Session 2C

TCP: Flow Control and Checksum

Reference: Ref1: TCP/IP Illustrated-Volume 1: Chapter 12: TCP

Mouli Sankaran

Session 2C: Focus

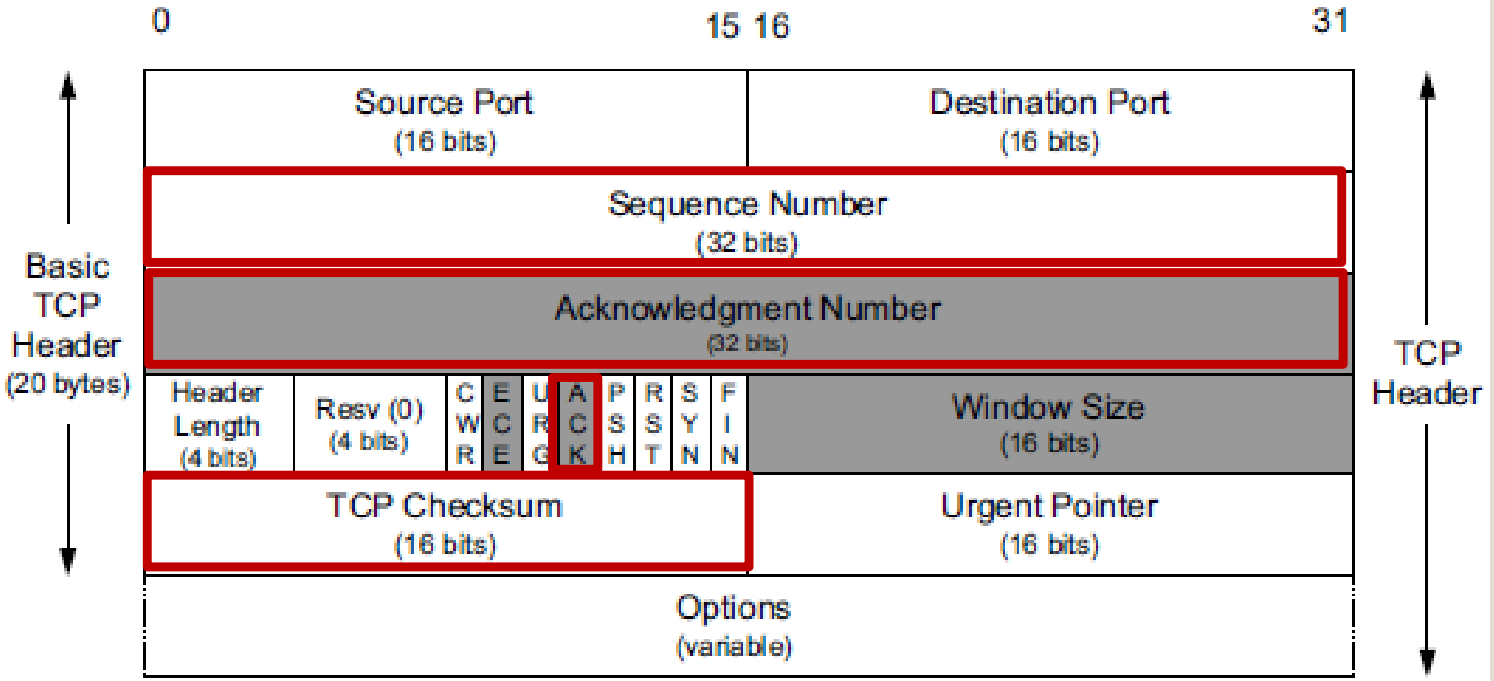
- TCP Segment Header Structure
  - Source and Destination Port Numbers
- Flow Control
  - Window Size
  - Sliding Window Protocol
- Options Field
  - TCP Data length
- TCP Checksum (including pseudo header)

Course page where the course materials will be posted as the course progresses:

TCP Segment Structure

TCP Segment Structure (Header)

What has been covered so far

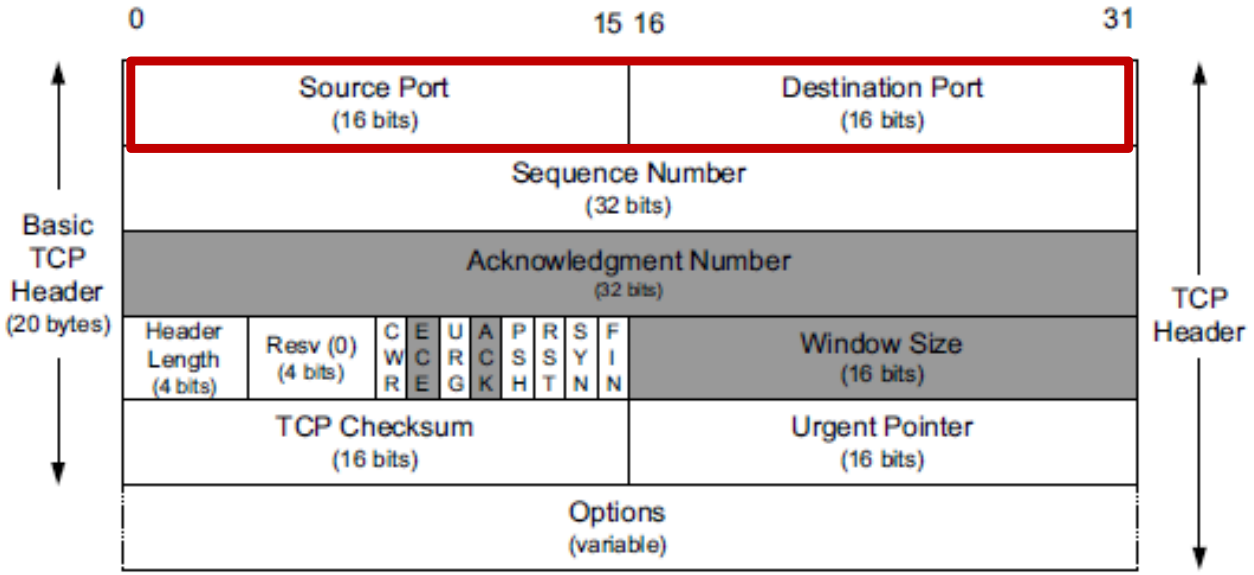


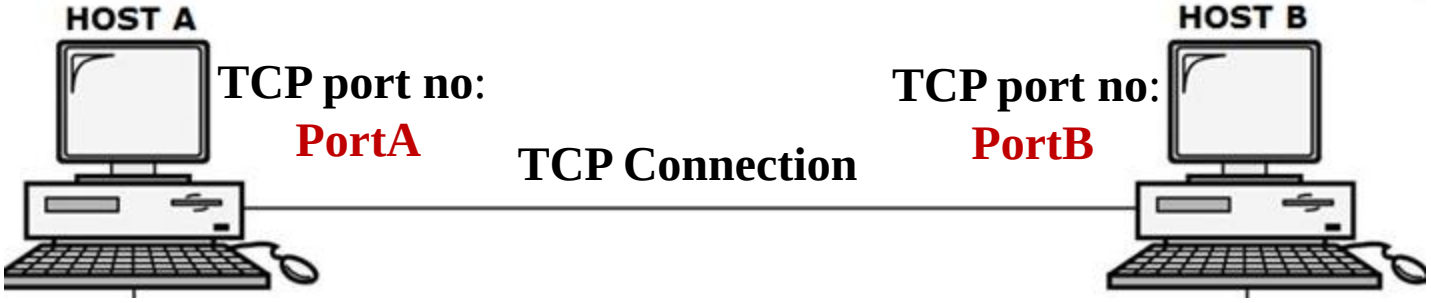
- Sequence Number (32 bits)
- Acknowledgement Flag (ACK) and Acknowledgement Number
- TCP Checksum

TCP Header Field: Source and Destination Ports

Note: The 16-bit values of **Server ports** and **Client ports** were discussed in the **Session 2B**

Note: Based on the role played by the host either as a Client or a Server, the **TCP connections** are **made** to **relevant ports**.





TCP Segment

Source Port: PortA
Destination Port: PortB

Source Port: PortB
Destination Port: PortA

TCP Segment

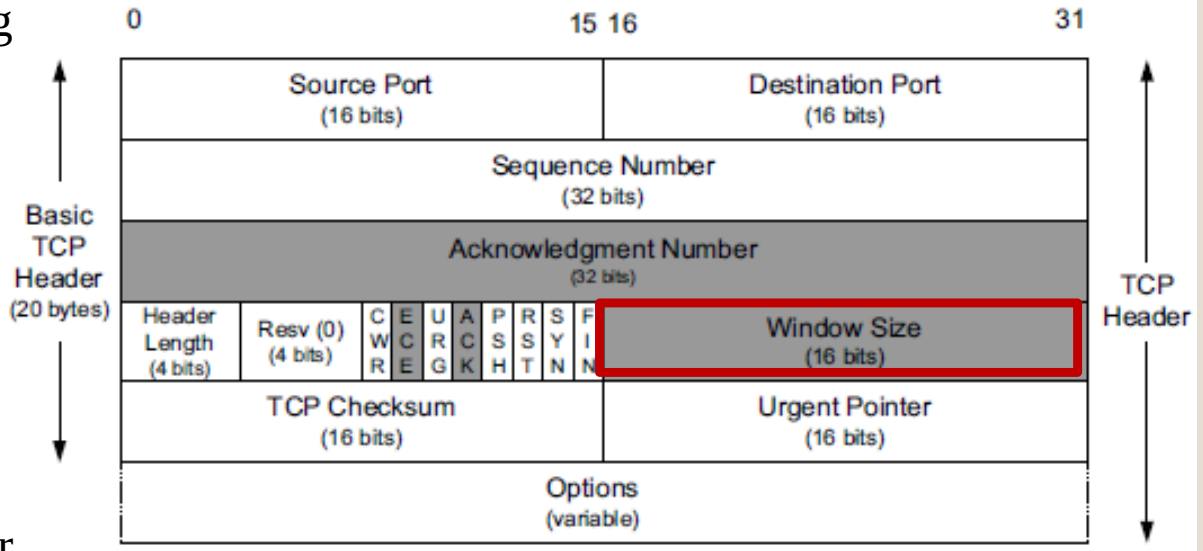
TCP- Flow Control

Flow Control – Sliding Window Protocol

TCP Header Field: Window Size

Flow control: The receiver making sure, that the sender does not overflow its buffers by sending data to the receiver at a higher rate than it can accept and process.

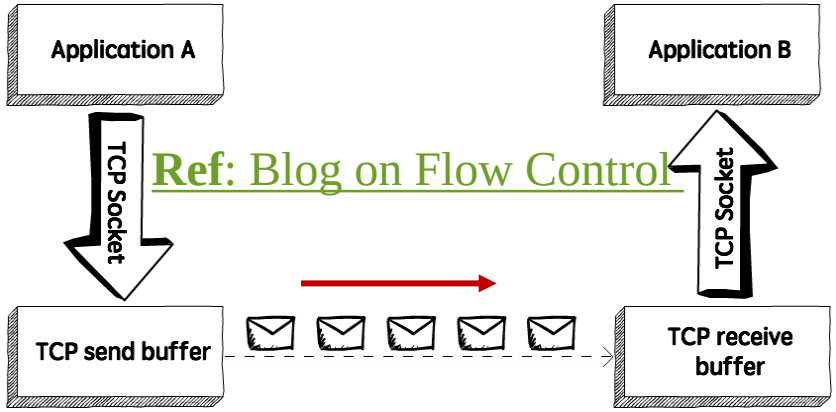
Reason: The sender could be a faster and powerful machine than the receiver, thus capable of generating data much faster than the rate of consumption by receiver.



- **TCP's flow control** is provided by each end **advertising** a **window size** using the Window Size field. This is the number of bytes, starting with the one specified by the ACK number, that the receiver is willing to accept.
- This is a 16-bit field, limiting the window to **65,535 bytes**, and thereby limiting TCP's throughput performance.
  - Even if the underlying network technology supports a higher rate of transmission, this window size limits what can be sent in one go, until the acknowledgment for the data that has been sent already, is received.

Flow Control: Using Window Size

- TCP stores the data it needs to send in the **send buffer**, and the data it receives in the **receive buffer**.
- Application and TCP/IP stack run as separate processes on the host and they exchange data between them.

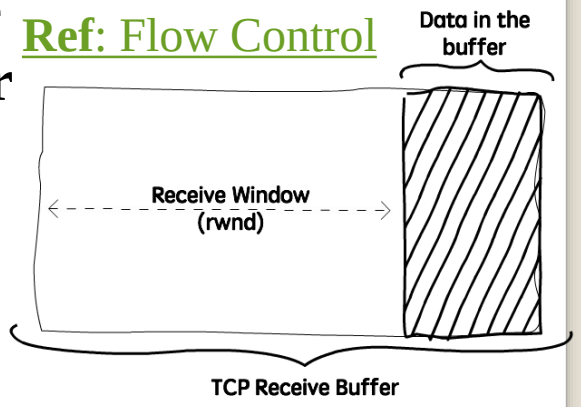


- When the application is ready, it will then read the data from the receive buffer.
- **Flow Control** is all about making sure sender doesn't send more packets when the receive buffer is already full, as the receiver wouldn't be able to handle them and would need to drop those packets.
- Dropping segments would mean that they need to be sent by the sender again which would introduce additional delays in completing the data transfer between the hosts.

Flow Control: Using Window Size … contd.

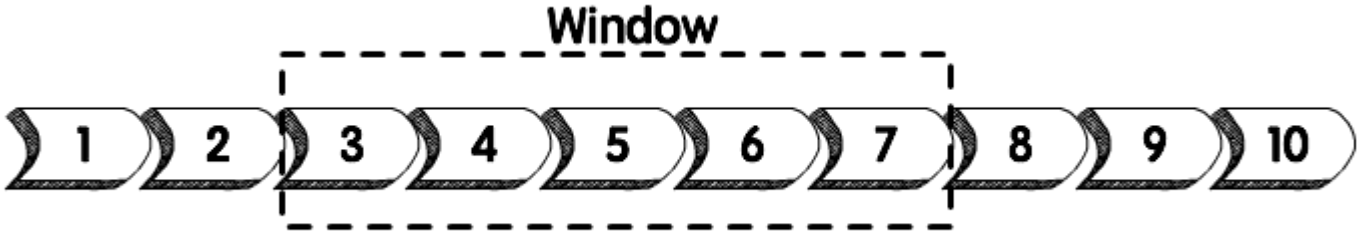
- To control the amount of data that TCP can send, the receiver will advertise its Receive Window (**rwnd or Window Size**), that is, the spare room in the receive buffer.
- "Data in the buffer" (shaded region) is the data for which the receiver has already sent the ACK to the sender.

Ref: Flow Control



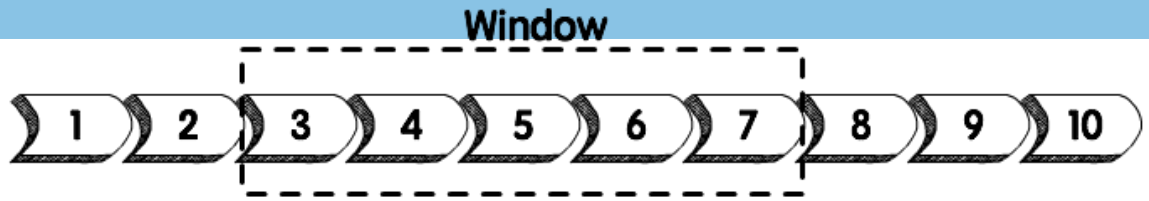
- Every time TCP receives a packet, it needs to send an **ACK** message to the sender, acknowledging that it received that packet correctly.
- And with this **ACK** message it sends the value of the **current receive window (in the window size field)**, so that the sender knows whether it can keep sending data and if yes, how many more bytes can be accommodated in the receive buffer, before receiving another ACK from the receiver .
- This is an adaptive flow control based on the current buffer available at the receiver.

Sliding Window Protocol



- TCP uses a **sliding window protocol** to control the number of bytes in flight, it can have. In other words, the number of bytes that were sent but not yet **ACKed** by the receiver
- Let's say we want to send a file with a size of 150,000 bytes from a node A to another node B.
- TCP could break this file down into 100 packets, 1500 bytes each.
- Assume that after the connection between node A and B is established, node B advertises a receive window of 45,000 bytes (based on it receive buffer size)
- Seeing that, TCP at the sender knows it can send the first 30 segments (1500 * 30 = 45000) before it receives an acknowledgment from node B.

Sliding Window Protocol … contd.



- If it gets an ACK message for the first 10 packets (meaning we now have only 20 packets in flight), and if the receive window present in these ACK messages is still 45,000, it can send the next 10 segments.
- This makes the number of segments in flight back to 30, that is the limit defined by the receive window.
- In other words, at any given point in time, there can be 30 segments in flight, that were sent but not yet **ACKed.**
- Now, if for some reason the application reading the data in node B slows down, TCP will still ACK the packets that were correctly received, but as these packets need to be stored in the receive buffer until the application decides to read them, the receive window will be made smaller (say 30,000) to slow down the sender.

TCP Segment Structure

Options Field and TCP Data length

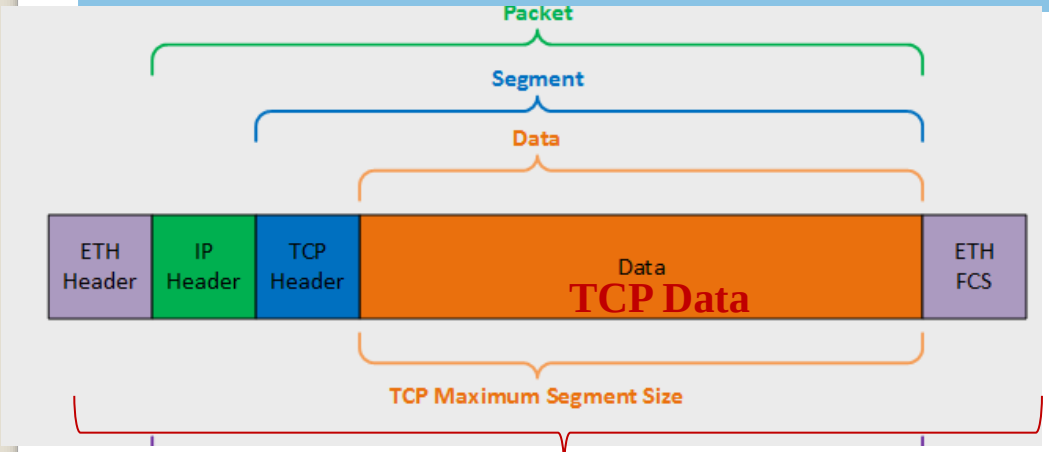
TCP Header: Options Field & Header Length

TCP Header

Offsets	Octet	0								1								2								3							
Octet	Bit	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	Source port																Destination port														
4	32	Sequence number																														
8	64	Acknowledgment number (if ACK set)																														
12	96	Data offset **HDR Len**				Reserved 0 0 0				NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size													
16	128	Checksum																	Urgent pointer (if URG set)													
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) **Max 40 bytes**																														

- The TCP header (without the Options field) is 20 bytes in length.
- The **Header Length** field gives the **length** of the **header** in **32-bit words**.
- This is required because the **length** of the ***Options*** field is **variable**.
- With a 4-bit Data Offset field, TCP is limited to a 60-byte header. Without options, however, the size is 20 bytes. **Note**: Max Options field length is 0-40 bytes
- It provides a way to deal with limitations of the original **TCP header.**

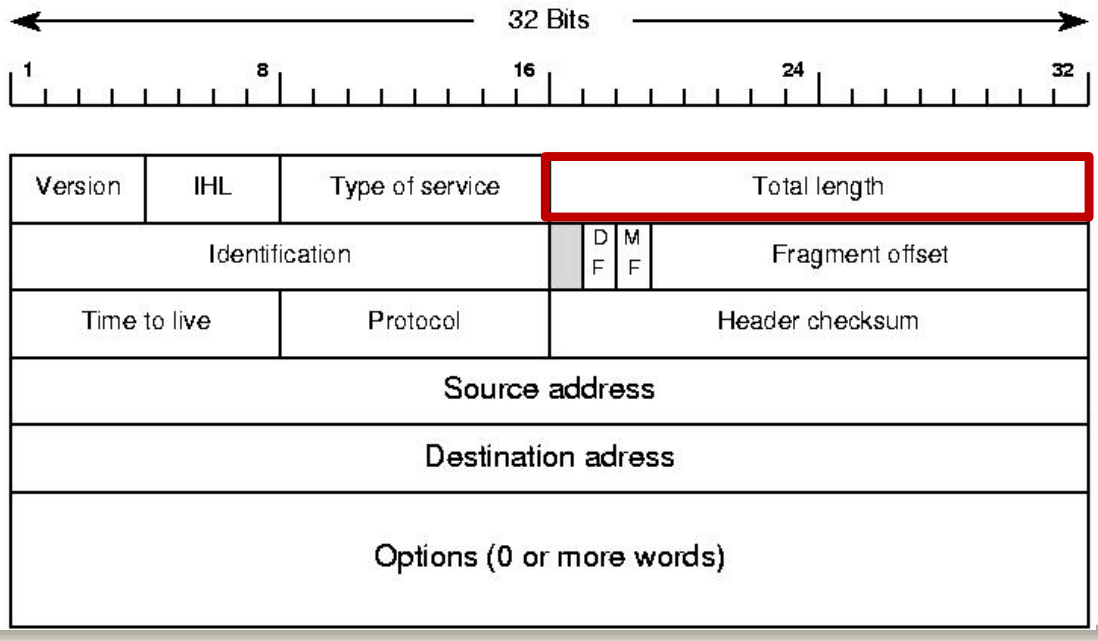
TCP Data Length (not part of TCP header)



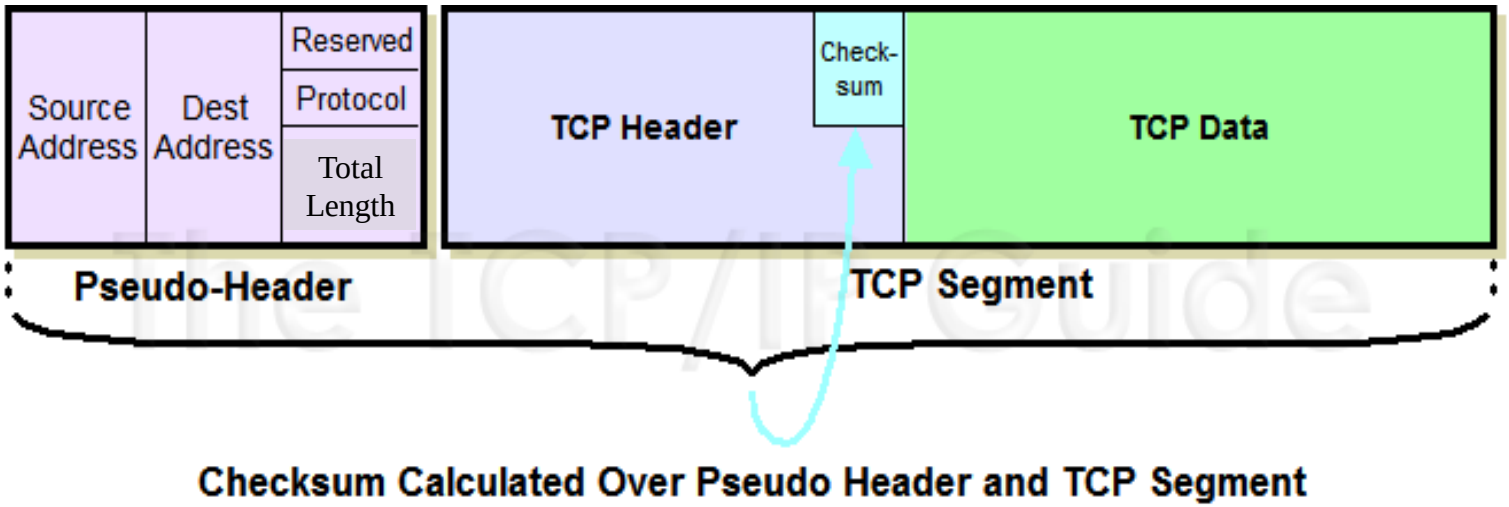
- The "**Total Length**" field in **IP header** captures the **length** of TCP Data + Header, in **bytes**.
- There is no separate field in TCP header for TCP data length.

Total length: Field in the IP header contains the Length in bytes of the IP header and its payload

IP Header



TCP Checksum Calculation with "Pseudo Header values" from IP Header



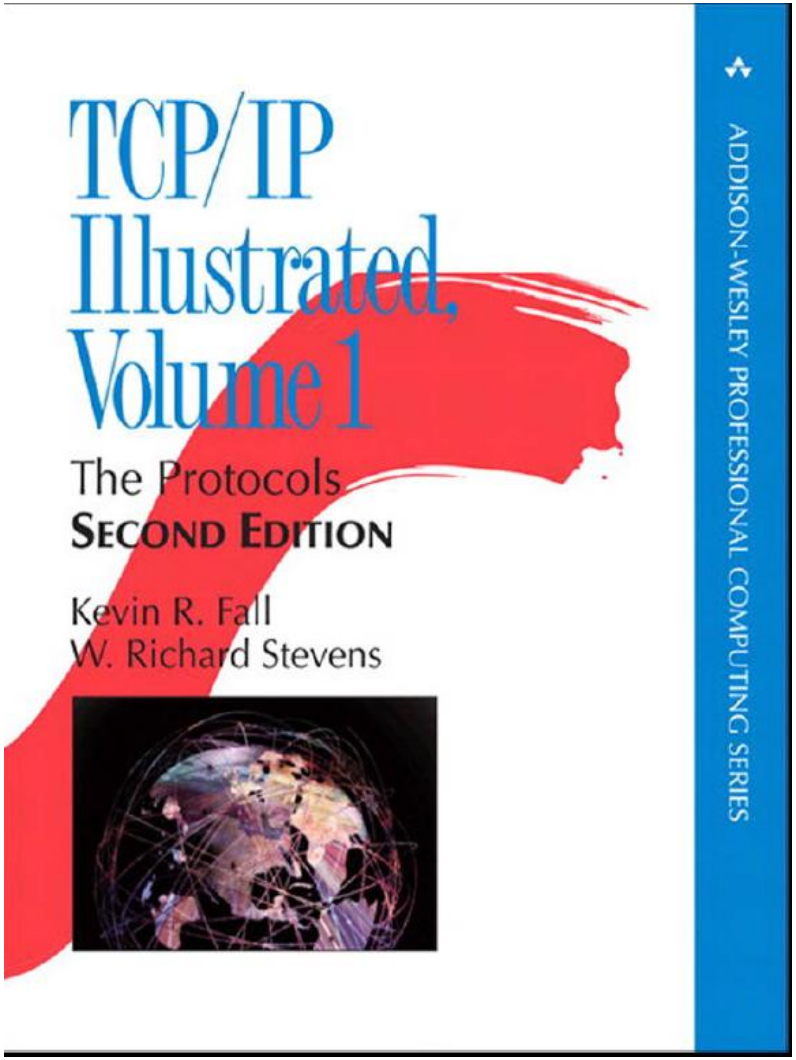
- It shows how the TCP check sum is calculated by including the contents of part of the IP header
- Pseudo-header is of 12 bytes:
  - Source and Destination IP Addresses in the IPv4 header: 4 + 4 = **8 bytes**
  - Total length (IP header): **2 bytes**
  - Protocol field: **1 byte**
  - Reserved (filled with all zeros): **1 byte**

Session 2C: Summary

- TCP Segment Header Structure
  - Source and Destination Port Numbers
- Flow Control
  - Window Size
  - Sliding Window Protocol
- Options Field
  - TCP Data length
- TCP Checksum (including pseudo header)

References

Ref 1



Ref 2

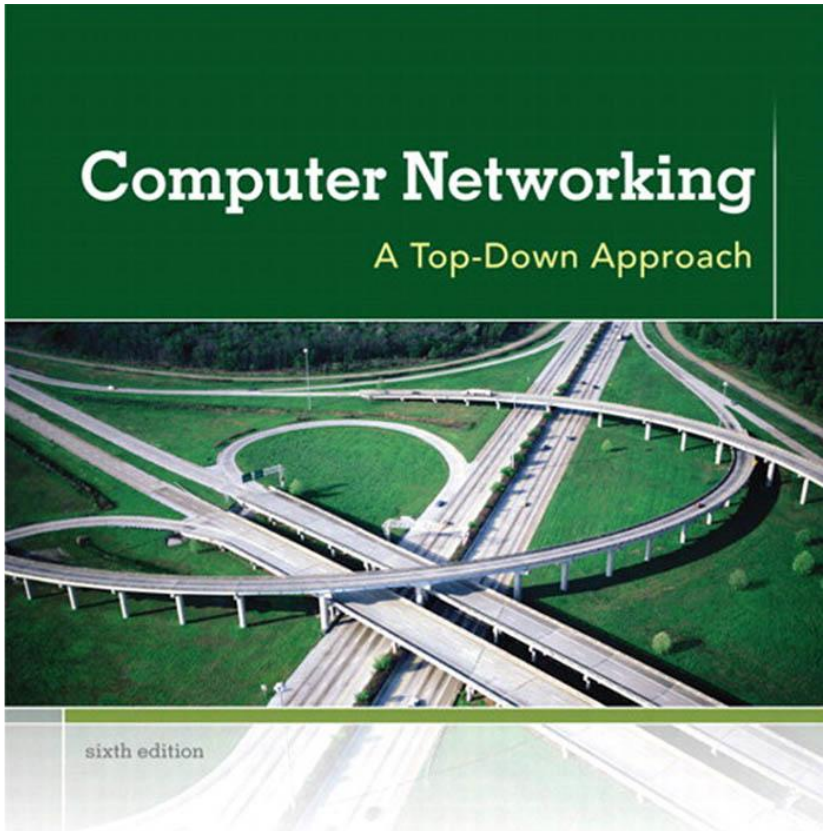
TCP Congestion Control: A Systems Approach

TCP Congestion Control: A Systems Approach

Peterson, Brakmo, and Davie

References

Ref 3



Ref 4