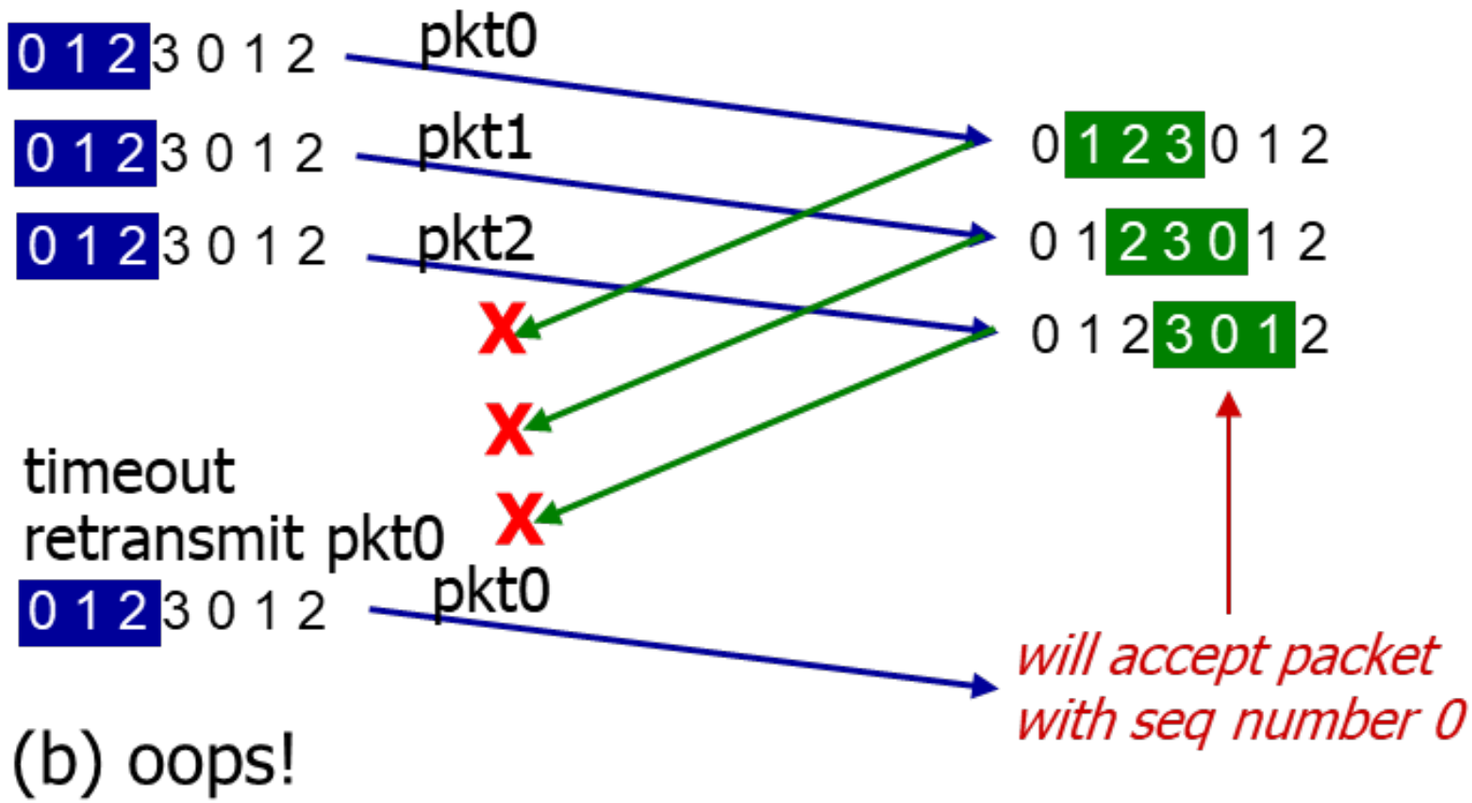
Protocol	Buffer at sender	Buffer at receiver	ACK	Timeout/Retransmission
Stop & Wait	No	No	No out-of-order	Retransmit timeout packet
Go-Back-N	Yes	No	Accumulative Seq#	Retransmit all packets in window
Selective Repeat	Yes	Yes	Received Seq#	Retransmit timeout packet

Sequence number in sliding-window schemes

- Window size = N
- Go back N: need N+1 sequence numbers
- Selective repeat: need 2N sequence numbers

Sequence number in sliding-window schemes

- In selective repeat



- How about go back N?
  - Sender send pkt0 but receiver will not accept it

TCP: overview

- The Transmission Control Protocol (TCP), documented in RFC 793, makes up for IP's deficiencies by providing reliable, stream-oriented connections that hide most of IP's shortcomings. The protocol suite gets its name because most TCP/IP protocols are based on TCP, which is in turn based on IP. TCP and IP are the twin pillars of TCP/IP.

TCP: key functionalities

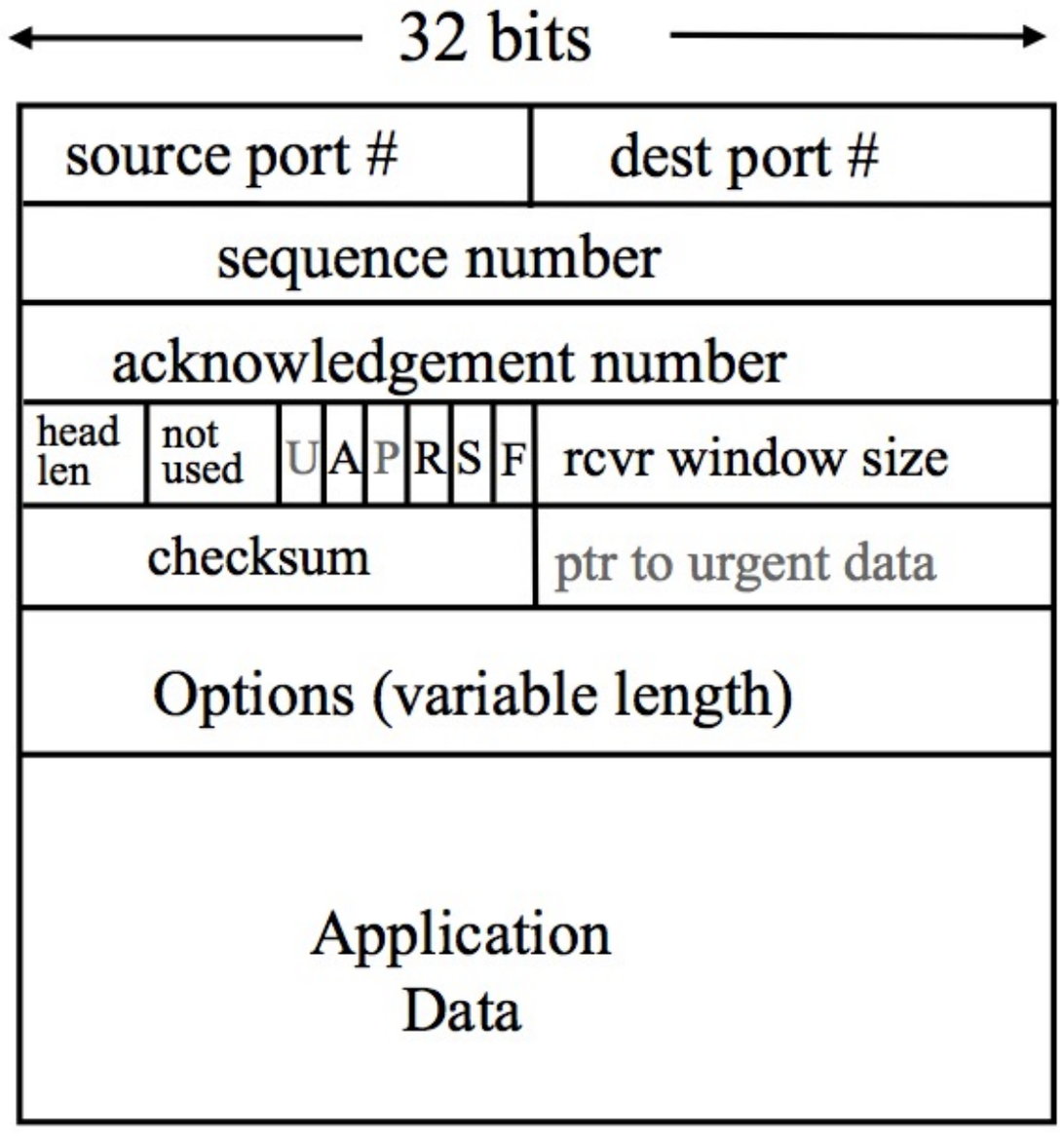
- TCP adds a great deal of functionality to the IP service it is layered over:
  - **Streams.** TCP data is organized as a stream of bytes, much like a file. The datagram nature of the network is concealed. A mechanism (the Urgent Pointer) exists to let out-of-band data be specially flagged.
  - **Reliable delivery.** Sequence numbers are used to coordinate which data has been transmitted and received. TCP will arrange for retransmission if it determines that data has been lost.
  - **Network adaptation.** TCP will dynamically learn the delay characteristics of a network and adjust its operation to maximize throughput without overloading the network.
  - **Flow control.** TCP manages data buffers, and coordinates traffic so its buffers will never overflow. Fast senders will be stopped periodically to keep up with slower receivers.

Comparison of Reliable Transport Protocol

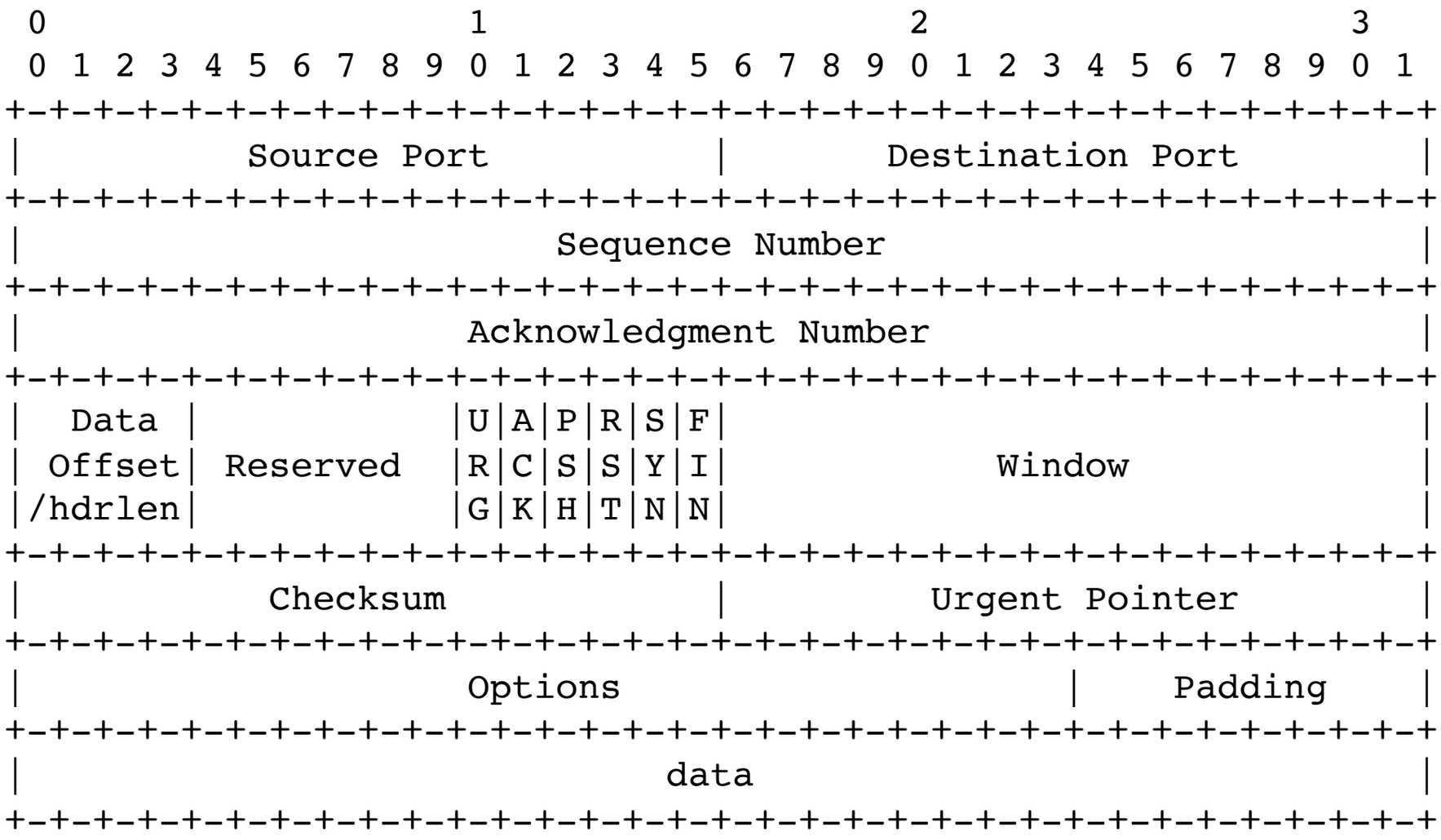
Protocol	Buffer at sender	Buffer at receiver	ACK	Timeout/Retransmission
Stop & Wait	No	No	No out-of-order	Retransmit timeout packet
Go-Back-N	Yes	No	Accumulative Seq#	Retransmit all packets in window
Selective Repeat	Yes	Yes	Received Seq#	Retransmit timeout packet
TCP	Yes	Yes	Next expected Seq#	Retransmit timeout packet

TCP

- Point-to-point, byte-stream reliable transport protocol
  - **Multiplexing/de-multiplexing**: Source/Dest port
  - **Reliable data transfer**: sequence number, ack, checksum, RTT estimation
  - **Connection setup**: sequence number, SYN, receive window
  - **Connection teardown**: sequence number, FIN



TCP: header format



TCP Header Format

Note that one tick mark represents one bit position.

TCP: header explained

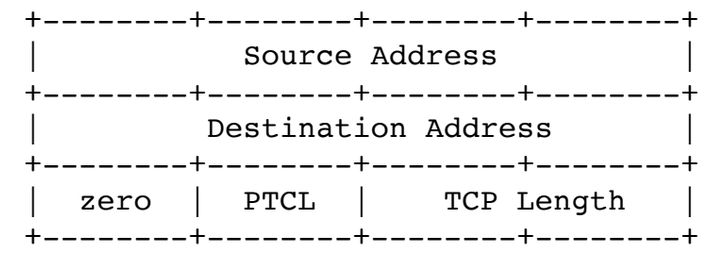
- Source Port: 16 bits
  - The source port number.
- Destination Port: 16 bits
  - The destination port number.
- Sequence Number: 32 bits
  - The sequence number of the first data octet in this segment (except when SYN is present). If SYN is present the sequence number is the initial sequence number (ISN) and the first data octet is ISN+1.
- Acknowledgment Number: 32 bits
  - If the ACK control bit is set this field contains the value of the next sequence number the sender of the segment is expecting to receive. Once a connection is established this is always sent.
- Data Offset: 4 bits
  - The number of 32 bit words in the TCP Header. This indicates where the data begins. The TCP header (even one including options) is an integral number of 32 bits long.

TCP: header explained (cont'd)

- Reserved: 6 bits
  - Reserved for future use. Must be zero.
- Control Bits: 6 bits (from left to right):
  - URG: Urgent Pointer field significant
  - ACK: Acknowledgment field significant
  - PSH: Push Function
  - RST: Reset the connection
  - SYN: Synchronize sequence numbers
  - FIN: No more data from sender
- Window: 16 bits
  - The number of data octets beginning with the one indicated in the acknowledgment field which the sender of this segment is willing to accept.

TCP: header explained (cont'd)

- Checksum: 16 bits
  - The checksum field is the 16 bit one's complement of the one's complement sum of all 16 bit words in the header and text. While computing the checksum, the checksum field itself is replaced with zeros.
  - The checksum also covers a 96 bit pseudo header conceptually prefixed to the TCP header. This pseudo header contains the Source Address, the Destination Address, the Protocol, and TCP length.
  - Why? This gives the TCP protection against misrouted segments. This information is carried in the Internet Protocol and is transferred across the TCP/Network interface in the arguments or results of calls by the TCP on the IP.



Q & A on Project 1

- Submission
  - Must include source files, Makefile, readme file
- Demo
  - We will release a spreadsheet late
  - Test files will be uploaded to CCLE after DDL
  - Instructions will be posted together.