Computer Networks

Transport Control Protocol (TCP)

Lecture Contents

- References
- TCP Basics
- TCP Header
- TCP Handshake
- TCP Control Flags
- TCP and Web Pages
- Fundamental Concepts in Networking

References

- The most recent specification is RFC-9293 ***Transmission Control Protocol*** (**TCP**)
  - Obsoletes original specification (RFC-793) and collects many piecemeal additions

https://datatracker.ietf.org/doc/html/rfc9293

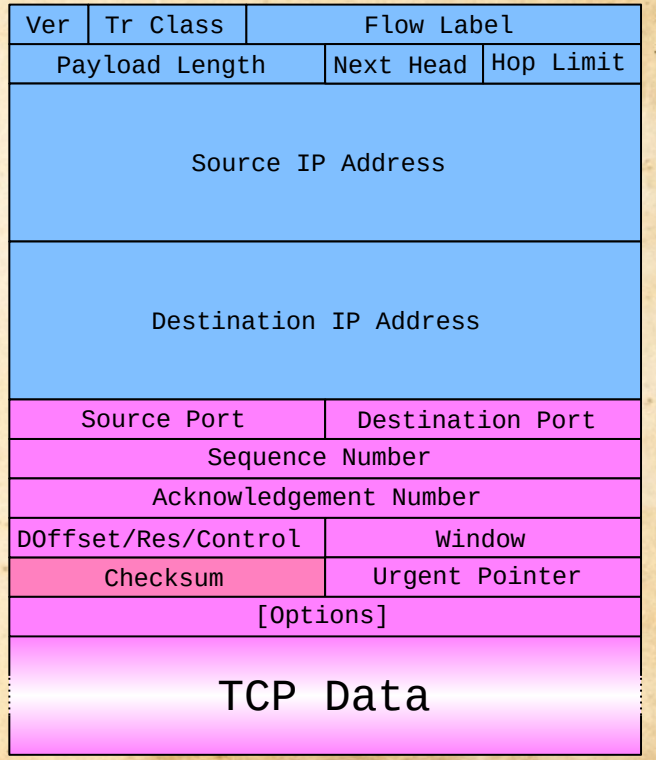
Transmission Control Protocol – Basics

- TCP Provides a reliable, in-order byte-stream service
  - Detects packet loss via sequence numbering
  - Detects errors via packet checksum
  - Requests retransmission of lost/errored packets
  - Uni- or bi-directional
- Connection oriented
  - Handshaking to initialize

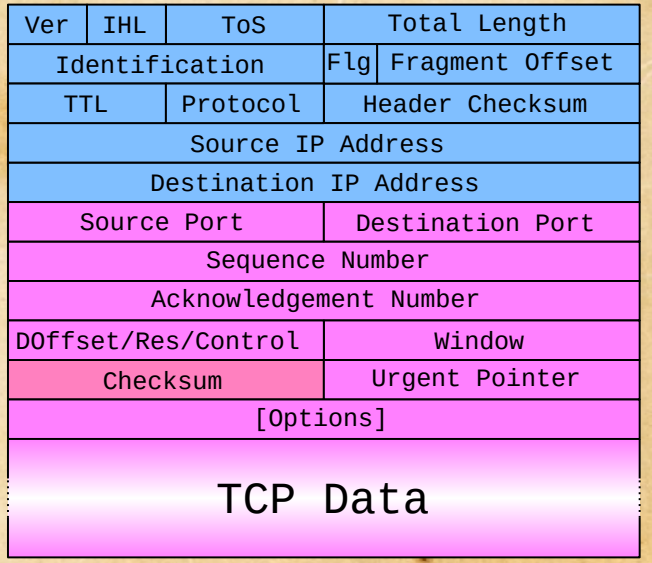
TCP Header

- Immediately follows the IPv4 or IPv6 header

IP version 6



IP version 4



TCP Header

- Immediately follows the IPv4 or IPv6 header

TCP Header

Source Port	Destination Port
Sequence Number	
Acknowledgement Number	
DOffset/Res/Control	Window
Checksum	Urgent Pointer
[Options]	
TCP Data	

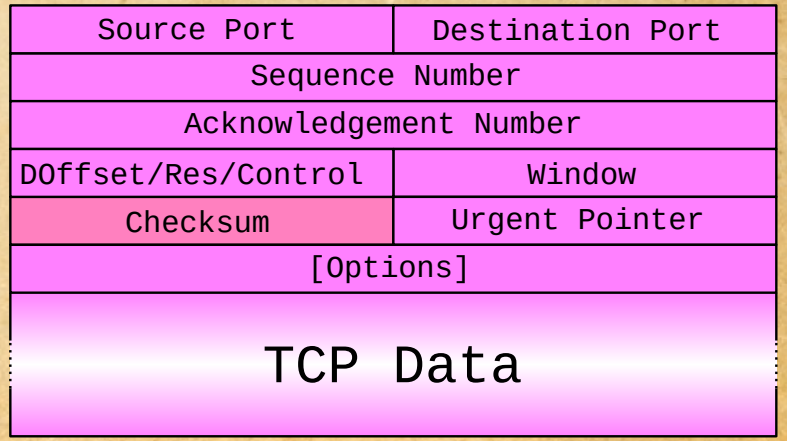
UDP Header

Source Port	Destination Port
Length	Checksum
UDP Data	

TCP Header

- Destination port
  - Allows determination of which process should receive the data
  - TCP port numbers are independent of UDP port numbers
  - Port numbers are assigned by *Internet Assigned Numbers Authority* (IANA).
    - 0-1023 well-known (FTP, HTTP, …)
    - 1024-49151 registered
    - 49152-65535 private / dynamic
- Source port
  - The port to sent the response to

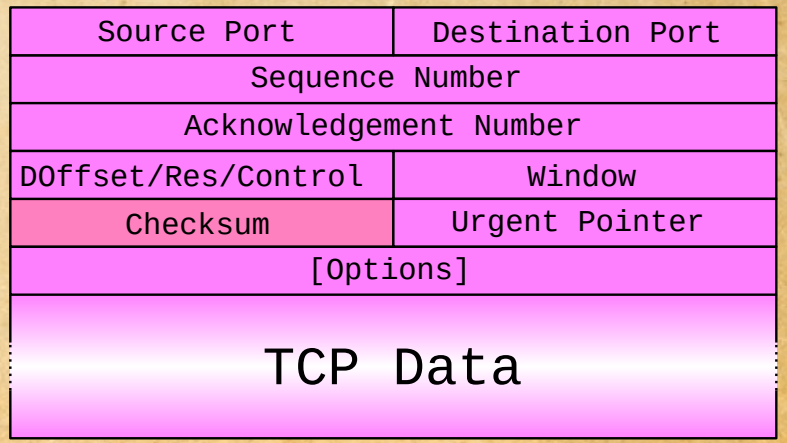
TCP Header



TCP Header

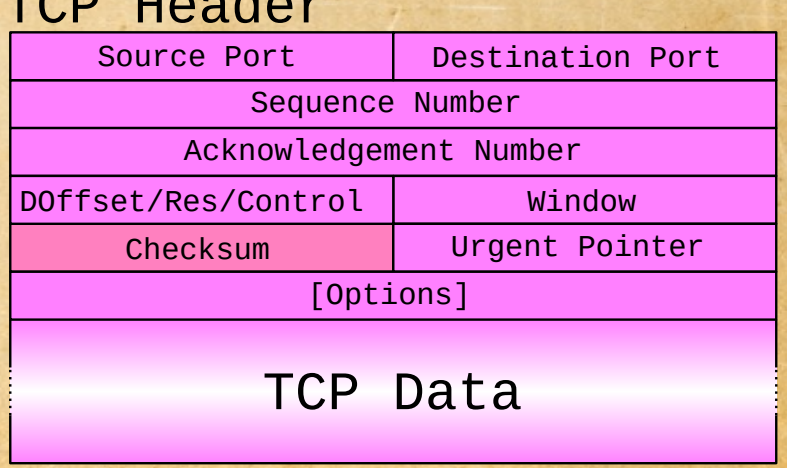
- Sequence Number
  - Number of the first byte of outgoing data
  - Tracks how much data has been transmitted
  - Starting value established during connection setup (often not zero)
- Acknowledgement Number
  - Informs the sender how much data has successfully been received
  - The sequence number of the ***next*** byte the receiver expects to receive

TCP Header



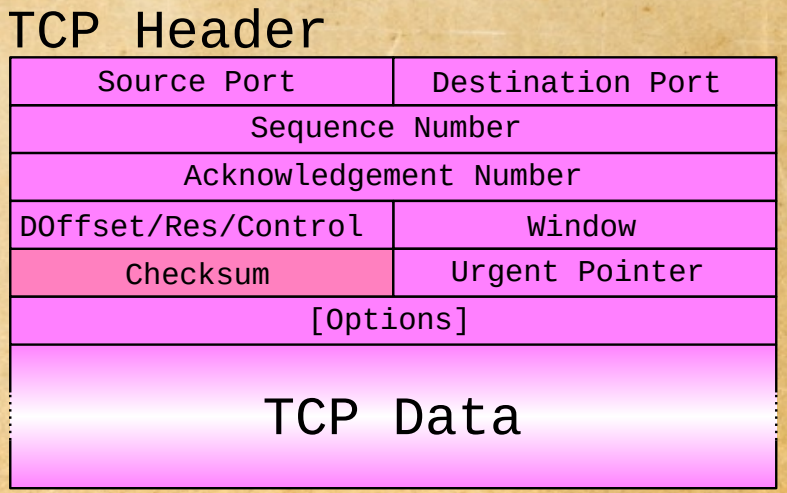
TCP Header

- Data Offset
  - 4 bit field indicating the length of the TCP header
- Reserved
  - 3 bits reserved for future use
- Control (flags)
  - 9 bits used to establish connections, send data, and terminate connections
- Window
  - Flow control: how much buffer space remains



TCP Header

- Urgent Pointer
  - Not really used
- Options
  - 0-40 bytes



TCP Handshake

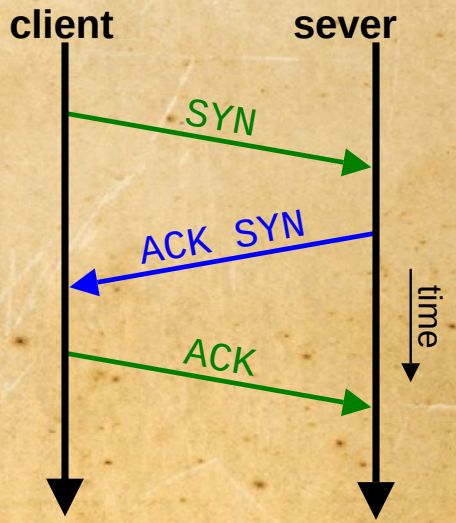
- “Three-way” handshake
  - Client request:
    - SYN with initial client sequence number
  - Server reply:
    - ACK, acknowledging the client sequence number
    - SYN with initial server sequence number
  - Client reply:
    - ACK, acknowledging the server sequence number
- Sequence numbers are now synchronized

TCP Header

Source Port	Destination Port
Sequence Number	
Acknowledgement Number	
DOffset/Res/Control	Window
Checksum	Urgent Pointer
[Options]	
TCP Data	

Control Bits

NS	CWR	ECE	URG	ACK	RST	PSH	SYN	FIN



TCP Header Control Bits

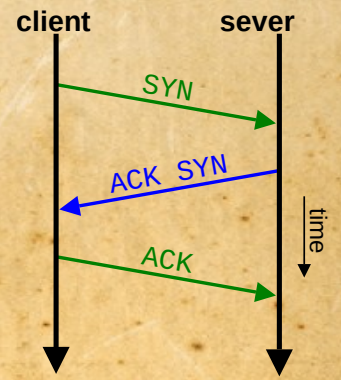
- SYN – synchronization
  - Sent during handshaking with an initial sequence number
- ACK – acknowledgement
  - Sent with every packet, *except* :
    - during the initial ***SYN*** handshake request when there's no sequence number to acknowledge
    - when a ***RST*** reset is sent

TCP Header

Source Port	Destination Port
Sequence Number	
Acknowledgement Number	
DOffset/Res/Control	Window
Checksum	Urgent Pointer
[Options]	
TCP Data	

Control Bits

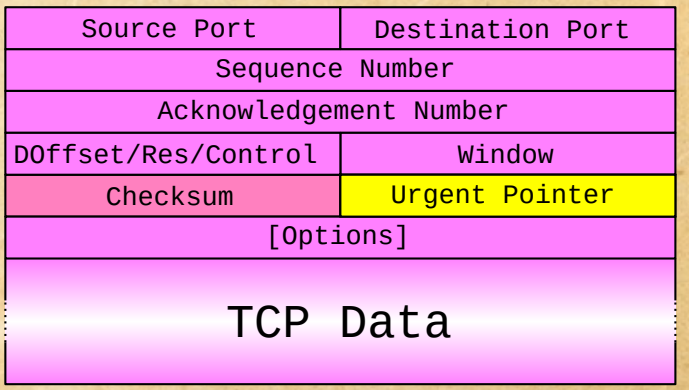
NS	CWR	ECE	URG	ACK	RST	PSH	SYN	FIN



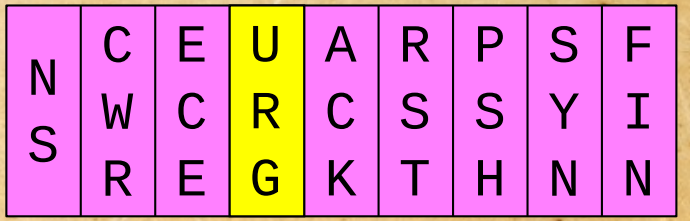
TCP Header Control Bits

- URG – urgent
  - Rarely used
    - Differences in implementation across operating systems makes use unpredictable
    - Not well implemented
    - Urgent data may use a separate channel, or an entirely different approach.
  - Used in conjunction with the `Urgent Pointer`
  - Points to urgent data within the packet

TCP Header



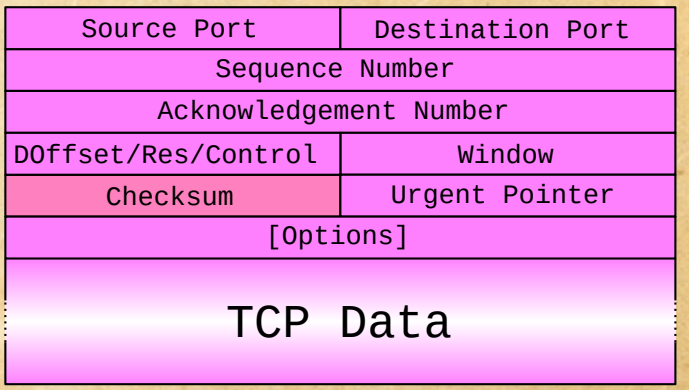
Control Bits



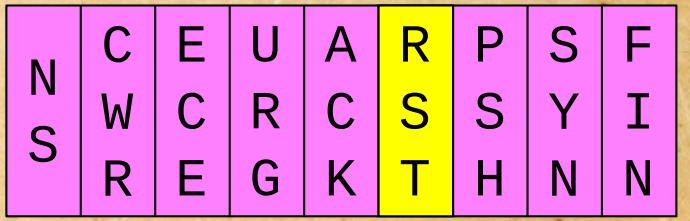
TCP Header Control Bits

- RST – reset
  - Reset the connection abruptly
  - Error or unexpected packet
  - Delete state information and forget everything about the connection
  - To send more data, a new connection must be established.

TCP Header

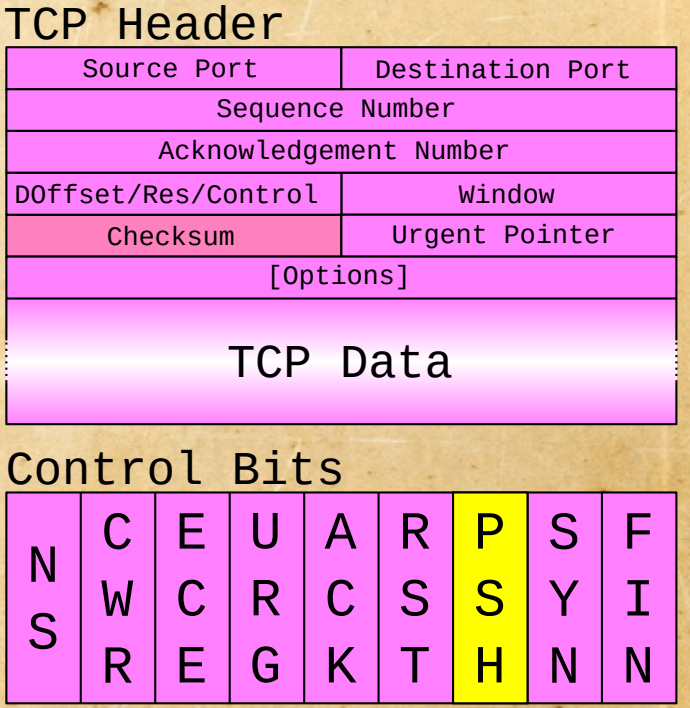


Control Bits



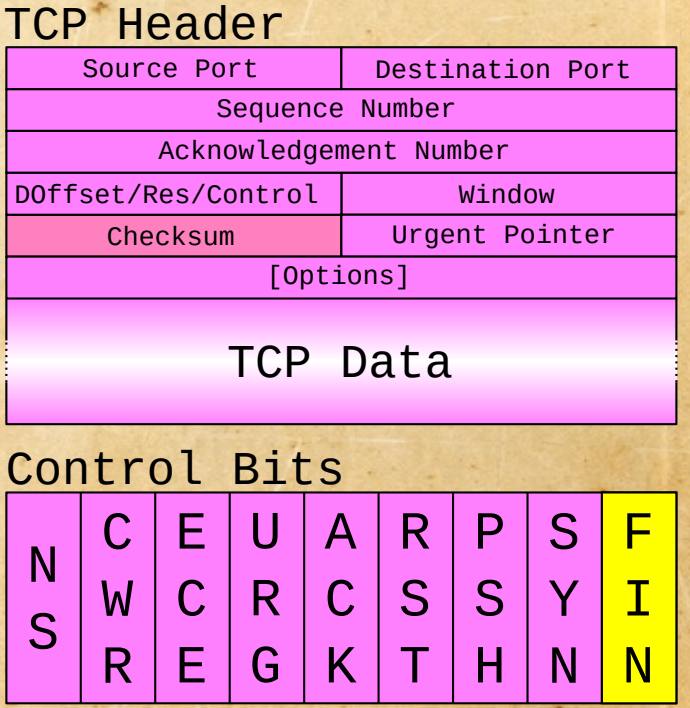
TCP Header Control Bits

- ***PSH*** – Push
  - Indicates the data should be immediately pushed to the application layer without buffering.
    - If no ***PSH***, the TCP layer may buffer and notify the application when a set amount of data is received.



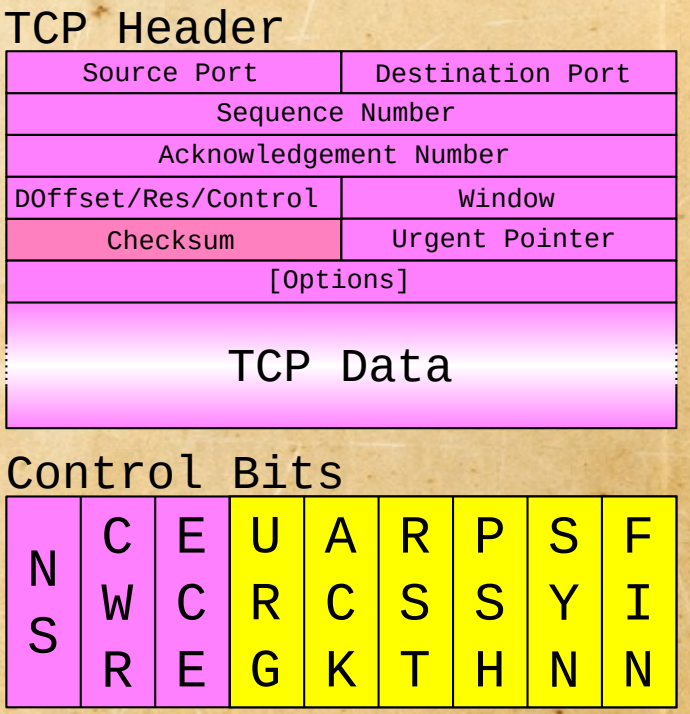
TCP Header Control Bits

- ***FIN*** – Finish
  - Graceful end of transmission request
  - In case the *FIN* bit is send in a packet with no data, setting the *FIN* bit will increase the ***Sequence Number*** field by 1
    - So that the receiver can acknowledge the end of data separately from acknowledging the ***FIN*** bit.
  - The other side may continue to send data (“half-open connection”)
    - Finished side can only send empty control packets.
    - When other side is finished, it also sends ***FIN***.



TCP Header Control Bits

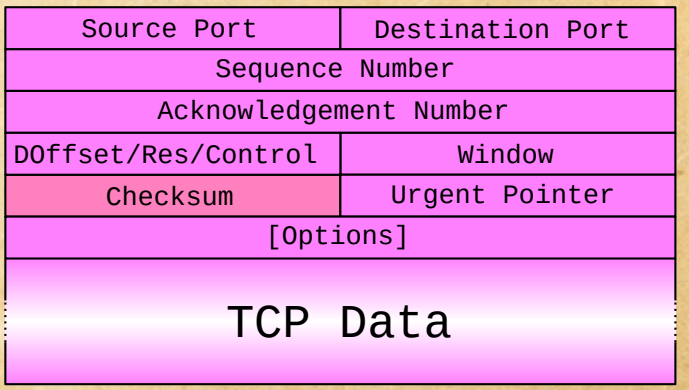
- URG, ACK, RST, PSH, SYN, FIN
  - These six bits were defined in the original specification.



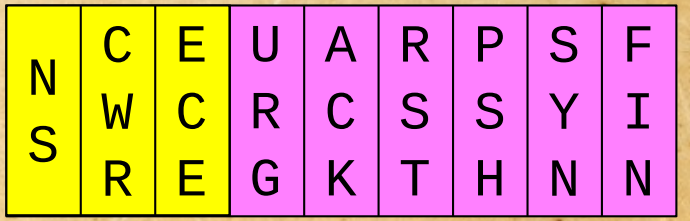
TCP Header Control Bits

- ***ECE*, *CWR***
  - *Echo of Congestion Encountered* and *Congestion Window Reduced*
  - Defined in ***RFC 3168***, along with changes in the IP header
  - Allow for end-to-end notification of network congestion
- ***NS*** – nonce sum
  - Defined in ***RFC 3540***; not widely deployed
  - Protects against accidental or malicious concealment of marked packets from the TCP sender

TCP Header



Control Bits



TCP and Web Pages

- For transmitting web pages, a separate TCP connection is typically made for each file.
- Real-time web applications may use Websockets, which allows full-duplex communication between server and client
  - Useful for applications where the server wants to notify of updated info
  - TCP connection is kept open long-term
    - This requires TCP "*keepalive*" packets to be sent if no data for a long time
    - *Keepalive* packets have no payload and a sequence number one less than the next sequence number in order to prompt the other side to provide an ***ACK*** packet

Fundamental Networking Concepts

- ***Connection-oriented***: a ***state*** of communication transfer is maintained, for example
  - Receiving:
    - Buffers of data received (including out-of-order packets)
    - Sequence number to be acknowledged
  - Sending:
    - Window size available for transmission
    - Send buffers for packets to be sent and for packets awaiting acknowledgement
    - Timers for retransmission of unacknowledged data
  - Flags: error, finished, etc.
- ***Connectionless Protocols*** do <u>not</u> need to maintain any state info

Fundamental Networking Concepts

- ***Connection-oriented*** Protocols are able to provide:
  - ***Flow Control*** – allows the receiver to notify the sender about the amount of data it can receive (its *buffer capacity*).
  - ***Ordered Data Transfer*** – the order of incoming packets is not guaranteed at the network (IP) layer, and UDP (connectionless) also does not guarantee. A connection is required to ensure ordered data transfer.
  - ***Reliability*** – if packets are lost, retransmission can be requested.

Computer Networks

Transport Control Protocol (TCP)