# Lesson 01

## Conventional Transport Layer Protocols— User datagram protocol (UDP) and Transmission control protocol (TCP)

# Conventional Transport Control Protocols

- Application-data first encodes using the application-layer protocol header-words which prefix over the data
- Then the encoded data from the application layer encodes again using the transport-layer protocol header-words by prefixing them over the previously encoded data

# Encoding and Decoding

- Encoding at Layers on Transmission and Decoding on Reception

Refer Figure 6.1

# Conventional Transport Control Protocols

- At the receiver end the reverse process of decoding at each layer to retrieve back the application data takes place

# Conventional Transmission Control Protocols– UDP and TCP

- UDP and TCP
- The data transported from the transport layer to next layer (*L*3) using TCP (or UDP in case of datagram)
- TCP– a transport layer protocol for the Internet

# TCP

- A connection oriented protocol
- Session for establishment, data flow and congestion control, and session termination in TCP

# Port

- Means a service access point (software) for data input and output
- A service (application) is rendered by a node
- Examples of services through the Internet are e-mail transmission, e-mail reception, and web browsing
- Example of a port– port number 80 which specifies the www (HTTP) application in the TCP/IP protocol suite

# User Datagram Protocol

- A connectionless protocol– there is no session for establishment, data flow and congestion control, and session termination in UDP
- Transmits like a person using a phone who just speaks without waiting, irrespective of whether the receiver at the other end is listening or not, replying or not

# Function of the transport layer

- To transport the port data
- UDP header specifies the ports
- Used at the subsequent layers (from the transmitter transport layer up to the receiver transport layer) during transmission of port data (application layer data) to the receiver

# UDP

- Useful in transmitting datagrams
- One datagram length ≤ $2^{16}$ words
- Usage Examples– as those for multicasting, registration request and registration reply

# UDP header fields

- First-word bits *b*0–*b*15 for source port (optional) and *b*16–*b*31 for destination port
- Second-word bits *b*0–*b*15 for length of the datagram and *b*16–*b*31 for the header checksum

# Pseudo header after the header

- Used as prefix to datagram
- First word– source IP address
- Second word– destination IP address
- Third word for the protocol and length– bits b0–b7 are all zeros, b8–b15 specify the network layer protocol, and b16–b31 specify the UDP length

# UDP Pseudo Header

- Pseudo header enables identification of source and destination IP addresses of the ports, for example, when a datagram for registration request or registration reply is sent

# UDP Pseudo Header

- Enables identification of the protocol to be used to route the datagram
- For example, ‘protocol = 17’ shows the use of the IP protocol by the network layer

# UDP Pseudo Header

- Also enables specification of the length up to which the UDP header and data extends so that the remaining part of datagram may be used for conveying other information

# Transmission of data streams

- A segment from higher layer transmitted as a data stream of bytes to lower layer IP where it is packetized using IP protocol
- Octets (bytes of the words) transmit sequentially

# Receiving of data streams

- Octets received by the transport layer at the other end in the same sequence as they are sent

# TCP header inclusion of sequence number

- A sequence number is first assigned to each byte (octet) before a TCP connection transfers the data
- A set of 32 bits of a word (4 bytes) has four sequences

# TCP header inclusion of sequence number

- The application layer data is transmitted and received as a stream consisting of sequences
- The TCP header includes the sequence number of the first byte (not of each byte) to be transmitted through the stream

# Buffering

- The TCP transmitter buffers the segment(s)
- Receiver acknowledges a data-sequence
- The transmitter empties the bytes up to the acknowledged sequence from the buffer

# Retransmission

- From the byte next to the data sequence number of the last successfully transmitted byte
- Also retransmits in case no acknowledgement received within a timeout period

# Fully acknowledged end-to-end

- Acknowledgement of Session-start, data transfer, and session-finish
- Handshaking of packets– for acknowledgement
- All transactions (connection start, establishment, data streaming, and finish) acknowledged

# In-order delivery

- The TCP transport layer delivers the segments in sequential order When the *n*$^{th}$ segment has been delivered, only then is the ($n$ + 1)$^{th}$ delivered to the application layer and not before that

# Congestion control

- Congestion– lack of acknowledgement within the timeout period in a step or receipt of DACK (duplicate acknowledgement)
- Number of methods to control congestion

# A method of controlling congestion

- To slowly increase exponentially the number of bytes in the stream in successive steps
- The number of transmitted bytes in a step reduced to half on encountering congestion

# TCP Header

- First word: The function of the transport layer is to transport the port data
- Therefore, the first 16 bits (*b*0–*b*15) are for the source port number and the next 16 (*b*16–*b*31) for the destination port number

# TCP Header

- Second word: This 32-bit field defines a 32-bit sequence number.
- The sequence number field reset to 0 in case there are no more octets left to transmit from the segment

# TCP Header

- Using the sequence numbers (data byte number sent in the sequence), the packets, which reach non-sequentially, are reassembled and the receiver sets the acknowledgment number field as per the sequences successfully received by it

# TCP Header

- Third word: It gives a 32-bit acknowledgement number (interpreted when the *A*-flag is set)— the value of the next sequence number (byte number), which the sender of the segment is expecting to receive from the receiver in case the sequences of bytes has been received successfully

# TCP Header

- Acknowledgement number helps the receiver in knowing how much of the segment data was successfully transported
- Helps the transmitter in knowing from which sequence number is the data now to be sent to the receiver

# TCP Header

- When the connection is established, both, the transmitter and the receiver, use the field to inform each other about the upcoming sequences

# TCP Header

- Fourth word: First 4 bits (*b*0–*b*3) are for the data offset and the next 6 bits (*b*4–*b*9) are reserved
- Reserved bits facilitate the addition of more flag bits or provide a provision for extension of the present functions and features of TCP in any subsequent changes in the recommendations for TCP

# Field for offset

- The offset field specifies the word from where the application layer data octets will begin (after the header, options, and padding)

# Fields for flags and window size

- The next 6 bits (*b*10–*b*15) are flags (*F*, *S*, *R*, *P*, *A*, and *U*) and the next 16 bits (*b*16–*b*31) are for setting window size, which is used for congestion control during transport
- The window size field specifies the number of bytes the sender is willing to receive starting from the acknowledgement field value

# TCP Header

- Fifth word: The first 16 bits (*b*0–*b*15) for checksum of the header and the data and the next 16 bits (*b*16–*b*31) are for the urgent field
- Urgent field interpreted when the U-flag has been set during segment transport

# Urgent field

- Communicates an offset value to be added to get a sequence number from the sequence number of the present segment
- The value then points to the sequence number of the octet (in application layer data) following the urgent data
- Helps in data flow control by specifying the urgent part of the data octets for transporting to the other end

# TCP Header

- Sixth and subsequent header words: These are used for options (each of 8 bits) and padding

# Flags

- U– URG, set means urgent pointer field is being used for control of data flow
- A– ACK is set means acknowledgement number word field has significance and can be used to control flow and congestion
- *P* – PSH set means push the data

# Flags

- *R*– RST set means reset the connection (sequence from the beginning)
- *S*– SYN set means synchronize the sequence number
- *F*– (FIN) set means data is finished (no more data for sending at present)

# Summary

- UDP for sending datagrams using connectionless protocol
- TCP for sending segment of data from application layer using connection oriented protocol
- Sessions for establishment, data flow and congestion control, and session termination in TCP

# … Summary

- TCP header fields for port numbers of source and destination, sequence number, acknowledgement number, offset, flags and window size, check-sum and urgent sequence offset
- TCP header options and paddings

## End of Lesson 01

**Conventional Transport Layer Protocols— User datagram protocol (UDP) and Transmission control protocol (TCP)**