第四次作业

1

```
X0 = 1000  |  X1 = 0
X2 = 500   |  X3 = 1001
X4 = 1001  |  X5 = 501
Some message along with X6 = 1001 and X7 = 501
```

TCP establishes a connection through a three-way handshake. During this process, the side that sends the initial SYN packet performs an active open, while the side that receives this SYN and replies with a SYN-ACK performs a passive open.

Let's illustrate this with a client initiating a connection to a server.

1. The client sends a SYN packet (synchronization segment) to request a connection. The initial sequence number (ISN) is a randomly generated value, and the acknowledgment number is 0.
2. The server receives the SYN packet and responds with a SYN-ACK. The acknowledgment number is the client's ISN plus 1, and the sequence number is a randomly generated value.
3. The client receives the SYN-ACK packet and sends an ACK back to the server. The acknowledgment number is the server's ISN plus 1, and the sequence number is the acknowledgment number from the previous packet.

```
Round 1 begin: cwnd = 1 × 2 KB, rwnd = 64 KB
Round 2 begin: cwnd = 2 × 2 KB, rwnd = 62 KB
Round 3 begin: cwnd = 8 KB, rwnd = 58 KB
Round 4 begin: cwnd = 16 KB, rwnd = 50 KB
Round 5 begin: cwnd = 32 KB, rwnd = 34 KB
Round 6 begin: cwnd = 64 KB, rwnd = 2 KB
```

TCP/IP provides a reliable transport layer, using acknowledgments to confirm data receipt. Both data and acknowledgments can be lost, so TCP sets a timer when data is sent. If the timer expires without receiving an acknowledgment, the data is retransmitted. This section discusses the timeout and retransmission strategies.

Basic Concepts in Timeout and Retransmission Strategies:

1. **Congestion Window (cwnd):** A key parameter in congestion control that limits the amount of data a sender can send without acknowledgment.
2. **Advertised Window (awnd):** The limit set by the receiver on the sender's window size, communicated during the connection establishment through ACKs.
3. **Slow Start Threshold (ssthresh):** A threshold used to limit the size of the sending window, distinguishing between the slow start and congestion avoidance phases. Its initial value is set to 65,535 bytes or the size of awnd.
4. **Round-Trip Time (RTT):** The time interval from when a packet is sent by the sender until an acknowledgment for that packet is received.
5. **Retransmission Timeout (RTO):** The interval from when a packet is sent until it is considered lost and retransmitted. This value helps determine packet loss and network congestion.

RTT and RTO Calculation:

RTT is calculated as a weighted average of the round-trip time samples of various segments:

[\text{RTT} = a \times (\text{old RTT}) + (1 - a) \times (\text{new round-trip time sample})]

where (0 \leq a < 1), typically (a = 0.9).

The retransmission timeout (RTO) is slightly greater than (RTT):

[\text{RTO} = b \times \text{RTT}]

where (b) is a value greater than 1 but difficult to determine precisely. The retransmission ambiguity problem arises when an ACK is received after a retransmission, making it unclear whether it acknowledges the original or the retransmitted segment. The Karn's Algorithm suggests not using the round-trip time sample from retransmitted packets to update RTT and RTO.

TCP Retransmission Strategy and Congestion Control:

1. **Slow Start Phase:** When the connection is established, TCP enters the slow start phase with an initial (cwnd) of one segment and (ssthresh) of 65,535 bytes. The actual sending window is the minimum of (cwnd) and (awnd). Each ACK doubles the (cwnd), causing exponential growth during this phase to avoid network congestion from sending too many packets at once. When (cwnd) exceeds (ssthresh), TCP switches to the congestion avoidance algorithm.
2. **Congestion Avoidance Phase:** If a timeout occurs or three duplicate ACKs are received, TCP assumes congestion and enters the congestion avoidance phase. Here, (ssthresh) is set to half the current (cwnd), with a minimum of two segments. If (cwnd < ssthresh), TCP reenters the slow start phase; otherwise, it performs congestion avoidance, increasing (cwnd) linearly with each ACK.

3. **Fast Retransmit Phase:** Upon receiving three duplicate ACKs, the sender retransmits the lost packet and sets (ssthresh) to half the current (cwnd) and (cwnd) to (ssthresh) plus three segments.
4. **Fast Recovery Phase:** Following fast retransmit, TCP avoids reducing (cwnd) to one segment by using the congestion avoidance algorithm, increasing (cwnd) by one segment for each duplicate ACK, and returning to the congestion avoidance phase upon receiving a non-duplicate ACK.