Mobile Transport Layer

Lesson 02

TCP Data Stream and Data Delivery

TCP Data Stream

- Consists of bytes
- Delivered using a virtual connection between sockets
- Each socket has the port number and IP address for rendering service to an application

TCP Data Stream

- The number of bytes = equal to the transport protocol data units (TPDU) that are placed in a memory buffer and transmitted by the TCP protocol
- A TCP segment defines the TPDUs in a TCP data stream
- PDU depends on MTU (maximum transfer unit) presentable in a given network state

TCP Protocol

- Employs timers to synchronize the data stream between both ends
- Provides a delivery mechanism for data streams

RTT (Round Trip Time)

- The time interval between the start of the transmission to the receiver layer and receiving the acknowledgement of successful transmission
- To set the time interval during which the transmitter expects the receiver to send a TCP stream with the acknowledgement field set to indicate the successfully sent bytes

RTT (Round Trip Time)

- The transmitter retransmits from the acknowledge-field sequence number onwards

TCP Data Delivery

- In TCP there is acknowledgment of sequences sent from one end to the other. (The acknowledgement field specifies the sequenced byte number from which the receiver end expects the sender end to transmit in the next sequence.)
- The bytes that do not reach the receiver successfully within a timeout period are retransmitted

Checksum field

- Detects errors
- Receiver discards any erroneous packets in a data stream
- Transmitter started retransmitting as per the previous acknowledgement and thus it retransmitted the delayed octets
- Receiver discards the duplicate packets

TCP Data Delivery

- Retransmission in case of no acknowledgement
- To ensure reliable, guaranteed, and error-free data transfer
- Unlike UDP— in which transmission is with no buffering, no acknowledgement, and no retransmission

Similarity to the transparent and non-transparent data transmission

- No retransmission mechanism similar to the transparent data transmission Retransmission mechanism similar to the non-transparent data transmission in the radio layer of a GSM system

Step 1 for delivering the data

- Step 1: Connection establishment during four states LISTEN, SYN_SENT, SYN_RECEIVED, and ESTABLISHED
- Three data streams are transferred in the last three phases
- Third one– with the data octets

Step 2 for delivering the data

- Step 2: Data stream of application layer octets transferred after establishment of the connection
- Before termination of the connection on completion of the transmission data octets

Step 3 for delivering the data

- Step 3: Connection Finish and then Close

STATE LISTEN OF TCP CONNECTION BETWEEN THE TCP_A AND RECEIVER TCP_B

- LISTEN: TCP_B receiver waiting for listening a connection request from TCP_A

State SYN_SENT

- SYN_SENT: A SYN data stream sent by the transmitter TCP$_A$ with the *S* (SYN) flag set to 1
- Waiting for the TCP$_B$ data stream with SYN and ACK

State SYN-RECEIVED

- SYN-RECEIVED: A SYN_ACK*B* sent by receiver data stream TCP$_B$ for TCP$_A$ with the *S* (SYN) and *A* (ACK) flags set to 1
- TCP$_B$ uses the acknowledgment field at the packet and sends an acknowledgment to TCP$_A$

Example

- SYN sequence number sent in SYN_SENT = *n*
- Then TCP_B transmits an acknowledgement with the sequence number in the acknowledgement field set as *n* + 1 and the *A*-flag set to 1 in SYN-RECEIVED state

Example

- TCP_A expects the sequence number request n + 1 from TCP_B
- TCP_B sequence number field has sequence number m in it when TCP*B* sends a sequence m to TCP*B*

State DATA _TRANSFER

- DATA _TRANSFER: A TCP_A stream always has the sequence number field for the byte starting from which the data of a segment is being transferred through the stream and the acknowledgement number field for the byte starting from which the data of a segment is expected by TCP_B

State DATA _TRANSFER

- TCP$_B$ stream always has the sequence number field for the byte starting from which the data of a segment is being transferred through the stream and the acknowledgement number field for the byte starting from which the data of a segment is expected in a TCP$_A$ stream

State DATA_TRANSFER

- TCP$_B$ acknowledges a data stream by setting the acknowledgement field value to the sequence number of the last byte received successfully (from other end) plus 1
- TCP$_A$ is expected to send the new data stream starting from this (last successfully received byte + 1) position

Example

- A segment of 4020 bytes with sequence numbers from 1025 to 5044 is to be transmitted
- If the TCP_A layer begins transmitting 4020 bytes with a sequence number field = 1025
- All bytes are transmitted without loss and without error, then the TCP_B stream acknowledgement number field = 5045

Example

- If only the bytes up to sequence number 2047 received successfully by the TCP$_B$ layer in RTT, then the receiver will send the TCP$_B$ stream with an acknowledgement number field = 2048, as the receiver expects the transmitter to send bytes starting from sequence number 2048

State Data _Transfer of TCP connection between the TCP_A and TCP_B

- Acknowledgements for data sent to the receiver, or the lack of acknowledgements from the receiver within a timeout period– Used by the sender to judge the present network conditions between the TCP sender and receiver

STATE FIN_WAIT_1 OF TCP CONNECTION BETWEEN THE TCP_A AND TCP_B

- FIN_WAIT_1: TCP_A sends a data stream with the *F*-flag (FIN) set to 1 and expects a FIN_ACK from TCP_B within a wait period

TIME_WAIT STATE IN FIN_WAIT_1

- Used to wait for FIN_ACK (acknowledgement of FIN) from the other end
- After that the state changes to FIN_WAIT_2, else if the FIN_ACK received from TCP$_B$, then the receiver TCP$_B$ state changes to CLOSING

State FIN_WAIT_1

- FIN_WAIT_2: TCP$_A$ sends a data stream with the *F*-flag (FIN) set to 1 and expects a FIN_ACK from TCP$_B$ during a wait period

TIME_WAIT STATE IN FIN_WAIT_2

- TIME_WAIT: Used to wait for FIN_ACK (acknowledgement of FIN) from the other end and after that if no FIN_ACK is received, then the state changes (after timeout) to CLOSED
- If FIN_ACK received then the TCP$_B$ state changes to CLOSING

CLOSE_WAIT state

- CLOSE_WAIT: TCP$_A$ goes to CLOSE_WAIT state on receipt of the FIN_ACK in either FIN_WAIT_1 or CLOSE_WAIT, LAST_ACK is sent after CLOSE_WAIT to finally close the connection

States CLOSING and CLOSED

- CLOSING: TCP$_B$ sends a CLOSING ACK to TCP$_A$, waits for timeout, and goes to the CLOSED state

Summary

- TCP
- TCP acknowledgment of sequences sent from one end to the other
- The acknowledgement field specifies the sequenced byte number from which other end expects the sender end to transmit in the next sequence
- RTT

. . .

… Summary

- Connection establishment during four states LISTEN, SYN_SENT, SYN_RECEIVED, and ESTABLISHED
- Then Data transfer
- Then Finish and Close

End of Lesson 02
TCP Data Stream and Data Delivery