Computer Network Lab3

107081028 常安彥

1. How to send correctly?

   Store loss packets in an array, and if the loss packets have not been resent, send them in the sequence order. Otherwise, the next sent sequence is the number of the latest sent sequence ever plus one.

```
if (cur_loss == loss)
    data_seq->seq_num = cur_final;
else
    data_seq->seq_num = losses[cur_loss++];
if (send(client_sock, (void*)data_seq, sizeof(*data_seq), 0) < 0){
    printf("Can't send\n");
    return;
}
printf("send: seq_num = %d\n", data_seq->seq_num);
data_seq->seq_num = data_seq->seq_num + 1;
cur_final = (data_seq->seq_num > cur_final) ? data_seq->seq_num : cur_final;
```

2. How to simulate packet loss?

   I scheduled some loss in the client code, and print failures with same ACKs sent when encountering these scheduled sequence numbers. Loss will only occur on the same number once in my implementation.

```
if ((data_seq.seq_num == 8 && cur_final == 8) || (data_seq.seq_num == 13 && cur_final == 13) ||
{
    printf("loss: seq_num = %d\n", data_seq.seq_num);
    losses[loss] = data_seq.seq_num;
    loss += 1;
    ACK.seq_num = ACK.seq_num;
}
```

3. How to detect 3 duplicate ACKs?

   Simply count the incoming ACK duplication. Store the ACK number and check if it is the same as the previous one.

```
if (duplicate_start == ACK->seq_num)
    count += 1;
if(count == 3)
{
    set_one = true;
    printf("3-duplicate ACKs: seq_num = %d, cwnd = %d, ssthresh = %d\n", duplicate_
}
```

4. How to update?

   If encountering packet loss, set ssthresh to half of the window size. If encountering three duplication, set window size to 1, else increase window size by doubling if window size is smaller than thresh or by 1 otherwise.

```
if (cut_half)
{
    ssthresh = cwnd / 2;
    cut_half = false;
}
if (set_one)
{
    cwnd = 1;
    set_one = false;
    printf("state: slow start\n");
    printf("cwnd = %d, ssthresh = %d\n", cwnd, ssthresh);
}
else if (cwnd < ssthresh)
{
    cwnd *= 2;
    printf("cwnd = %d, ssthresh = %d\n", cwnd, ssthresh);
}
else
{
    cwnd += 1;
    printf("state: congestion avoidance\n");
    printf("cwnd = %d, ssthresh = %d\n", cwnd, ssthresh);
}
```

5. How does client receive and send correctly?

   If packet loss occurs, duplicate the ACK. Otherwise, send ACK numbers in order of from the loss array. If none is left in the array, simply request the number of the latest received sequence ever plus one.

```
cur_final = (data_seq.seq_num > cur_final) ? data_seq.seq_num : cur_final;
if ((data_seq.seq_num == 8 && cur_final == 8) || (data_seq.seq_num == 13 && c
{
    printf("loss: seq_num = %d\n", data_seq.seq_num);
    losses[loss] = data_seq.seq_num;
    loss += 1;
    ACK.seq_num = ACK.seq_num;
}
else if (data_seq.seq_num == ACK.seq_num && cur_loss != loss)
{
    printf("received: seq_num = %d\n", data_seq.seq_num);
    if (cur_loss == loss - 1)
    {
        ACK.seq_num = cur_final + 1;
        cur_loss++;
    }
    else
        ACK.seq_num = losses[++cur_loss];
}
else if (cur_loss == loss)
{
    ACK.seq_num = data_seq.seq_num + 1;
    printf("received: seq_num = %d\n", data_seq.seq_num);
}
else
    printf("received: seq_num = %d\n", data_seq.seq_num);
```

Screenshot

```
Connected successfully
received: seq_num = 0
received: seq_num = 1
received: seq_num = 2
received: seq_num = 3
received: seq_num = 4
received: seq_num = 5
received: seq_num = 6
received: seq_num = 7
loss: seq_num = 8
received: seq_num = 9
received: seq_num = 10
received: seq_num = 11
received: seq_num = 12
loss: seq_num = 13
received: seq_num = 14
received: seq_num = 8
received: seq_num = 13
received: seq_num = 15
received: seq_num = 16
received: seq_num = 17
received: seq_num = 18
```

```
ACK: seq_num = 1
cwnd = 2, ssthresh = 8
send: seq_num = 1
send: seq_num = 2
ACK: seq_num = 2
ACK: seq_num = 3
cwnd = 4, ssthresh = 8
send: seq_num = 3
send: seq_num = 4
send: seq_num = 5
send: seq_num = 6
ACK: seq_num = 4
ACK: seq_num = 5
ACK: seq_num = 6
ACK: seq_num = 7
cwnd = 8, ssthresh = 8
send: seq_num = 7
send: seq_num = 8
send: seq_num = 9
send: seq_num = 10
send: seq_num = 11
send: seq_num = 12
send: seq_num = 13
send: seq_num = 14
ACK: seq_num = 8
ACK: seq_num = 8
ACK: seq_num = 8
3-duplicate ACKs: seq_num = 8, cwnd = 8, ssthresh = 8
ACK: seq_num = 8
ACK: seq_num = 8
ACK: seq_num = 8
ACK: seq_num = 8
ACK: seq_num = 8
state: slow start
cwnd = 1, ssthresh = 4
send: seq_num = 8
ACK: seq_num = 13
cwnd = 2, ssthresh = 4
send: seq_num = 13
send: seq_num = 15
ACK: seq_num = 15
ACK: seq_num = 16
cwnd = 4, ssthresh = 4
send: seq_num = 16
send: seq_num = 17
send: seq_num = 18
send: seq_num = 19
ACK: seq_num = 17
ACK: seq_num = 18
ACK: seq_num = 19
ACK: seq_num = 20
state: congestion avoidance
cwnd = 5, ssthresh = 4
```

In above screenshot, we can see that the window size increase by double in the slow state, and increases by 1 in the congestion avoidance state. Also, the client sent correct ACK in response to packet loss, and the reaction to 3 duplication is propriate.