Mobile Transport Layer

Lesson 03

TCP Data flow control and Congestion control

TCP Data Flow Control

- Octets in a segment transmit as data streams which are transmitted after packetizing
- A few packets of a data stream may reach the other end with an error, may be lost, or may not reach in the expected time
- They need retransmission starting from the octet succeeding the last successfully received one. A controlled data flow prevents need of large number of retransmissions

Window size adjustment (TCP tuning) Method

- Adjusted and throughput depends on RTT (round trip time) interval for the acknowledgement
- The transmitter transmits all bytes received up to a sequence number specified by *w* at the window size field of the other end

Window size adjustment (TCP tuning) Method

- Transmit from i to $i + w$ in case the next sequence number (in the acknowledgement field from receiver) to be sent from transmitter is i

Timeout

- At the transmitter sets equal to the RTT for acknowledgement
- If there is no acknowledgment in the timeout period
- Then the bytes sent using the window are considered as lost and are retransmitted

Cumulative acknowledgement

- Receiver acknowledges all bytes received up to a sequence number defined in the acknowledgement field

Cumulative acknowledgement

- Cumulative acknowledgement from the TCP_B end called partial acknowledgement (PACK) if the acknowledgement field shows a lesser value than expected after transmission of all the bytes from the TCP_A end
- PACK— shows the lost packets or data during transmission to the TCP_B end

Reverse packet acknowledgement

- Reverse packet from the receiver piggybacks the acknowledgement

Duplicate acknowledgement (DACK)

- One of the packets of a segment may reach after a delay in comparison to the packet succeeding it
- Acknowledgement from the receiver is duplicated by retransmission of an earlier acknowledgement without delay
- DACK thus shows recovery of lost packets or data received after a delay at the TCP_B end

Delayed acknowledgement

- Acknowledgement from the receiver is delayed if the receiver responds by including, after the TCP header, a large number of octets of the receiver-end segment
- Due to packetization time at the receiver network layer and the packets tracking different paths and hops to the transmitter

Delayed Acknowledgement

- Receiver has started receiving another transmitted data stream from the next sequence number at the transmitter
- Receiver now assigned a dual role at the same time
- Delay at the transmitter can be set at 100 ms to 250 ms in order to reduce the number of delayed acknowledgements

Adjusting the window-size field method

- Window-size field is varied for congestion control during data flow
- Window size, *w*, a 16-bit field specifies the number of bytes the receiver (TCP_A or TCP_B) is ready to receive from the sender (TCP_B or TCP_A, respectively
- *w* set to maximum 2^{16} bytes

TCP tuning

- Method of specifying window size
- The bytes (from the other end) start from the acknowledgement field value *i* and are transmitted up to the *i* + *w* value of the sequence number in a TCP segment
- Refer Examples 6.1 and 6.2 in the text

Window scaling method

- During high speed data transfer, the window scaling method
- The TCP tuning method scales up the widow size to 2^{30} bytes in case of high-speed networks

Window scaling method

- Number of bits, which left shifts the bits *w* at the window-size field, is set during a three way handshake session SYN_SENT, SYN_RECEIVED and ESTABLISHED
- A left shift of 1 multiplies *w* by 2, a left shift of 2 multiplies *w* by 4, and so on

Window scaling method

- Number of shifts can be between 0 and 14 and the window size = $2^{30 + s}$, where s is the number of shifts
- Window scaling factor, $s_w = 2^s$

Sliding Window Method

- A window specified from a sequence number to another sequence number
- S_{A0} and S_{A1} are the two sequence numbers which define the lower and upper boundaries of the window. $S_{A0} – S_{A1} = S_W$ (sliding window size)

Sliding Window Method

- TCP_A can send only S_W bytes before it must wait for an acknowledgment and a window update from the receiving end TCP_B

Example

- TCP_A transmit as per its present sliding window setting
- The setting such that the TCP_A data stream is transmitted between S_{A0} and S_{A1} before an acknowledgement is expected from the receiving end

Example

- TCP_A sets the sliding window to the next set of sequence numbers $S_{A1} – S_{A2}$.
- When the receiver's acknowledgement is not received in a specified interval, then the window slides back to the original and the sequences between S_{A0} and S_{A1} are retransmitted

Example

- When the receiver's acknowledgement is received in the specified interval and it equals S'_{A0}, then the window slides back and sequences between S'_{A0} and $S'_{A0} + S_W$ is retransmitted
- Sliding window size, S_W, also defines an acknowledgement delay period

RTT

- Assume that one sequence is transmitted and acknowledged in time RTT_0, then the next acknowledgement is expected after $RTT_0 \times S_W$
- S_W specified in the TCP_A data stream at the window-size field
- TCP_B receives the stream and sends the acknowledgement number field after setting the acknowledging time interval as per the S_W value received from TCP_A

Congestion Control

- A TCP data stream receiver sets a window
- The data from TCP_A (transmitter) to be sent from sequence number S_0 and S_4
- Let us assume that the data from a transmitter is received and sent to an application or a service access point (SAP) up to sequence number S_0

Congestion control

- Assume that at an instant, the situation is as follows—transmitted TCP_A octets are received and acknowledged by TCP_B receiver up to the sequence number S_1
- But data is still being sent to the receiver SAP (application layer)
- TCP_A octets transmit from sequence number $S1 + 1$

Congestion control

- Let us now assume that at the next instant, the situation is as follows—the data from TCP_A is received at TCP_B up to sequence number S_2 but has not yet been acknowledged and that it is yet to be sent to SAP

Congestion control

- TCP_A octets retransmit from sequence number $S1 + 1$ after the timeout
- The data from the transmitter can be received up to a maximum of sequence number S_3 when the window of the receiver extends from S_1 to S_3 and the window-size field bits in the TCP_A data stream header equal $(S_3 - S_1) \div s_w$, and s_w is 1

Congestion control

- Congestion network window size (cnwd) = $(S_3 – S_1)$ and the window is for $(S_3 – S_1)$ octets and when s_w = 2 the window is for $[(S_3 – S_1) \div 2]$ octets
- The window scaling factor is set at the transmitter
- By adjusting s_w the congestion can be controlled

Methods for congestion control in conventional TCP

- Slow start and congestion avoidance
- Fast recovery (in place of slow start) after packet loss

The problems faced while employing the conventional TCP on a mobile network

(a) Slow start method employed in the conventional TCP presumes that a packet is lost due to congestion, not due to any other reason

- Mobile network– The transmission quality problem more likely reason for packet loss

The problems faced while employing the conventional TCP on a mobile network

(b) Mobile network– BERs high, which leads to high transmission repetition rates and, therefore, the higher window slide-back rates (Transmission repeats due to windows slide back in TCP)

The problems faced while employing the conventional TCP on a mobile network

(c) The duplicate acknowledgements (DACKs)– lead to reduced window sizes

Methods for Wireless and Mobile Networks

- Split TCP– TCP splits into two layers
- The upper layer to take care of requirements in mobile networks and send the data streams to the conventional TCP layer
- Four methods using split TCP– indirect TCP, selective repeat TCP, mobile-end Transport, and mobile TCP

Methods for Wireless and Mobile Networks

- Fast retransmit and fast recovery
- Selective acknowledgement
- Explicit congestion notification

TCP-aware Link-Layer Methods for Wireless and Mobile Networks

- The three TCP-aware link layer protocols – snooping TCP, WTCP, and delayed duplicate acknowledgement protocol
- Snooping– secretly looking into or examining something
- The data-link layer snoops into the TCP layer data

Link-layer (L2) modification methods for Wireless and Mobile Networks

- Data-link layer in the mobile nodes (MNs) FEC and ARQ methods (in place of methods like L4 TCP window sliding method) for error control
- FEC: FEC code lengths and frame sizes can be varied depending on the bit error rate (BER)

Link-layer (L2) modification methods for Wireless and Mobile Networks

- ARQ (automatic repeat request): When errors are detected, a repeat request is generated
- The throughput is not affected in case there are no errors
- However, in case of errors, the throughput, round trip time (RTT), and congestion in the network affected due to repeated retransmission of data streams

Link-layer (L2) modification methods for Wireless and Mobile Networks

- When using adaptive techniques, retransmission can be limited and can be varied depending on the bit error rate (BER)
- For example, in voice communication, a certain BER is tolerable
- Hence, retransmission can be skipped

SUMMARY

- Data flow
- Window size adjustment
- Cumulative partial acknowledgement
- Reverse packet acknowledgement
- Delayed acknowledgement
- Adjusting window size field
- window scaling method
- Sliding window method

. . .

...Summary

- Slow start and slow-start fast-recovery
- Wireless and mobile network methods
- Split TCP methods for congestion control in wireless and mobile networks
- TCP aware link layer methods
- Link layer modification methods
- Explicit notification methods

End of Lesson 03

TCP Data flow control and Congestion control