TCP

네트워크프로그래밍

휴먼지능정보공학과
201810776 소재휘

Lab activities

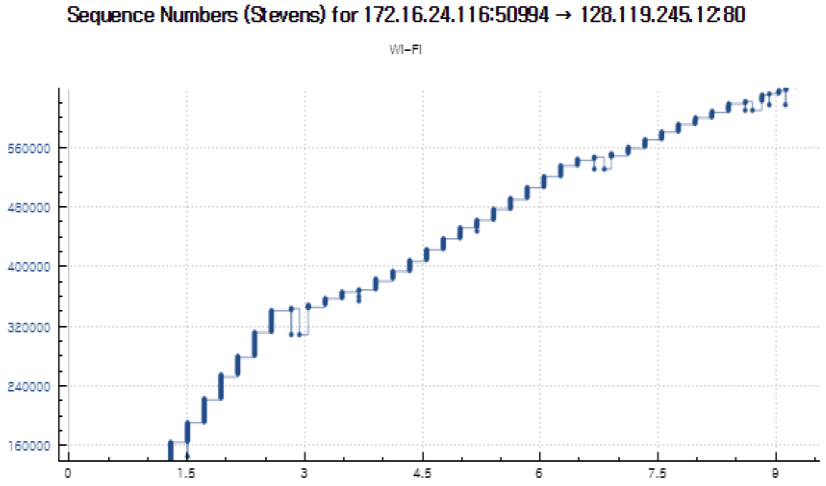
- Wireshark_TCP_v6.0
  - Section 4. TCP Congestion Control in action
  - Try to upload files with various sizes
  - Select a TCP segment in the Wireshark's "listing of captured-packets" window. Then select the menu : *Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens*).
  - Problems 13~14
    - Use the *Time-Sequence-Graph(Stevens*) plotting tool to view the sequence number versus time plot of segments being sent from the client to the gaia.cs.umass.edu server.
      - Can you identify where TCP's slowstart phase begins and ends, and where congestion avoidance takes over? Comment on ways in which the measured data differs from the idealized behavior of TCP that we've studied in the text.
    - Answer the questions above for the trace that you have gathered when you transferred a file from your computer to gaia.cs.umass.edu

#1 TCP

Network programming

tcp && ip.addr==172.16.24.116 && ip.addr==128.119.245.12

No.	Time	Source	Destination	Protocol	Length	Info
91	0.880626	172.16.24.116	128.119.245.12	TCP	66	50994 → 80 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
122	1.092736	128.119.245.12	172.16.24.116	TCP	66	80 → 50994 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 W…
123	1.092840	172.16.24.116	128.119.245.12	TCP	54	50994 → 80 [ACK] Seq=1 Ack=1 Win=262144 Len=0
124	1.093013	172.16.24.116	128.119.245.12	TCP	561	50994 → 80 [PSH, ACK] Seq=1 Ack=1 Win=262144 Len=507 [TCP segment of a r…
125	1.093560	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=508 Ack=1 Win=262144 Len=1460 [TCP segment of a rea…
126	1.093560	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=1968 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
127	1.093561	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=3428 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
128	1.093563	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=4888 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
129	1.093563	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=6348 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
130	1.093563	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=7808 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
131	1.093564	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=9268 Ack=1 Win=262144 Len=1460 [TCP segment of a re…
132	1.093564	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=10728 Ack=1 Win=262144 Len=1460 [TCP segment of a r…
133	1.093564	172.16.24.116	128.119.245.12	TCP	1514	50994 → 80 [ACK] Seq=12188 Ack=1 Win=262144 Len=1460 [TCP segment of a r…



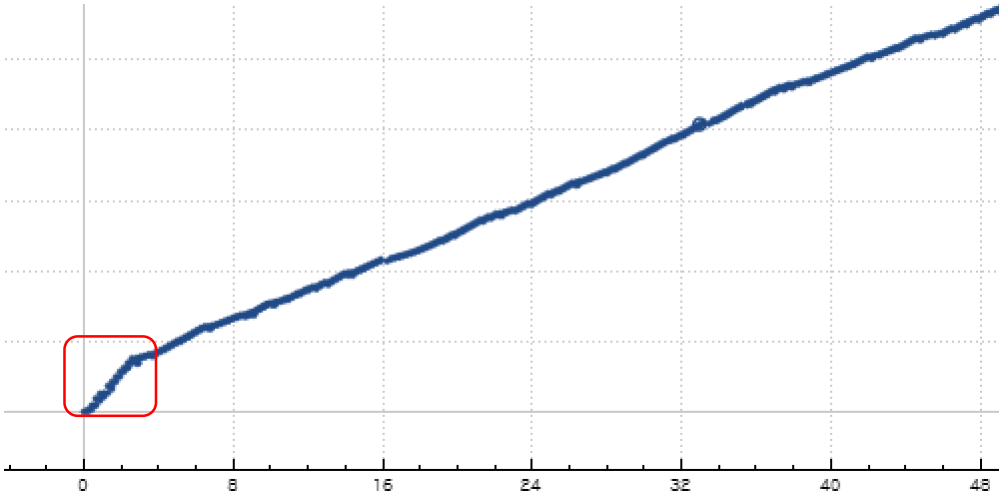
Section 4. TCP Congestion Control in action

Try to upload files with various sizes

Select a TCP segment in the Wireshark's "listing of captured-packets" window. Then select the menu : *Statistics->TCP Stream Graph-> Time-Sequence-Graph(Stevens)*.

#1 TCP

Network programming



우선 TCP에서 *Time-Sequence-Graph(Stevens)*를 관측한 결과이다. 다음의 빨간 네모 상자 부분을 보면 전송 byte에 해당하는 그래프가 가파르게 증가하는 것을 확인할 수 있었다. 이 부분에서 slow start를 관측할 수 있었다.
Slow Start : congestion window 사이즈를 1부터 시작해서 exponetial하게 증가해 나가면서 적절한 window size를 찾는 과정
다음만 놓고 봐서는 계속해서 증가하는 곡선을 보이므로 Congestion에 대한 관측은 불가능하다.

#1 TCP

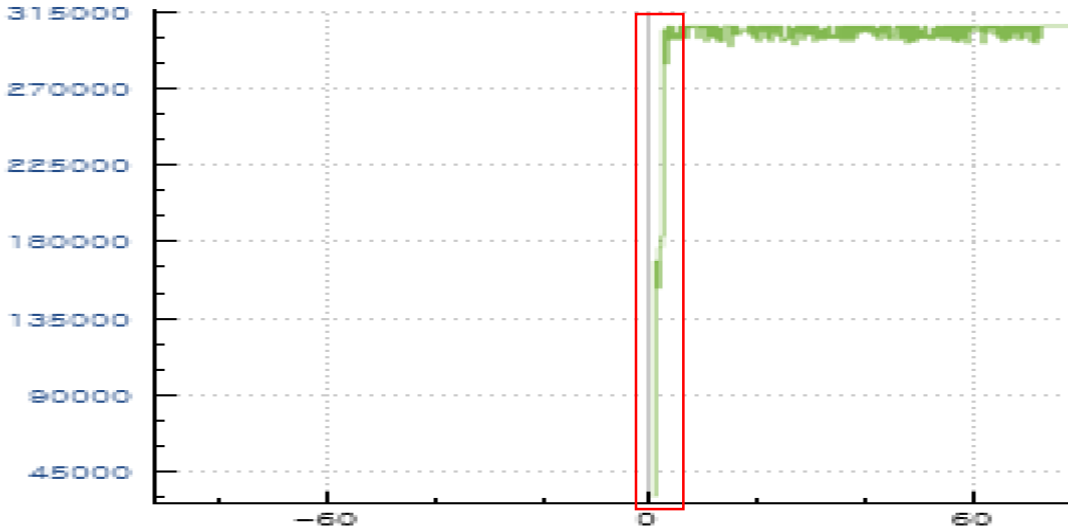
Network programming

555 3.710359	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 502#2] 80 → 50994 [ACK] Seq=1 Ack=308568 Win=302592 Len=0 S.
556 3.710360	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 502#3] 80 → 50994 [ACK] Seq=1 Ack=308568 Win=302592 Len=0 S.
557 3.710433	172.16.24.116	128.119.245.12	TCP	1514 [TCP Fast Retransmission] 50994 → 80 [ACK] Seq=308568 Ack=1 Win=262144 L.
558 3.710481	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=342148 Ack=1 Win=262144 Len=1460 [TCP segment of a .
559 3.710514	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=343608 Ack=1 Win=262144 Len=1460 [TCP segment of a .
574 3.814034	172.16.24.116	128.119.245.12	TCP	1514 [TCP Out-Of-Order] 50994 → 80 [ACK] Seq=308568 Ack=1 Win=262144 Len=1460.
585 3.927192	128.119.245.12	172.16.24.116	TCP	60 80 → 50994 [ACK] Seq=1 Ack=342148 Win=286592 Len=0
586 3.927193	128.119.245.12	172.16.24.116	TCP	60 80 → 50994 [ACK] Seq=1 Ack=345068 Win=284544 Len=0
587 3.927284	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=345068 Ack=1 Win=262144 Len=1460 [TCP segment of a .
588 3.927284	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=346528 Ack=1 Win=262144 Len=1460 [TCP segment of a .
589 3.927410	172.16.24.116	128.119.245.12	TCP	1514 [TCP Out-Of-Order] 50994 → 80 [ACK] Seq=345068 Ack=1 Win=262144 Len=1460.

다음은 Slow start가 끝나고 additive하게 증가하기 시작하는 시점의 segment들에서 관측할 수 있었던 결과이다. Congestion avoidance는 slow start 후 Window size가 너무 커져서 혼잡 감지 후 congestion control이 이루어지는 것이다. Slow start의 증가곡선 후 segment에서 congestion avoidance로 추측되는 근거들을 관측할 수 있다. 따라서 앞에서 그래프에서 관측했던 것이 Slow start라는 것의 근거가 된다.

#1 TCP

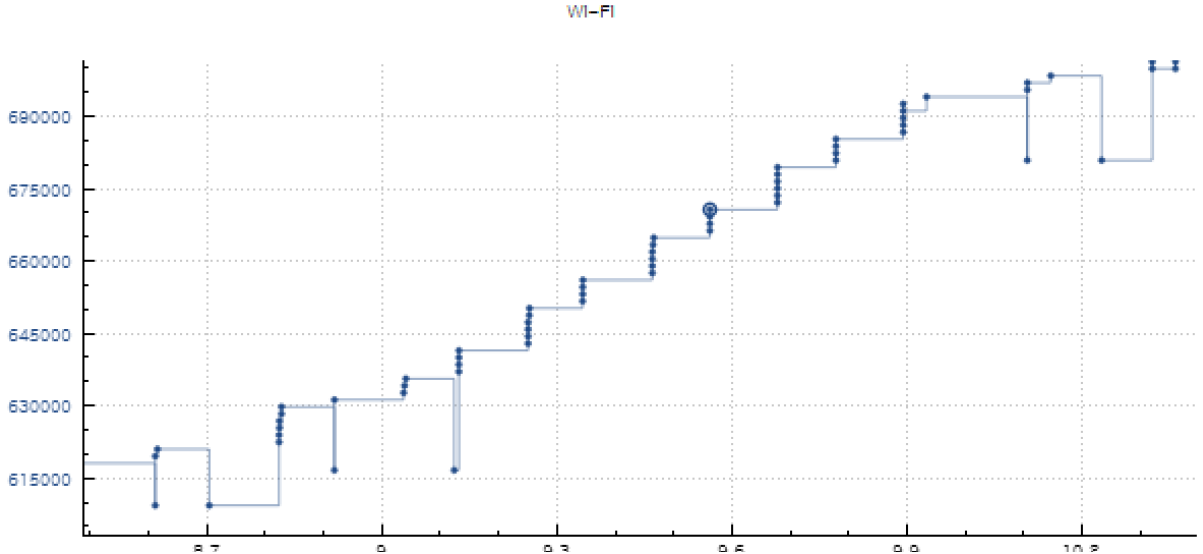
Network programming



Window scaling 과정에서도 보면 시작할 때 Window size가 가파르게 증가하는 것을 확인할 수 있다. 그 후 Window size를 줄이는 congestion avoidance 과정이 따른다. 따라서 slow start를 관측할 수 있었음에 설득력을 실어준다.

#1 TCP

Network programming



- 그래프를 확대해 보면 Sequence number가 급격하게 감소하였다가 증가하는 구간이 보인다. 이 부분에서 TCP segment의 Retransmission이 이루어 졌다는 것을 확인할 수 있다.

#1 TCP

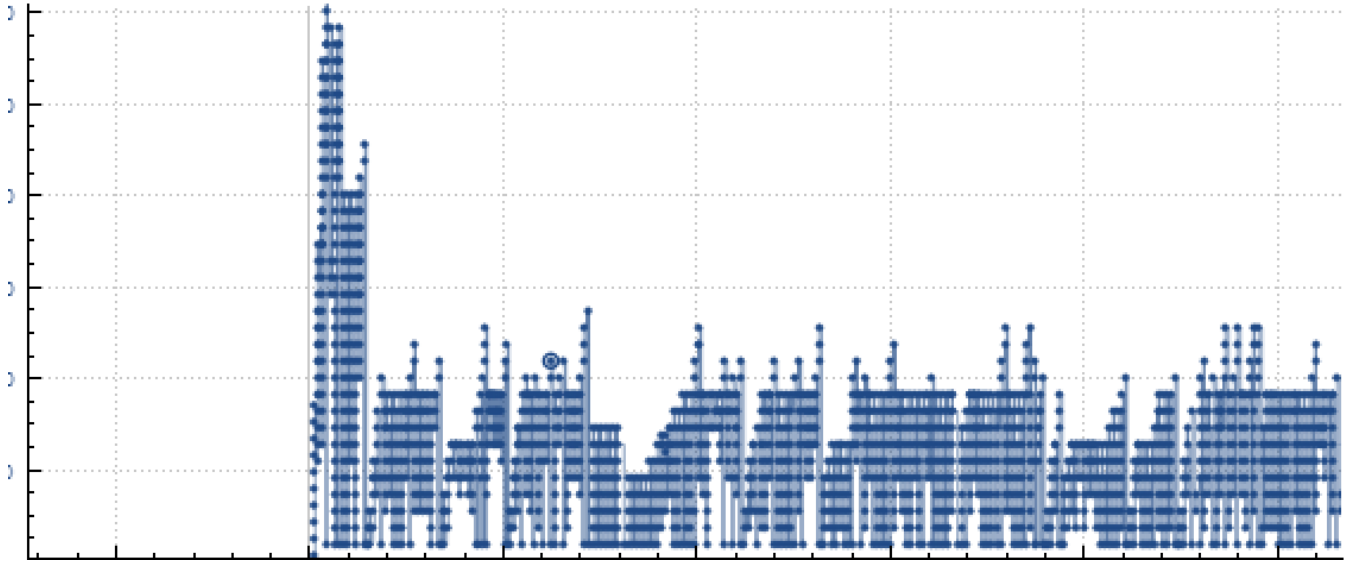
Network programming

5966	59.073529	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 5960#1] 80 → 50994 [ACK] Seq=1 Ack=3038115 Win=306816 Len=0…
5967	59.073614	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=3052715 Ack=1 Win=262144 Len=1460 [TCP segment of a…
5975	59.285373	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 5960#2] 80 → 50994 [ACK] Seq=1 Ack=3038115 Win=306816 Len=0…
5976	59.285429	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=3054175 Ack=1 Win=262144 Len=1460 [TCP segment of a…
5977	59.285645	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 5960#3] 80 → 50994 [ACK] Seq=1 Ack=3038115 Win=306816 Len=0…
5978	59.285696	172.16.24.116	128.119.245.12	TCP	1514 [TCP Fast Retransmission] 50994 → 80 [ACK] Seq=3038115 Ack=1 Win=262144 …

- Sequence number의 급격한 감소를 보이는 부분에서 확인할 수 있는 일부 segment이다. 다음과 같이 Duplicate ACK를 보내고 Retransmission을 하는 것을 확인 할 수 있었다. 이 과정에서 TCP의 Congestion control을 확인할 수 있었다.

#1 TCP

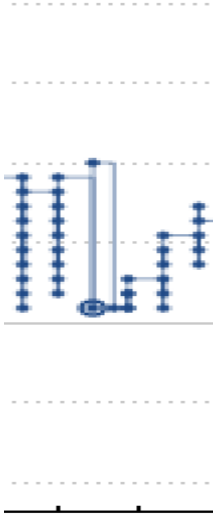
Network programming



Window scaling 과정을 보면 보내는 데이터 byte가 증가하다가 0에 근접한 값으로 감소하는 것을 볼 수 있다. 이로 보았을 때 Congestion control의 방식이 TCP tahoe 방식을 주로 이용하였을 것이라 추측하였다.

#1 TCP

Network programming

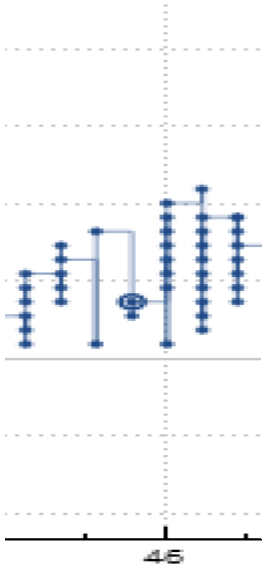


4740	45.597187	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 4714#1] 80 → 50994 [ACK] Seq=1 Ack=2376735 Win=300544 Len=0…
4741	45.597253	172.16.24.116	128.119.245.12	TCP	1514 [TCP Retransmission] 50994 → 80 [ACK] Seq=2376735 Ack=1 Win=262144 Len=1…
4742	45.597319	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2391335 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4758	45.725944	172.16.24.116	128.119.245.12	TCP	1514 [TCP Out-Of-Order] 50994 → 80 [ACK] Seq=2376735 Ack=1 Win=262144 Len=146…
4772	45.811595	128.119.245.12	172.16.24.116	TCP	60 80 → 50994 [ACK] Seq=1 Ack=2391335 Win=299520 Len=0
4773	45.811595	128.119.245.12	172.16.24.116	TCP	60 80 → 50994 [ACK] Seq=1 Ack=2392795 Win=298624 Len=0
4774	45.811659	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2392795 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4775	45.811659	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2394255 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4776	45.811733	172.16.24.116	128.119.245.12	TCP	1514 [TCP Out-Of-Order] 50994 → 80 [ACK] Seq=2392795 Ack=1 Win=262144 Len=146…
4777	45.811736	172.16.24.116	128.119.245.12	TCP	1514 [TCP Retransmission] 50994 → 80 [ACK] Seq=2394255 Ack=1 Win=262144 Len=1…

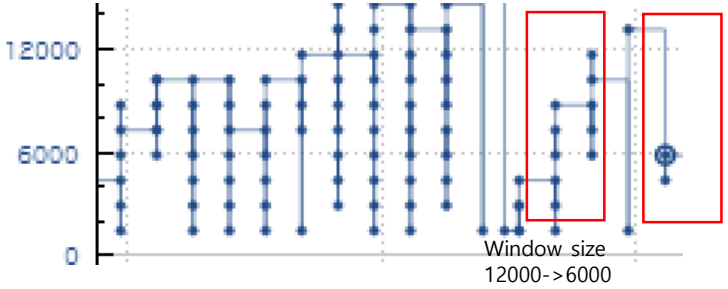
- 이는 Window scaling에서 byte 그래프에서의 일부이다. TCP tahoe는 Window size를 늘이다 보니 중복적인 ack가 일정량 이상 쌓일 때(3-dulplicate ack) congestion이라고 판단하고 window size를 1로 줄인 후 additive하게 증가시키는 방식이다. 다음 segment들을 보면 Duplicate ACK와 Retransmission이 발생하였음을 확인할 수 있고 보내는 byte가 0에 근접하게 줄어든 것을 확인할 수 있다.
- 여기서 Out of order은 segment들의 순서가 뒤죽박죽으로 들어왔다는 것을 의미한다.

#1 TCP

Network programming



4817	46.241900	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 4816#1] 80 → 50994 [ACK] Seq=1 Ack=2401555 Win=306560 Len=0…
4818	46.241963	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2405935 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4819	46.242023	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2407395 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4820	46.242023	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2408855 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4821	46.242025	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2410315 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4822	46.242025	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2411775 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4842	46.456003	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 4816#2] 80 → 50994 [ACK] Seq=1 Ack=2401555 Win=306560 Len=0…
4843	46.456092	172.16.24.116	128.119.245.12	TCP	1514 [TCP Fast Retransmission] 50994 → 80 [ACK] Seq=2401555 Ack=1 Win=262144 …
4844	46.456176	172.16.24.116	128.119.245.12	TCP	1514 50994 → 80 [ACK] Seq=2413235 Ack=1 Win=262144 Len=1460 [TCP segment of a…
4861	46.675160	128.119.245.12	172.16.24.116	TCP	60 80 → 50994 [ACK] Seq=1 Ack=2411775 Win=302592 Len=0
4862	46.675160	128.119.245.12	172.16.24.116	TCP	66 [TCP Dup ACK 4861#1] 80 → 50994 [ACK] Seq=1 Ack=2411775 Win=302592 Len=0…

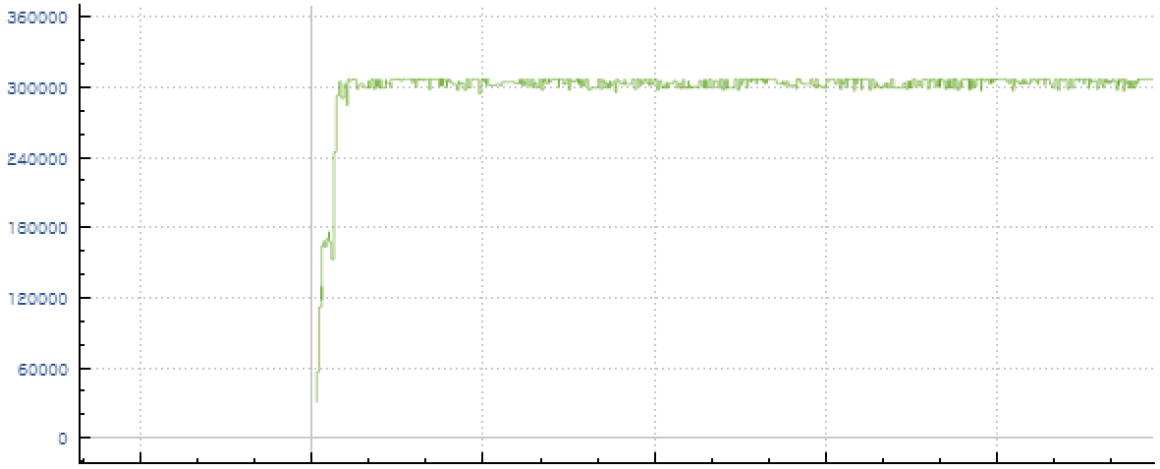


Window size
12000 이상->6000-6000이하

- 그래프를 분석하다가 배운 내용과 다르게 관측 된 점 중 하나이다. 다음을 보았을 때는 TCP reno 방식으로 congestion control을 하고 있다고 유추할 법한 경우도 있었다. TCP reno는 window size를 treshold까지 줄이는 방식이다. 왼쪽의 그래프를 보면 Dup ACK가 발생하였고 보내는 byte수가 절반 수준으로 감소하였다

#1 TCP

Network programming



앞의 내용에서 Window size에 따라서 전송되는 byte 수가 조절된다는 점을 근거로 Congestion control 방식을 전송 byte에 근거하여 설명하였다 그러나 Window scaling에서 관측할 수 있는 Window size가 receiver 입장에서 보게 되어 Size를 0으로 떨어뜨리는 tahoe도, 반으로 떨어뜨리는 reno도 근거를 들어 설명하기에는 부족하므로 시간당 전송 byte에 근거하여 Congestion control을 설명하였다.

Discussion

Network programming

TCP 동작 방식을 알아보기 위해 위의 일련의 과정을 거쳐 보았다. 이로 인해 알 수 있었던 점이 몇 가지 있다.

1. TCP segment들의 retransmission이 이루어 진다는 것은 혼잡하다는 것을 의미한다. 이 부분에서 그래프를 관측해 보면 Window size를 줄이는 등의 Congestion control이 이루어지고 있음을 확인할 수 있었다.

2. Sequence 그래프와 Window size 그래프를 통해서 급격하게 Window size양을 늘여 데이터를 보내는 slow start를 관측할 수 있었으며 retransmission 과정에서 Window size가 1로 감소하는 것을 보면 TCP Congestion control의 방식 중 Tahoe의 방식을 사용하는 것을 확인할 수 있었다.

3. 파일의 크기가 그렇게 크지 않아서인지 timeout은 발생하지 않았으며 이를 통해 심각한 Congestion은 일어나지 않았고 적절히 Control하였음을 확인할 수 있었다.