

# Transport Protocols

Raj Jain

Professor of CIS

The Ohio State University

Columbus, OH 43210

Jain@ACM.Org

<http://www.cis.ohio-state.edu/~jain/cis677-98>



## □ TCP

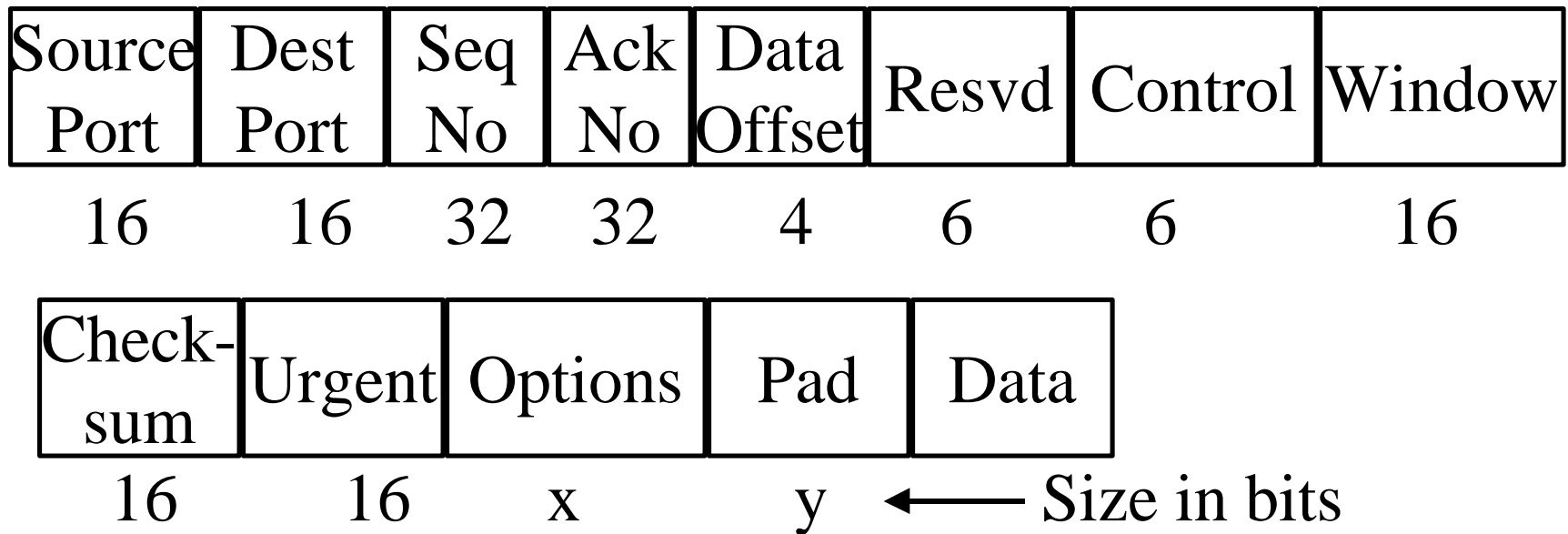
- Key features
- Header format
- Mechanisms
- Implementation choices
- Slow start congestion avoidance

## □ UDP

# TCP

- ❑ Transport Control Protocol
- ❑ Key Services:
  - Send: Please send when convenient
  - Data stream push: Please send it all now, if possible.
  - Urgent data signaling: Destination TCP! please give this urgent data to the user  
(Urgent data is delivered in sequence. Push at the should be explicit if needed.)
  - Note: Push has no effect on delivery.  
Urgent requests quick delivery

# TCP Header Format

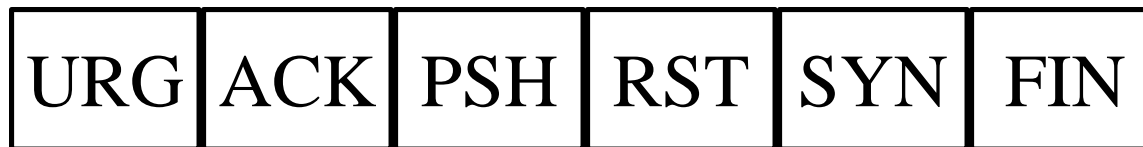


# TCP Header

- ❑ Source Port (16 bits): Identifies source user process  
20 = FTP, 23 = Telnet, 53 = DNS, 80 = HTTP, ...
- ❑ Destination Port (16 bits)
- ❑ Sequence Number (32 bits): Sequence number of the first byte in the segment. If SYN is present, this is the initial sequence number (ISN) and the first data byte is ISN+1.
- ❑ Ack number (32 bits): Next byte expected
- ❑ Data offset (4 bits): Number of 32-bit words in the header
- ❑ Reserved (6 bits)

# TCP Header (Cont)

- Control (6 bits): Urgent pointer field significant,  
Ack field significant,  
Push function,  
Reset the connection,  
Synchronize the sequence numbers,  
No more data from sender



- Window (16 bits): Will accept [Ack] to [Ack]+[window]

# TCP Header (Cont)

- ❑ Checksum (16 bits): covers the segment plus a pseudo header. Includes the following fields from IP header: source and dest adr, protocol, segment length. Protects from IP misdelivery.
- ❑ Urgent pointer (16 bits): Points to the byte following urgent data. Lets receiver know how much data it should deliver right away.
- ❑ Options (variable):  
Max segment size (does not include TCP header, default 536 bytes), Window scale factor, Selective Ack permitted, Timestamp, No-Op, End-of-options

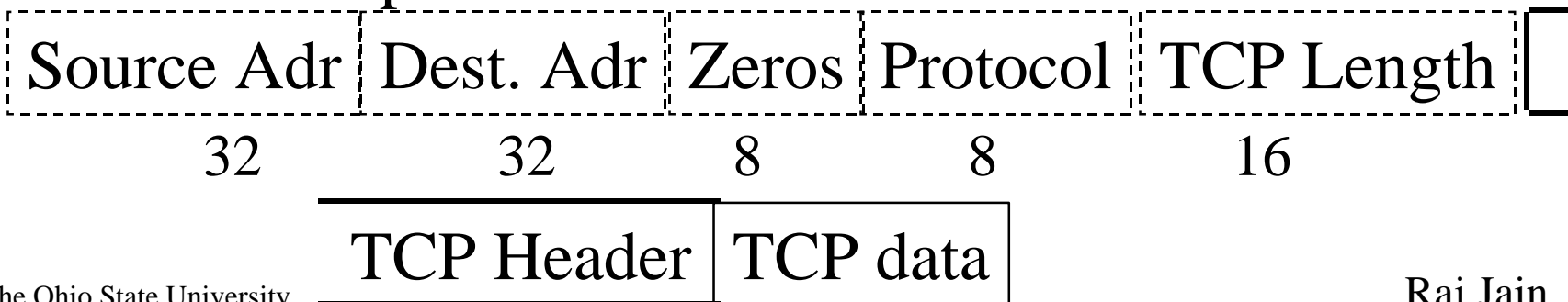
# TCP Options

Kind	Length	Meaning
0	1	End of Valid options in header
1	1	No-op
2	4	Maximum Segment Size
3	3	Window Scale Factor
8	10	Timestamp

- ❑ End of Options: Stop looking for further option
- ❑ No-op: Ignore this byte. Used to align the next option on a 4-byte word boundary
- ❑ MSS: Does not include TCP header

# TCP Checksum

- ❑ Checksum is the 16-bit one's complement of the one's complement sum of a pseudo header of information from the IP header, the TCP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.
- ❑ Checksum field is filled with zeros initially
- ❑ TCP length (in octet) is not transmitted but used in calculations.
- ❑ Efficient implementation in RFC1071.



# TCP Service Requests

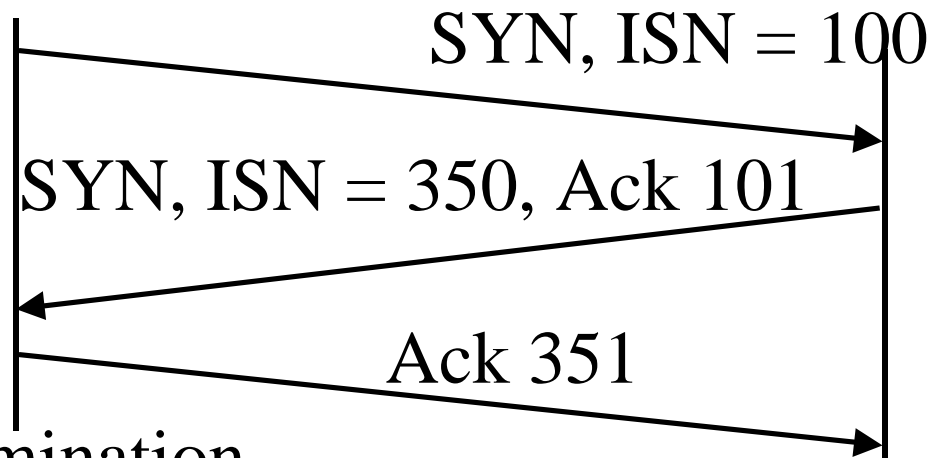
- ❑ Unspecified passive open:  
Listen for connection requests from any user (port)
- ❑ Full passive open:  
Listen for connection requests from specified port
- ❑ Active open: Request connection
- ❑ Active open with data: Request connection and transmit data
- ❑ Send: Send data
- ❑ Allocate: Issue incremental allocation for receive data
- ❑ Close: Close the connection gracefully
- ❑ Abort: Close the connection abruptly
- ❑ Status: Report connection status

# TCP Service Responses

- ❑ Open ID: Informs the name assigned to the pending request
- ❑ Open Failure: Your open request failed
- ❑ Open Success: Your open request succeeded
- ❑ Deliver: Reports arrival of data
- ❑ Closing: Remote TCP has issued a close request
- ❑ Terminate: Connection has been terminated
- ❑ Status Response: Here is the connection status
- ❑ Error: Reports service request or internal error

# TCP Mechanisms

- Connection Establishment
  - Three way handshake
  - SYN flag set  $\Rightarrow$  Request for connection



- q Connection Termination
  - q Close with FIN flag set
  - q Abort

# Data Transfer

- ❑ Stream: Every byte is numbered modulo  $2^{32}$ .
- ❑ Header contains the sequence number of the first byte
- ❑ Flow control: Credit = number of bytes
- ❑ Data transmitted at intervals determined by TCP  
Push  $\Rightarrow$  Send now
- ❑ Urgent: Send this data in ordinary data stream with urgent pointer
- ❑ If TPDU not intended for this connection is received, the “reset” flag is set in the outgoing segment

# Implementation Policies (Choices)

- Send Policy:

Too little  $\Rightarrow$  More overhead. Too large  $\Rightarrow$  Delay  
Push  $\Rightarrow$  Send now, if possible.

- Delivery Policy:

May store or deliver each in-order segment.  
Urgent  $\Rightarrow$  Deliver now, if possible.

- Accept Policy:

May or May not discard out-of-order segments

# Implementation Policies (Cont)

- Retransmit Policy:

  - First only

  - Retransmit all

  - Retransmit individual

  - (maintain separate timer for each segment)

- Ack Policy:

  - Immediate (no piggybacking)

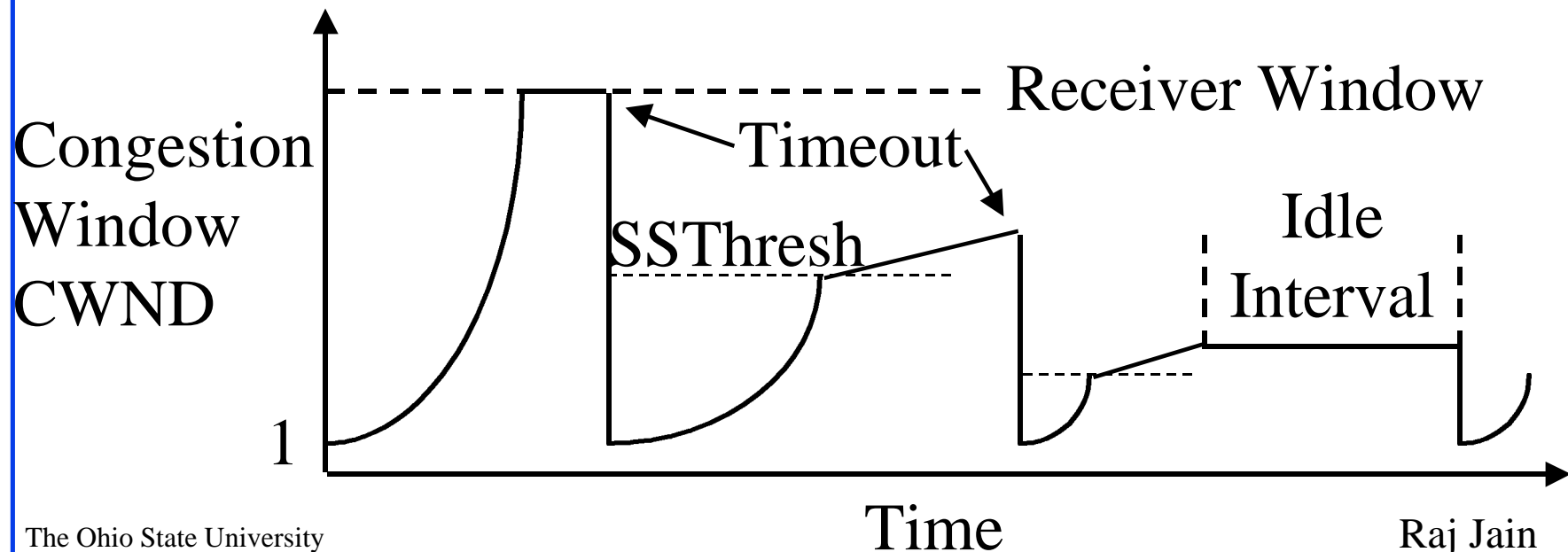
  - Cumulative (wait for outgoing data or timeout)

# Slow Start Flow Control

- ❑ Window = Flow Control Avoids receiver overrun
- ❑ Need congestion control to avoid network overrun
- ❑ The sender maintains two windows:
  - Credits from the receiver
  - Congestion window from the network
  - Congestion window is always less than the receiver window
- ❑ Starts with a congestion window (CWND) of 1 segment (one max segment size)
  - ⇒ Do not disturb existing connections too much.
- ❑ Increase CWND by 1 every time an ack is received

# Slow Start (Cont)

- If packets lost, remember slow start threshold (SSThresh) to  $CWND/2$   
Set  $CWND$  to 1  
Increment by 1 per ack until SSThresh  
Increment by  $1/CWND$  per ack afterwards



# Slow Start (Cont)

- ❑ At the beginning,  $SSThresh = \text{Receiver window}$
- ❑ After a long idle period (exceeding one round-trip time), reset the congestion window to one.
- ❑ Exponential growth phase is also known as “Slow start” phase
- ❑ The linear growth phase is known as “congestion avoidance phase”

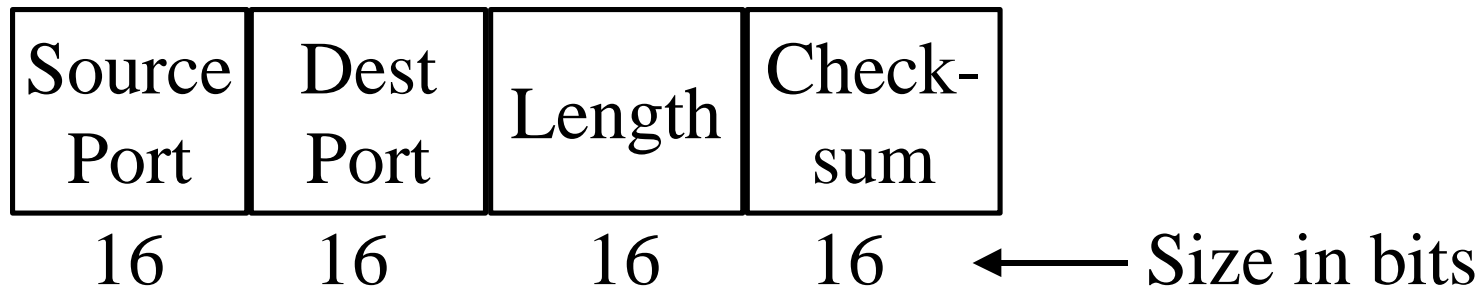


## FRR (Cont)

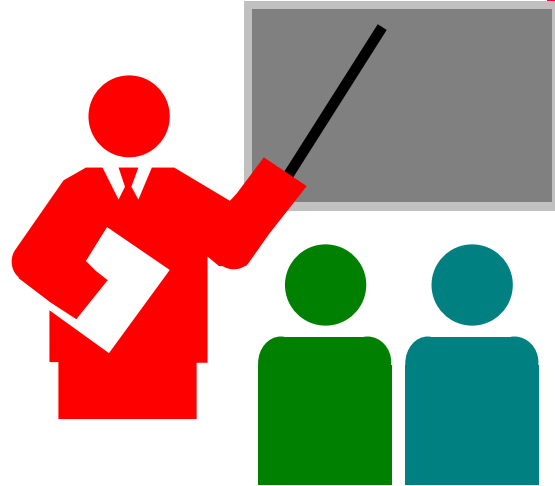
- ❑ Upon receiving the third duplicate Ack:
  - Set  $SS_{Thresh}$  to  $1/2$  of current  $CW_{ND}$
  - Retransmit the missing segment
  - Set  $CW_{ND}$  to  $SS_{Thresh}+3$
- ❑ For each successive duplicate Ack:
  - Increment  $CW_{ND}$  by 1 MSS
  - New packets are transmitted if allowed by  $CW_{ND}$
- ❑ Upon receiving the next (non-duplicate) Ack:
  - Set  $CW_{ND}$  to  $SS_{Thresh} \Rightarrow$  Enter linear growth phase
- ❑ Receiver caches out-of-order data.

# User Datagram Protocol (UDP)

- ❑ Connectionless end-to-end service
- ❑ No flow control. No error recovery (no acks)
- ❑ Provides port addressing
- ❑ Error detection (Checksum) optional. Applies to pseudo-header (same as TCP) and UDP segment. If not used, it is set to zero.
- ❑ Used by network management



# Summary



- ❑ TCP provides reliable full-duplex connections.
- ❑ TCP Streams, credit flow control
- ❑ Slow-start, Fast retransmit/recovery
- ❑ UDP is connectionless and simple.  
No flow/error control.

# Homework

- ❑ Read RFCs 0768 (UDP) 0793 (TCP), 2001 (Slow start and FRR)  
All RFCs can be found on <ftp://ftp.isi.edu/in-notes/>
- ❑ Read Sections 17.3 and 17.4 of Stallings' book
- ❑ Submit answers to problem 17.15