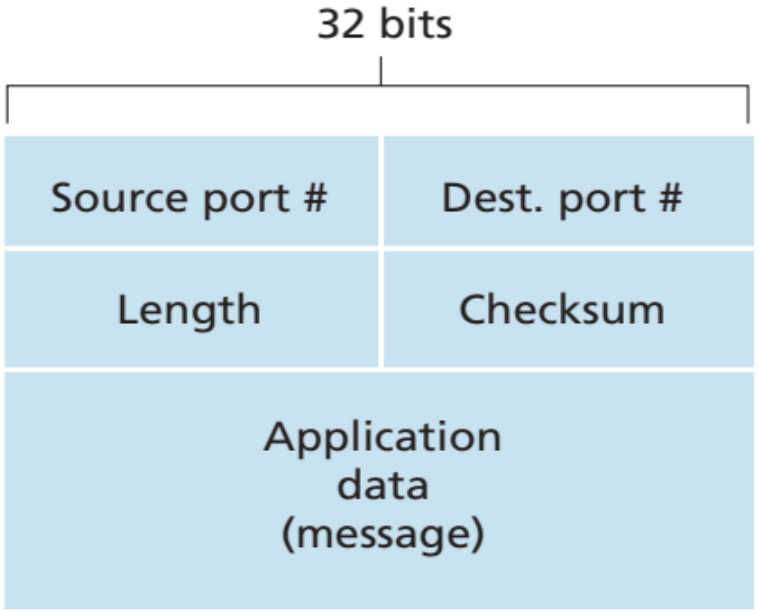
Module 3: Transport Layer (Lecture – 2)

Dr. Nirnay Ghosh
Assistant Professor
Department of Computer Science & Technology
IIEST, Shibpur

UDP Checksum

- Determines if the bits in the UDP segment have been altered as it moved from source to destination
- Bits may be altered due to noise in the links or while stored in the router memory
- Sender side UDP: performs 1's complement sum of all 16-bit words in the segment
  - Overflow encountered during sum is wrapped around
- The result is put into the checksum field of the UDP segment
- Receiver side UDP: all 16-bit words (including the checksum) are added
  - No error: sum will be all 1's
  - Otherwise, at least one bit is changed



UDP Segment Structure

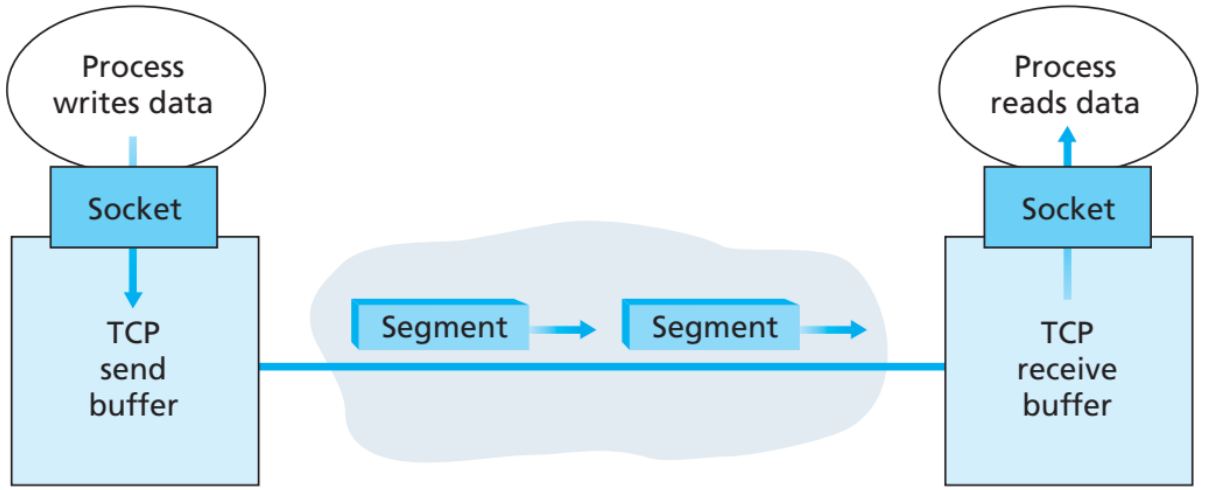
- UDP error detection: essential as neither link-by-link reliability nor router in-memory detection is guaranteed
- Provides error detection but no recovery
  - Damaged segment is discarded or passed to the application with a warning

Reliable Data Transfer Mechanisms

Mechanism	Use, Comments
Checksum	Used to detect bit errors in a transmitted packet.
Timer	Used to timeout/retransmit a packet, possibly because the packet (or its ACK) was lost within the channel. Because timeouts can occur when a packet is delayed but not lost (premature timeout), or when a packet has been received by the receiver but the receiver-to-sender ACK has been lost, duplicate copies of a packet may be received by a receiver.
Sequence number	Used for sequential numbering of packets of data flowing from sender to receiver. Gaps in the sequence numbers of received packets allow the receiver to detect a lost packet. Packets with duplicate sequence numbers allow the receiver to detect duplicate copies of a packet.
Acknowledgment	Used by the receiver to tell the sender that a packet or set of packets has been received correctly. Acknowledgments will typically carry the sequence number of the packet or packets being acknowledged. Acknowledgments may be individual or cumulative, depending on the protocol.
Negative acknowledgment	Used by the receiver to tell the sender that a packet has not been received correctly. Negative acknowledgments will typically carry the sequence number of the packet that was not received correctly.
Window, pipelining	The sender may be restricted to sending only packets with sequence numbers that fall within a given range. By allowing multiple packets to be transmitted but not yet acknowledged, sender utilization can be increased over a stop-and-wait mode of operation. We'll see shortly that the window size may be set on the basis of the receiver's ability to receive and buffer messages, or the level of congestion in the network, or both.

Connection-oriented Transport: TCP

- TCP: Internet's transport-layer protocol, reliable, connection-oriented
- Intermediate network elements (routers and link-layer switches) do not maintain TCP connection state
- TCP connection: point-to-point, full-duplex
  - Between a single sender and a single receiver
  - Processes running at two end systems can communicate simultaneously
  - Consists of two sets of buffers, variables, and sockets to connect two processes running on two end stations
- Connection-oriented: three-way handshake protocol between client and server
  - Client send a special TCP segment (no application-layer data)
  - Server responds with a second special TCP segment (no application-layer data)
  - Client responds again with a third special segment (may carry payload)
- Once the TCP connection is established the two applications can send data

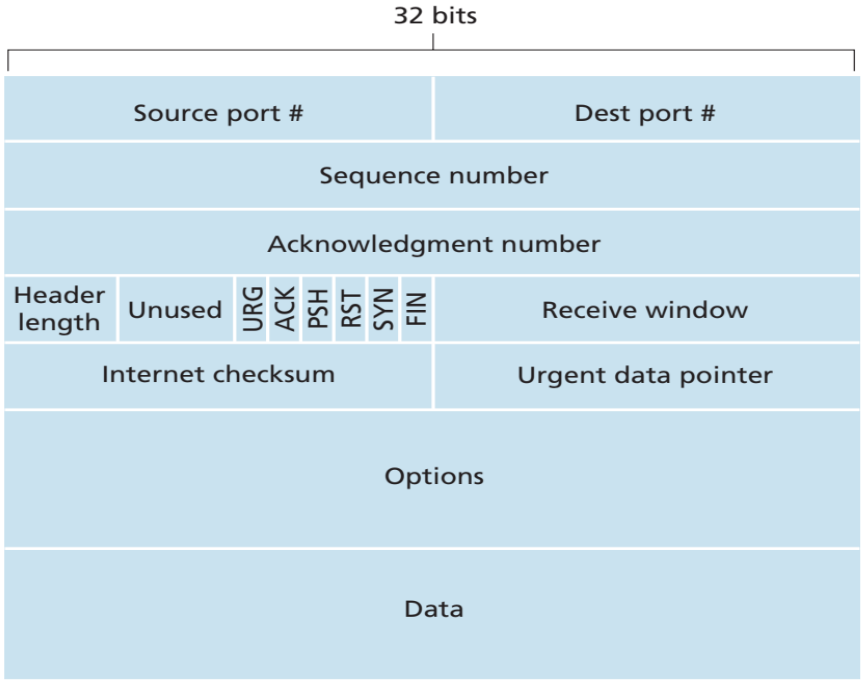


TCP Send and Receive Buffers

- Client process: passes a stream of data through the socket to the TCP running in the client
  - TCP directs this data to the connection's send buffer
- TCP will grab a chunk of data from the send buffer and pass it on the network layer
  - Maximum segment size (MSS): maximum amount of data that be placed in a TCP segment
- Server process: receives a segment and places its data in the TCP connection's receive buffer

TCP: MSS & Segment Structure

- Maximum Segment Size (MSS)
  - Typically set by determining the length of the largest link-layer frame (called maximum transmission unit (MTU)
  - Ensures that a TCP segment (encapsulated in an IP datagram) containing data and header (typically 40 bytes) fit into a single link-layer frame
  - Both Ethernet and PPP link-layer protocols have an MSS of 1500 bytes
  - It is the maximum amount of the application data
- TCP Segment Structure
  - Source and Destination port numbers (16-bit each): used for multiplexing/demultiplexing data from/to upper-layer applications
  - Sequence number (32-bit) & Acknowledgement number (32-bit): used by the TCP sender and receiver in implementing a reliable data transfer service
  - Receive window (16-bit): used for flow control – used to indicate the number of bytes that a receiver is willing to accept

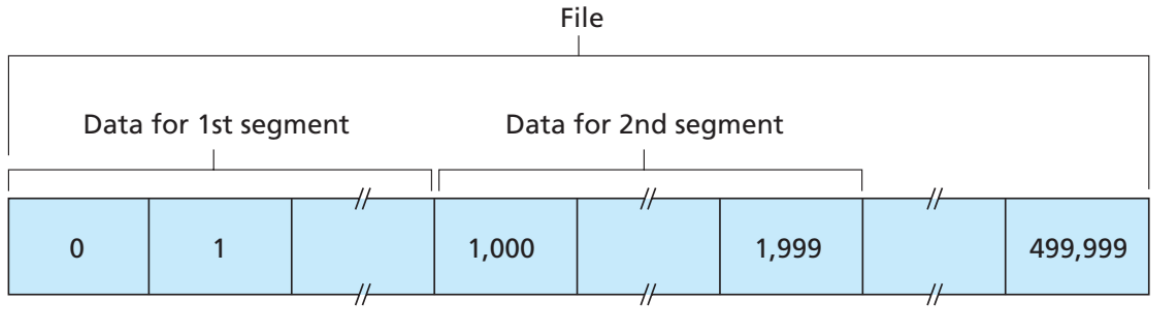


TCP Segment Structure

- Header length (4-bit): length of the TCP header as multiples of 32-bit word
- Flags (6-bit):
  - ACK bit: indicates the value carried in the acknowledgement field is valid
  - RST, SYN, FIN bits: used for connection setup and teardown
  - PSH bit: indicates that the receiver should pass the data to the upper layer immediately
  - URG bit: data in this segment is marked 'Urgent' by the sending side upper layer
- Urgent data pointer (16-bit): indicates the location of the last byte of this urgent data to the receiving side upper layer
- Options (32-bit): optional and variable-length field – used when a sender and receiver negotiate the MSS or a window scaling factor used for high-speed networks

TCP: Sequence Numbers & Acknowledgement Numbers

- Critical part of TCP's reliable data transfer service
- TCP views data as an unstructured, but ordered stream of bytes
- Sequence number:
  - Inserted in the sequence number field in the header of the TCP segment
  - Assigned over the stream of transmitted bytes and not over the series of transmitted segments
  - Byte-stream number of the first byte in the segment
  - Example: MSS is 1000 bytes and sequence numbers of first and second segments are 0 and 1000, respectively

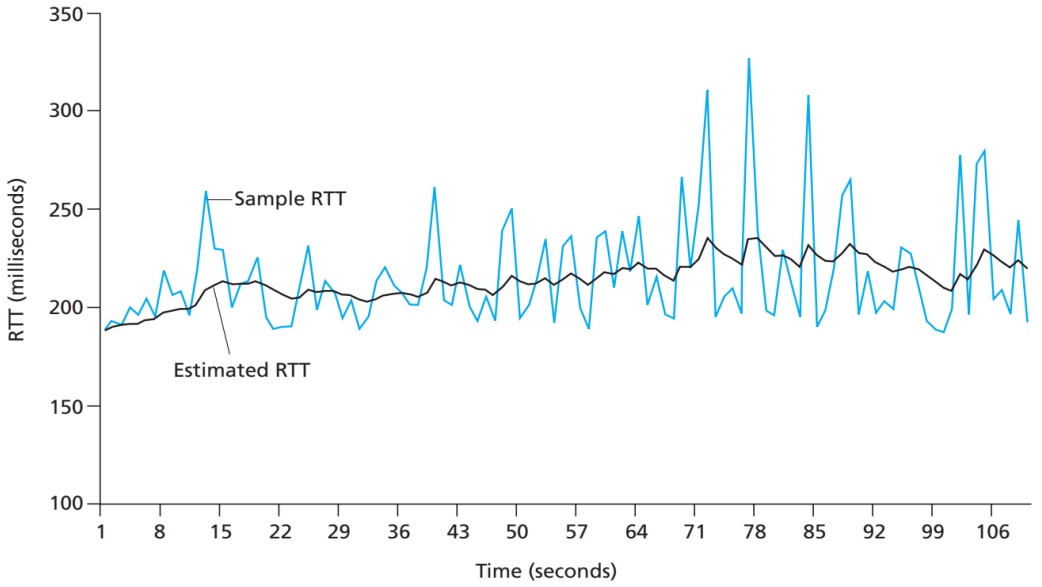


Dividing Data File into TCP Segments

- Acknowledgement number:
  - Inserted into the TCP segment by a receiving host
  - It is the sequence number of the next byte it is expecting from the sending host
  - Example: Acknowledgement number in the TCP segment sent by the receiver after receiving data for the second segment is 2000

TCP: Round Trip Time Estimation & Timeout

- TCP uses a timeout/retransmit mechanism to recover from lost segments
- Implementation challenge: how much larger the timeout should be than the connection's round trip-time (RTT)?
- Estimating RTT
  - *SampleRTT*: amount of time elapsed between when the segment is sent and the corresponding acknowledgement is received
  - It is measured for the segments which have been transmitted once
  - It fluctuates from segment to segment due to congestion in the routers and varying loads on the end systems
  - Estimation: take weighted combination of the previous value *of EstimatedRTT* and the new value of *SampleRTT*
  - Exponential Weighted Moving Average (EWMA): weight of a given *SampleRTT* decays exponentially fast as the updates proceeds



RTT Samples and RTT Estimates

$$\text{EstimatedRTT} = (1 - \alpha) \cdot \text{EstimatedRTT} + \alpha \cdot \text{SampleRTT}$$

- **Recommended value for α = 0.125**

- *EstimatedRTT*: smoothens the variations in the *SampleRTT*
- *DevRTT*: Variability of the RTT
  - Estimates of how much *SampleRTT* typically deviates from the *EstimatedRTT*
  - EWMA of the difference between *SampleRTT* and *EstimatedRTT*

$$\text{DevRTT} = (1 - \beta) \cdot \text{DevRTT} + \beta \cdot |\text{SampleRTT} - \text{EstimatedRTT}|$$

- **Recommended value for β = 0.125**