Network Simulation with Dynamic Timeout and Retransmission

High-Level Idea and Architecture

I chose the TCP project that simulates a reliable transport layer protocol over a network that delays, and drops packets, and sometimes delivers out-of-order. This system uses a simplified TCP implementation with the following features:

- Sliding Window Protocol: maintain a window of unacknowledged packets to optimize throughput and handle retransmissions.
- Timeout and Retransmission: resend packets if acknowledgments (ACKs) are not received within the estimated round-trip time (RTO).
- RTT Estimation: Implements RTT calculation with the RFC 6298 algorithm
- Dummy Packet Injection: simulates realistic network conditions by introducing dummy traffic into router buffers, which can cause delays and buffer overflows but these dummy packets don't get forwarded to the next router in the network.
- Packet Forwarding and ACK Handling: ensures reliable delivery by incrementally forwarding packets through routers and handling acknowledgment responses.

Changes I made:

- Dynamic Timeout and Retransmission: this project now handles retransmitting unacknowledged packets after a timeout using RTT-based RTO calculations.
- Resend logic ensures only packets that are still unacknowledged are retransmitted.
- Improved network simulation: injected dummy packets to simulate potential buffer overflows in routers.
- Improved handling of out-of-order delivery and buffer management.
- Adaptive RTT Calculation: Used the algorithm described in RFC 6298 for dynamic timeout adjustment based on observed RTT variance.
- Prevent long or short timeouts by fixing the RTO to predefined minimum and maximum values (specified in the RFC document).
- New ACK Handling: Improved acknowledgment validation to ensure that the sliding window advances correctly. Updated the base pointer to reflect the progression of successfully acknowledged packets.

Testing Process

I tested this new implementation by doing these:

- First I created a simple test program that sends a short message “Hello Minh Ann xinh dep!” to verify the basic functionality of this system works fine.
- Next, I checked that the packets were sent, forwarded through routers, and delivered to the receiver, the ACKs were received correctly, and the sliding window advanced as expected and the final message matched my original short message.
- My simple test showed me the issues with ACK handling and timeout logic, which were fixed after properly updating the base and using the correct sequence numbers for retransmission.
- Then I used the test.c file provided in the original homework to test the system with longer messages.
- This program tested the handling of larger datasets by sending multi-segment packets, including: splitting data into multiple packets using the sliding window protocol, managing timeouts and retransmissions for longer message sequences, receiving all packets, and reconstructing the original message.
- The longer test helps me confirm the proper operation of the sliding window mechanism, the correct retransmissions for unacknowledged packets within the timeout window, and the accurate assembly of the final message at the receiver, even under network stress when I set the drop packet probability to be high.