Assignment 2

By Raghav Sakhuja
2021274

Part B

Header File:stream_reassembler.hh

Private members:

- buffer: A **vector** of **packets** that stores received data along with their indices and lengths.
- acknowledged: The index up to which data has been successfully assembled.
- unassembled: The number of bytes inside the buffer to be assembled.
- capacity: The maximum capacity (in bytes) that the reassembler can hold.
- eof: A flag indicating whether the end of the byte stream has been reached.
- _output: An instance of the ByteStream class where the in-order byte stream is stored.

- Data Type
    - I have created a packet structure to store the incoming packets in the buffer in an ordered manner
    - The packet struct represents a data packet with an index, length, and the actual data.
    - It has an operator< function for sorting packets based on their indices.

Source File stream_reassembler.cc:

- The push_substring function receives a substring of data, its index in the sequence, and an end-of-file flag. It assembles contiguous bytes into the output stream in order, taking into account capacity limits and possible overlaps. It uses the overlap and merge functions to order the buffer.

- The overlap function checks whether two packets overlap in their data ranges.

- The merging function handles the merging of two packets when they overlap, ensuring that the data is correctly combined. It gives **preference to the newer data** to replace the older data in the buffer in case of overlaps.

- The unassembled_bytes function returns the number of bytes waiting to be assembled.

- The empty function checks if there are no substrings waiting to be assembled.

- The ack_index function returns the index of the next interested substring to be assembled.

Part c

Wrapping

- It results in a wrapped 32-bit sequence number by adding the 64-bit sequence number n to the 32-bit isn to create the wrapped sequence number.

UnWrapping

- It first calculates the difference (diff) between n and isn, considering wraparound.
- Then, it checks whether the checkpoint is closer to diff in the clockwise or counterclockwise direction (wrapping around the 64-bit space).
- It calculates two potential 64-bit sequence numbers (diff1 and diff2) by adding multiples of the 32-bit wraparound space (1ull<<32) to diff.
- It selects the closest one among diff1 and diff2 to the checkpoint and returns it as the absolute 64-bit sequence number.

TCP Reciever

- It extracts information from the segment, such as the sequence number (seqno), SYN flag (syn), payload data, and FIN flag.
- It handles SYN and FIN flags to establish and terminate connections.
- It computes the absolute sequence number (abs_seqno) relative to the initial sequence number (_isn) and the current acknowledgment index (checkpoint).
- It calls _reassembler.push_substring to reassemble and process the payload data, taking into account the FIN flag and the current state of the receiver.
- It updates the acknowledgment number (ack) based on the acknowledgment index (checkpoint2) after processing the segment.

Results

```
rag@rag-VirtualBox:~/Desktop/CN/CN/ComputerNetworks/Ass2/assignment2/build$ ctest
Test project /home/rag/Desktop/CN/CN/ComputerNetworks/Ass2/assignment2/build
      Start  1: wrapping_integers_cmp
 1/23 Test  #1: wrapping_integers_cmp ................   Passed    0.01 sec
      Start  2: wrapping_integers_unwrap
 2/23 Test  #2: wrapping_integers_unwrap .............   Passed    0.00 sec
      Start  3: wrapping_integers_wrap
 3/23 Test  #3: wrapping_integers_wrap ...............   Passed    0.00 sec
      Start  4: wrapping_integers_roundtrip
 4/23 Test  #4: wrapping_integers_roundtrip ..........   Passed    0.34 sec
      Start  5: byte_stream_construction
 5/23 Test  #5: byte_stream_construction .............   Passed    0.01 sec
      Start  6: byte_stream_one_write
 6/23 Test  #6: byte_stream_one_write ................   Passed    0.00 sec
      Start  7: byte_stream_two_writes
 7/23 Test  #7: byte_stream_two_writes ...............   Passed    0.00 sec
      Start  8: byte_stream_capacity
 8/23 Test  #8: byte_stream_capacity .................   Passed    1.11 sec
      Start  9: byte_stream_many_writes
 9/23 Test  #9: byte_stream_many_writes ..............   Passed    0.01 sec
      Start 10: recv_connect
10/23 Test #10: recv_connect .........................   Passed    0.01 sec
      Start 11: recv_transmit
11/23 Test #11: recv_transmit ........................   Passed    0.15 sec
      Start 12: recv_window
12/23 Test #12: recv_window ..........................   Passed    0.00 sec
      Start 13: recv_reorder
13/23 Test #13: recv_reorder .........................   Passed    0.01 sec
      Start 14: recv_close
14/23 Test #14: recv_close ...........................   Passed    0.00 sec
      Start 15: recv_special
15/23 Test #15: recv_special .........................   Passed    0.00 sec
      Start 16: fsm_stream_reassembler_cap
16/23 Test #16: fsm_stream_reassembler_cap ...........   Passed    0.26 sec
      Start 17: fsm_stream_reassembler_single
17/23 Test #17: fsm_stream_reassembler_single ........   Passed    0.01 sec
      Start 18: fsm_stream_reassembler_seq
18/23 Test #18: fsm_stream_reassembler_seq ...........   Passed    0.01 sec
      Start 19: fsm_stream_reassembler_dup
19/23 Test #19: fsm_stream_reassembler_dup ...........   Passed    0.01 sec
      Start 20: fsm_stream_reassembler_holes
20/23 Test #20: fsm_stream_reassembler_holes .........   Passed    0.01 sec
      Start 21: fsm_stream_reassembler_many
21/23 Test #21: fsm_stream_reassembler_many ..........   Passed    0.29 sec
      Start 22: fsm_stream_reassembler_overlapping
22/23 Test #22: fsm_stream_reassembler_overlapping ...   Passed    0.01 sec
      Start 23: fsm_stream_reassembler_win
23/23 Test #23: fsm_stream_reassembler_win ...........   Passed    0.28 sec

100% tests passed, 0 tests failed out of 23

Total Test time (real) =   2.56 sec
rag@rag-VirtualBox:~/Desktop/CN/CN/ComputerNetworks/Ass2/assignment2/build$
```