Davis 109006201 周志偉
LAB 3 Report

How to send a packet with the correct sequence number?

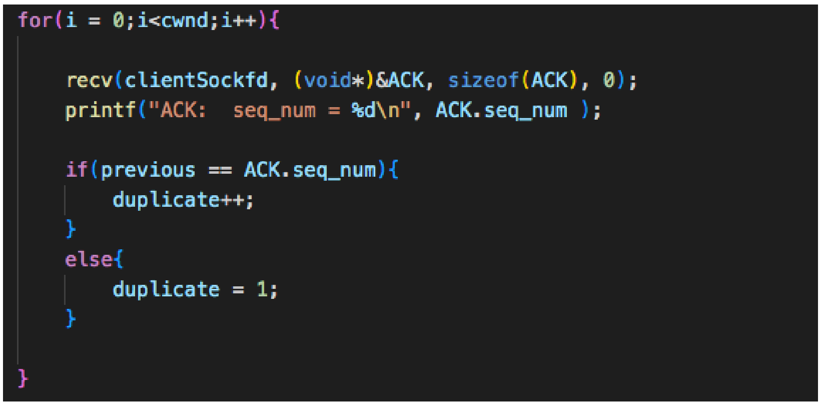
The server and client will send the packet in the correct order when they are connected successfully. When the server sends a packet the client will receive and after receiving the packet will send the back to the server. In the scenario packets are lost in the way, we have the 3-duplicate ACK detection to know if there was any packet loss. When this happens we will resend the packet again. But what if more than one packet is lost at the same loop, we will return the ACk after checking and finding out that the lost packet is not equal to the current sequence number. In my code for the server I have two if statements for when you resent the packet is there is a loss and a normal one that sends.

How to simulate packet loss?

There shouldn’t be a loss in our code, so in this case we can set the packet loss to any number. I simply set, data_seq.seq_num == 8 to simulate packet loss. If you want more loss you can just generate a random number and get the remainder.

How to detect 3 duplicate ACKs?

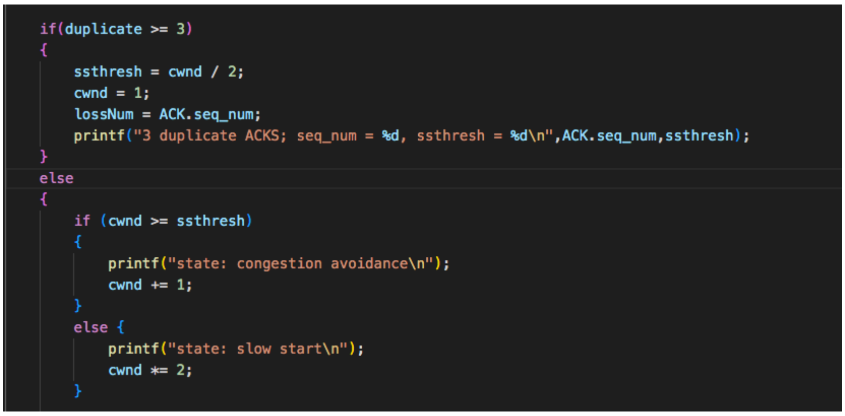
On the server side, everytime an ACK is received, it is compared to the previous ACK received. You do nothing when the previous ACK is different from the new one. If both ACK are the same our Duplicate will be increased by 1. If duplicate is 3, we  printf 3 duplicate ACKS, with sequence number and ssthresh.



```
for(i = 0;i<cwnd;i++){

    recv(clientSockfd, (void*)&ACK, sizeof(ACK), 0);
    printf("ACK:  seq_num = %d\n", ACK.seq_num );

    if(previous == ACK.seq_num){
        duplicate++;
    }
    else{
        duplicate = 1;
    }

}
```

How to update cwnd and ssthresh?

When we send a package on the server side the amount of time we send depends on our cwnd.  After sending, the receiver will accept ACKs. The cwnd and ssthresh will be updated in the receiver depending on the situation. If require fast retransmit (three duplicate ack ), and depending on phase if cwnd is less than or greater than ssthresh). if (cwnd >= ssthresh) the state is congestion avoidance will increase by 1, else it's a slow start and we double it. When three duplicate ACKs will set the cwnd to half and cwnd to 1.

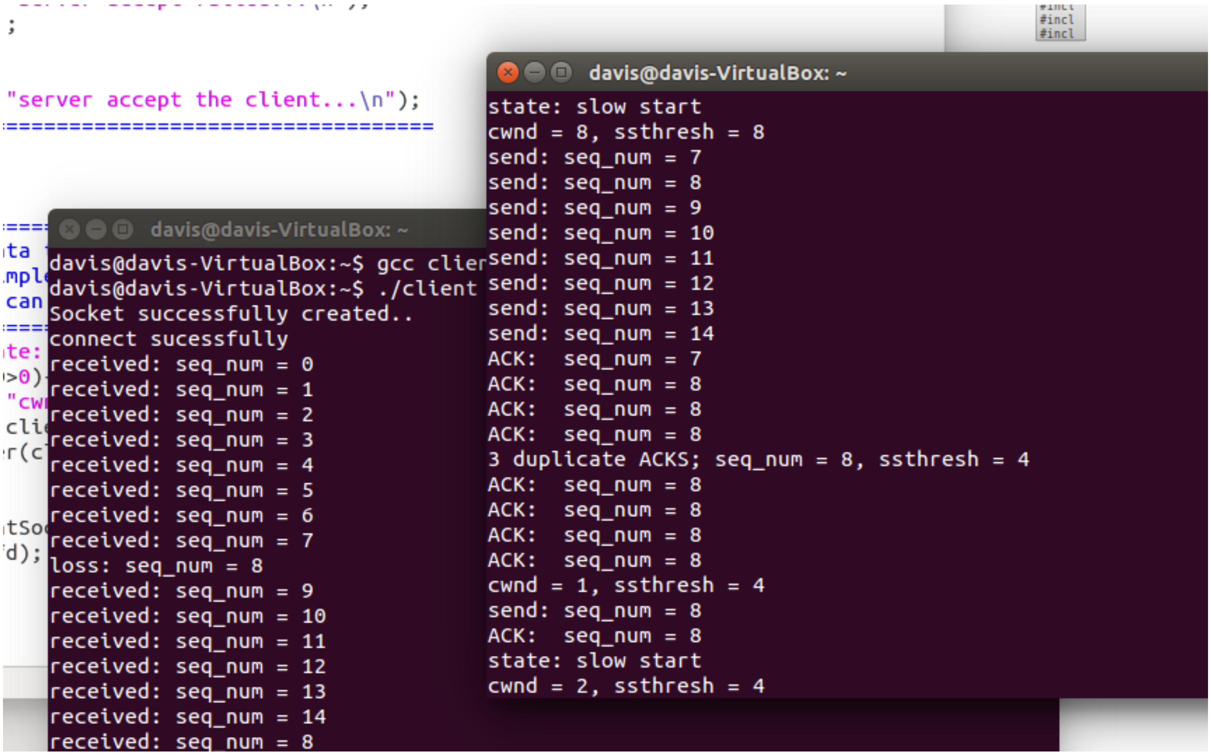


```
if(duplicate >= 3)
{
    ssthresh = cwnd / 2;
    cwnd = 1;
    lossNum = ACK.seq_num;
    printf("3 duplicate ACKS; seq_num = %d, ssthresh = %d\n",ACK.seq_num,ssthresh);
}
else
{
    if (cwnd >= ssthresh)
    {
        printf("state: congestion avoidance\n");
        cwnd += 1;
    }
    else {
        printf("state: slow start\n");
        cwnd *= 2;
    }
```

How do clients receive data and send ACKs with the correct sequence number?

When the sequence number that it is supposed to receive is correct they send it back, the client will check if the sequence number is one more than the previous one. If it is not equal to the one it expected it will keep sending ACK of the lost packet back. In my client I will have an array that stores the count loss and the unACK packet. So when the loss packet is resend it will ACK the correct packet and continue to ACK the correct Packet sequence number that is sent.

Output



```
"server accept the client...\n");
```

```
davis@davis-VirtualBox:~$ gcc clier
davis@davis-VirtualBox:~$ ./client
Socket successfully created..
connect sucessfully
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
received: seq_num = 13
received: seq_num = 14
received: seq_num = 8
```

```
state: slow start
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 = 7
ACK:  seq_num = 8
ACK:  seq_num = 8
ACK:  seq_num = 8
3 duplicate ACKS; seq_num = 8, ssthresh = 4
ACK:  seq_num = 8
ACK:  seq_num = 8
ACK:  seq_num = 8
ACK:  seq_num = 8
cwnd = 1, ssthresh = 4
send: seq_num = 8
ACK:  seq_num = 8
state: slow start
cwnd = 2, ssthresh = 4
```