Assignment 2 Part II and III

Name: Pourav Surya

Roll No: 2021271

In stream_reassembler.cc it first checks if the starting index is equal to the acknowledgement number. If it is, it writes the data into the _output bytestream and adds to the acknowledgment number. Else if, index is less than ack number, it writes only the substring of the data to the _output and adds to the ack. Lastly, else it writes the remaining capacity of the _output and conditionally stores it in the re_buffer map, that the index is greater than the ack number.

```
if (index == ack) {
    size_t bytes_written = _output.write(data);
    ack += bytes_written;
}

else if (index < ack){
    size_t new_index = ack - index;
    size_t bytes_written = _output.write(data.substr(new_index));
    ack += bytes_written;
}

else {
    int i = 0;
    size_t remaining_capacity = _output.remaining_capacity()+ack;
    while (i < n && i < remaining_capacity) {
        re_buffer[i+index] = data[i];
        ++i;
    }
}
```

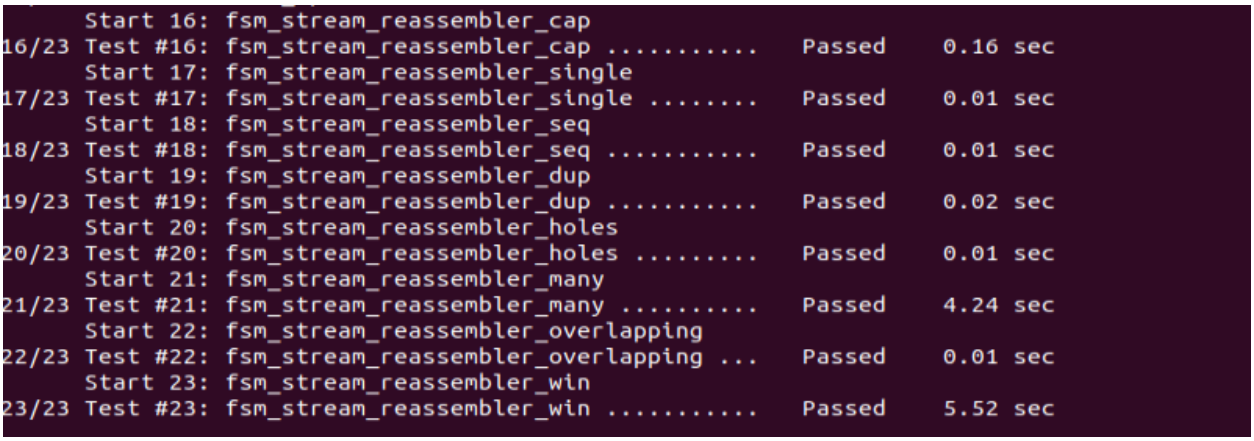
The below code is for iterating over the buffer and writing to the _output. Checks if the map index is greater than ack number then break; if they are equal, then push the value of the key to the _output or else erases it from the map. The code also handles the end of the file and accordingly checks the _output.

```
    auto i = re_buffer.begin();
    while(i!=re_buffer.end()){
        auto p = i->first;
        auto datum = i->second;

        if(p>ack){break;}
        else{
            if(p==ack){
                if(_output.remaining_capacity() != 0){
                string a = "";
                a.push_back(datum);
                _output.write(a);
                i=re_buffer.erase(i);
                ++ack;
                }
                else{
                    break;
                }
            }
            else{
                i = re_buffer.erase(i);
            }


        }
    }
```

All stream_reassembler test cases have passed.



For the tcp receiver flags for data arrival _synReceived is set. If payload is received, then is stored in d. d is unwraped from 32 bit to 64 bit and then pushed into the reassember. Flag fin is set to check if the reassembler is empty to close the output stream. In the lines wraps 64bit data using emplace into res variable and returns it.

```cpp
void TCPReceiver::segment_received(const TCPSegment &seg) {
    const TCPHeader &header = seg.header();
    bool syn = header.syn;
    bool fin = header.fin;

    //Checks seg using syn and synReceived
    if (!syn && !_synReceived)
        return;
    if (!_synReceived) {
        _isn = header.seqno;
        _synReceived = true;
    }
    string d = seg.payload().copy();
    if (!d.empty()) {
            size_t checkpoint = unwrap(header.seqno - (!syn), _isn, _reassembler.ack_index());
            _reassembler.push_substring(d, checkpoint, fin);
        // }
    }
    // setting fin flag and then check if the reassembler is empty to close the output stream
    if (fin || _finReceived) {
        _finReceived = true;
        if (_reassembler.unassembled_bytes() == 0)
            _reassembler.stream_out().end_input();
    }
}

optional<WrappingInt32> TCPReceiver::ackno() const {
    optional<WrappingInt32> res = nullopt;
    if (_synReceived) {
        //increment the checkpoint if the reassembler is empty
        uint64_t checkpoint = _reassembler.ack_index() + 1;
        if (_reassembler.stream_out().input_ended())
            checkpoint++;
        res.emplace(wrap(checkpoint,_isn));
    }
    return res;
```

All 23 test cases have passed without any errors.

```
Test project /home/suryabhai/Downloads/assignment3/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    0.01 sec
      Start  3: wrapping_integers_wrap
 3/23 Test  #3: wrapping_integers_wrap ..............   Passed    0.01 sec
      Start  4: wrapping_integers_roundtrip
 4/23 Test  #4: wrapping_integers_roundtrip .........   Passed    0.50 sec
      Start  5: byte_stream_construction
 5/23 Test  #5: byte_stream_construction ............   Passed    0.02 sec
      Start  6: byte_stream_one_write
 6/23 Test  #6: byte_stream_one_write ...............   Passed    0.01 sec
      Start  7: byte_stream_two_writes
 7/23 Test  #7: byte_stream_two_writes ..............   Passed    0.01 sec
      Start  8: byte_stream_capacity
 8/23 Test  #8: byte_stream_capacity ................   Passed    2.57 sec
      Start  9: byte_stream_many_writes
 9/23 Test  #9: byte_stream_many_writes .............   Passed    0.00 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    0.11 sec
      Start 12: recv_window
12/23 Test #12: recv_window .........................   Passed    0.01 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    0.01 sec
      Start 15: recv_special
15/23 Test #15: recv_special ........................   Passed    0.01 sec
      Start 16: fsm_stream_reassembler_cap
16/23 Test #16: fsm_stream_reassembler_cap ...........   Passed    0.16 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    0.02 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    4.24 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    5.52 sec

100% tests passed, 0 tests failed out of 23

Total Test time (real) =  13.33 sec
suryabhai@suryabhai-VirtualBox:~/Downloads/assignment3/assignment2/build$
```