Arsenij Percov

Homework 4

Problem 4.1

Solution:
a)
SEQ = 1030 (ACK doesn't occupy any space, so SEQ didn't increase compared to segment above it)
ACk = 3848
F = ACK
WIN = 4000 (No way to deduct it, since soonly we send another message updating it, and in between only 1 package of size 0 was sent, so I just assume it is 4000)
b)
There might have happened few distinct scenarios:
- The buffer of the server got emptied, so now it needs to send a window update, indicating that we again have 4000 bytes of space if buffer for the incoming data.
- Server before experienced some type of issues, which slowed it down, so it could only deal with 200 bytes, and after, once the server is running with normal speed, we speed up the transfer by giving it bigger window.
Additionally, the reason why no data got passed in between (judging by acknowledgments) might be the fact that window of 200 was to small, and client was trying to send packages of size 1200.

c)
Selective acknowledgment sends additional information to the recipient on the other side of connection in case if we lost segment, which is not the last segment that was sent (we received a segment with sequence number higher than expected). Usually, we would just send an acknowledgment for the last continuous segment, and ask server to send the data after this segment again. With selective acknowledgment we specify the borders (in number of byte) of the received packages after the missing one, so the sender doesn't have to waste time resending those packages. If we send only part of missing packages, the selective acknowledgment will update its block borders.
The two numbers in the block indicate the left border and the right border of the block (note that borders indicated the received data intervals, not segments , so 3 continuous segments after the missing one would merge in 1 block in SACK)

d)
In 8th package to indicate the missing segment. 3430 and 4630.

e)
Client send ACK,FIN in segment 14, initiating closing procedure. Client goes to the FIN-WAIT-1 state.
Server recieves the ACK,FIN and goes to CLOSE-WAIT sending the ACK, closing , and sending FIN as well (in this case both ACK and FYN are sent with same segment 15). That results in server entering LAST-ACK state. Client, receiving ACK, goes to FINWAIT-2.
Last, it segment 16, client sends ACK and enters TIMEWAIT. Server recieves the ACK and enters CLOSED state.

Problem 4.2

Solution:
a)
Max ACK was around 12850000 (1285000-0)/12 = 107083 bytes/s b)
Minimum : around 30000
Maximum: around 300000

c)
6 segments. (Holes in the SACKS (red lines))
d)
Advertised window grows. A lot of packages go lost before 1,6s. Then while recovering them (1,6 - 2,3), some got lost again, and that needed to be recovered again (until 3,3 approximately). After that connection gets normal. We can't know the reason from the graph.