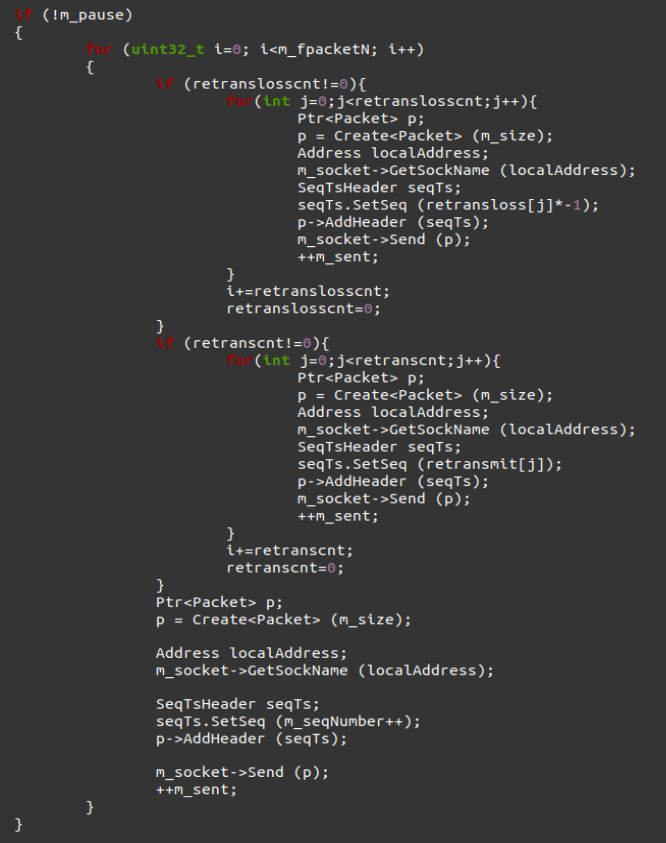
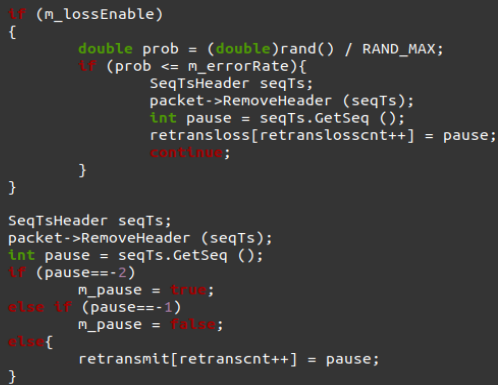
Final Project report Group 1 안현우, 김태은, 여혁수

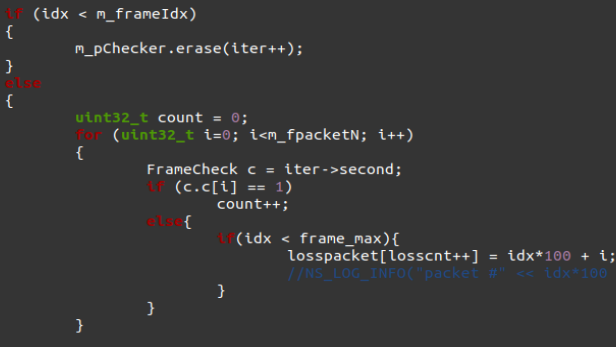
Detail code explanation which you implemented.



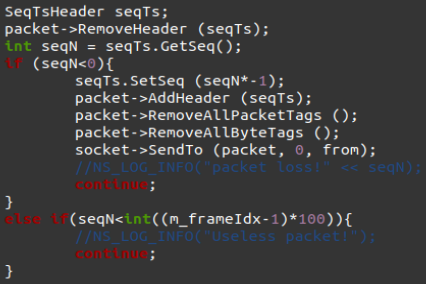
Streamer.cc 의 SendPacket 함수 코드 입니다. 함수가 한 번 호출될 때마다 패킷 100개를 보내도록 속도가 정해져 있기 때문에 for문을 100번 돌면서 패킷을 하나씩 client로 보냅니다. retransloss라는 배열이 있는데, 1차적으로 retransmit이 필요하다는 것을 알려주는 패킷이 또 loss로 처리되어 2차, 3차 그 이상의 loss인 패킷의 sequence number들이 들어 있습니다. 이 경우의 패킷들은 빠르게 처리하기 위해 시퀀스 넘버에 -1을 곱해 음수로 만들어주고 보냈습니다. 그 패킷들을 가장 먼저 client로 보내주고, 다음으로 1차 retransmit이 필요한 패킷들이 retransmit이라는 배열 안에 들어있는데 배열 안에 패킷이 존재하면 client로 보내줍니다. 마지막으로 원래 보내야하는 프레임을 이루는 패킷들을 보내줍니다.



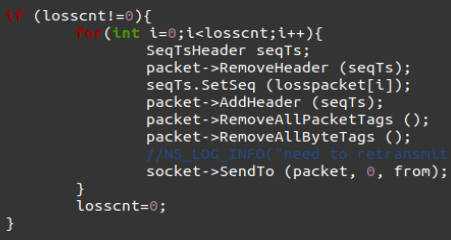
Streamer의 handleread에서 retransmit이 필요하다는 것을 받아 그 패킷의 시퀀스 넘버를 배열 안에 넣어주는 코드입니다.



Client.cc에서 FrameGenerator의 일부입니다. 현재 consume을 진행 중인 프레임 인덱스보다 작은 인덱스를 가진 프레임이 버퍼에 존재한다면 erase()를 통해 버퍼에서 지워줍니다. 또한 이 때까지 만들었던 프레임 중 인덱스가 가장 큰 프레임보다 작은 인덱스의 프레임 중 일부 패킷이 버퍼에 들어오지 않았다면 loss로 판단하여 losspacket이라는 배열에 넣어 retransmit을 요청할 수 있게 합니다.

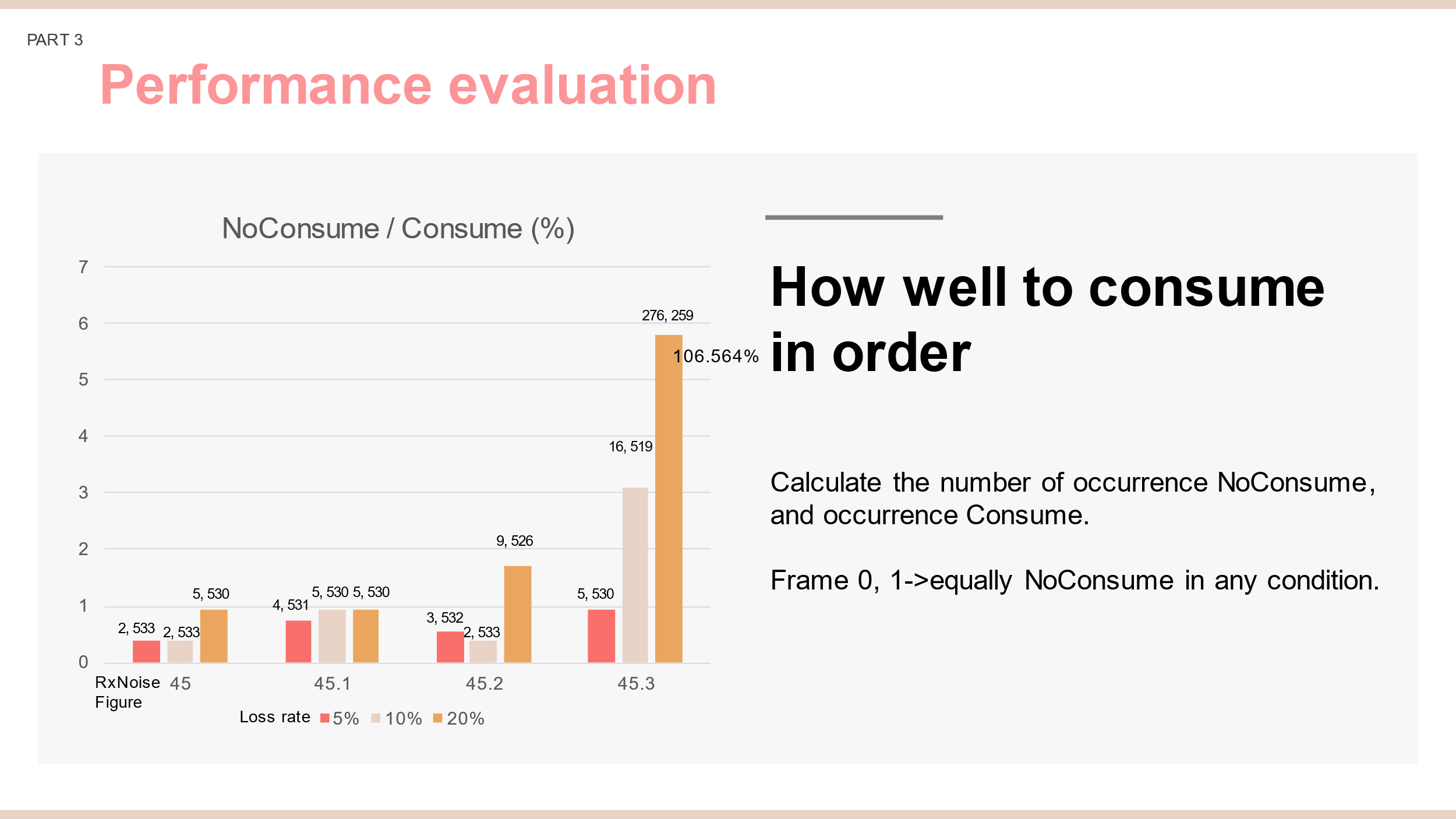


Client의 Handleread에서 패킷을 받았는데 시퀀스 넘버가 음수라면 그것은 streamer에서 2차적 loss 이상의 loss인 패킷으로 처리가 됩니다. 또한 retransmit 패킷 같은 경우로 인해 현재 소비하고 있는 프레임 이전에 패킷이 늦게 retransmit되는 경우가 있는데 이 때는 별다른 처리 없이 skip합니다.

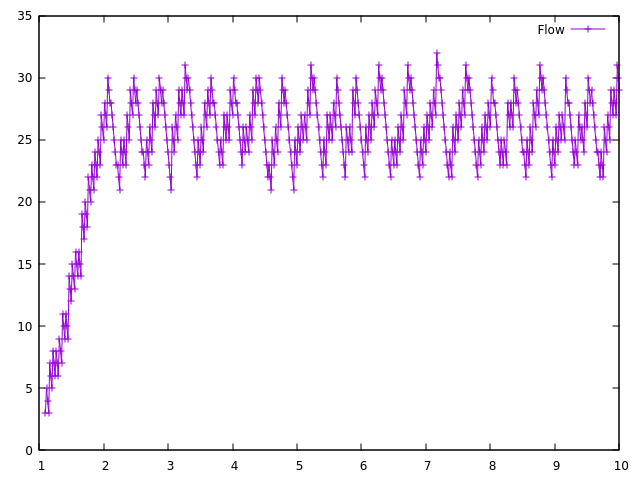


Client의 Handleread에서 로스 패킷의 retransmit 요청 패킷을 전송하는 코드입니다. losspacket이라는 배열에 패킷이 쌓여있다면 하나씩 전송을 하게 됩니다.

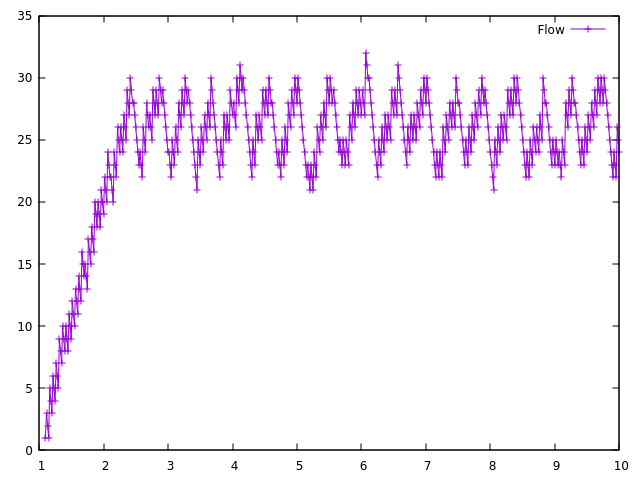
Performance evaluation.



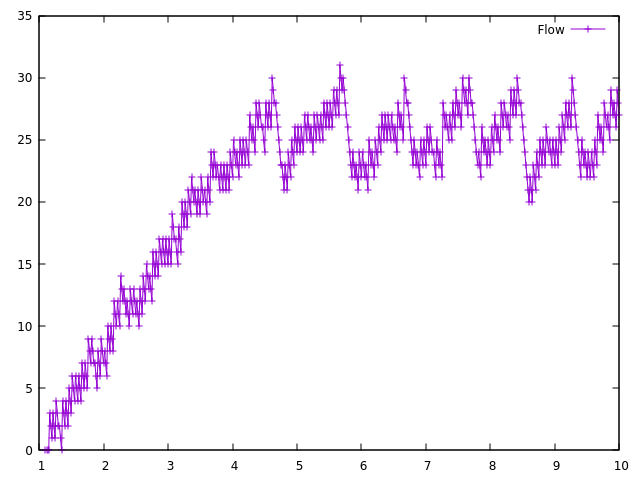
얼마나 순서대로 프레임 소비를 잘 하는지를 평가하기 위해서 전체 communication 과정 중 consume과 noconsume의 횟수를 세서 noconsume의 비율을 계산했습니다. RxNoiseFigure 값이 45.0일 때는 loss rate가 20%까지 올라가도 첫 2개의 프레임을 제외하고는 거의 noconsume 없이 프레임을 소비하는 것을 볼 수 있었습니다. RxNoiseFigure가 45.1일 때는 2,3개 정도 더 noconsume이 일어났지만 거의 성능 차이가 없었습니다. RxNoiseFigure가 45.2이상부터 성능이 눈에 띄게 저하되는 것을 볼 수 있는데, 45.2일 때 loss rate이 10%일 때까지는 원활하게 consume을 하지만 loss rate 20%에서 9개의 noconsume이 일어나는 것을 볼 수 있습니다. 45.3에서는 loss rate이 5%일 때부터 loss rate이 커질수록 급격하게 consume이 일어나지 않고 프레임이 만들어지지 않는 상황을 볼 수 있습니다. 20%의 loss rate일 때는 noconsume이 더 많이 일어나 consume에 대한 noconsume의 비율이 106%에 달하는 것을 볼 수 있습니다.



또한 얼마나 버퍼를 많이 가지고 유지를 하는가에 대한 평가는 시간에 따른 버퍼 사이즈를 gnuplot으로 그래프를 그래서 진행해보았습니다. RxNoiseFigure는 45.0으로 고정하고 버퍼 사이즈가 25이하일 때 resume 패킷을 전송하도록 고정했습니다. loss rate만 다르게 주어서 여러 그래프를 비교했습니다. 위 그래프는 loss rate이 5%일 때입니다. 시작한지 1초 정도 될 때 버퍼 사이즈가 30이 되고, 1초에 3번 정도 pause와 resume을 반복합니다. 버퍼 사이즈는 계속해서 잘 채워지고 있는 것을 볼 수 있고, 버퍼 사이즈가 21이하로는 떨어지지 않습니다.



위 그래프는 loss rate이 10%일 때입니다. 5%일 때보다 조금 늦게 1.5초 정도 지나서 버퍼 사이즈가 30이 됩니다. 마찬가지로 1초에 거의 3번 정도 pause와 resume을 반복하며 버퍼 사이즈가 잘 유지되는 것을 볼 수 있습니다. 버퍼 사이즈는 21이하로 떨어지지 않았습니다.



위 그래프는 loss rate이 20%일 때입니다. 앞선 두 그래프와는 다르게 버퍼가 매우 느리게 채워지는 것을 볼 수 있습니다. 거의 4초가 지나 버퍼 사이즈가 처음으로 30이 되고, 한번 pause와 resume을 반복했을 때 다시 사이즈를 30까지 채우는 데 1초 정도가 걸리는 것을 볼 수 있습니다.