Lab - 07

Introduction to TCP(Transmission Control Protocol) and Analysis using Netsim

Program: MScIT

Sem-2

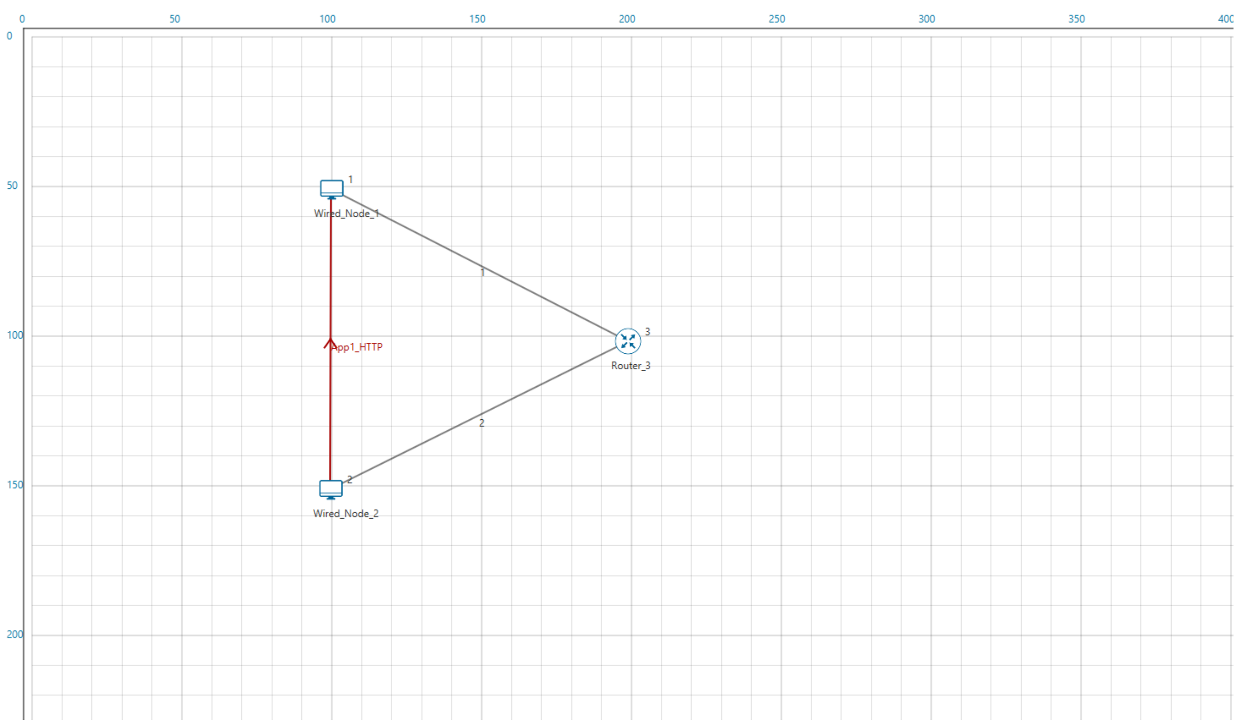
Group ID : 28

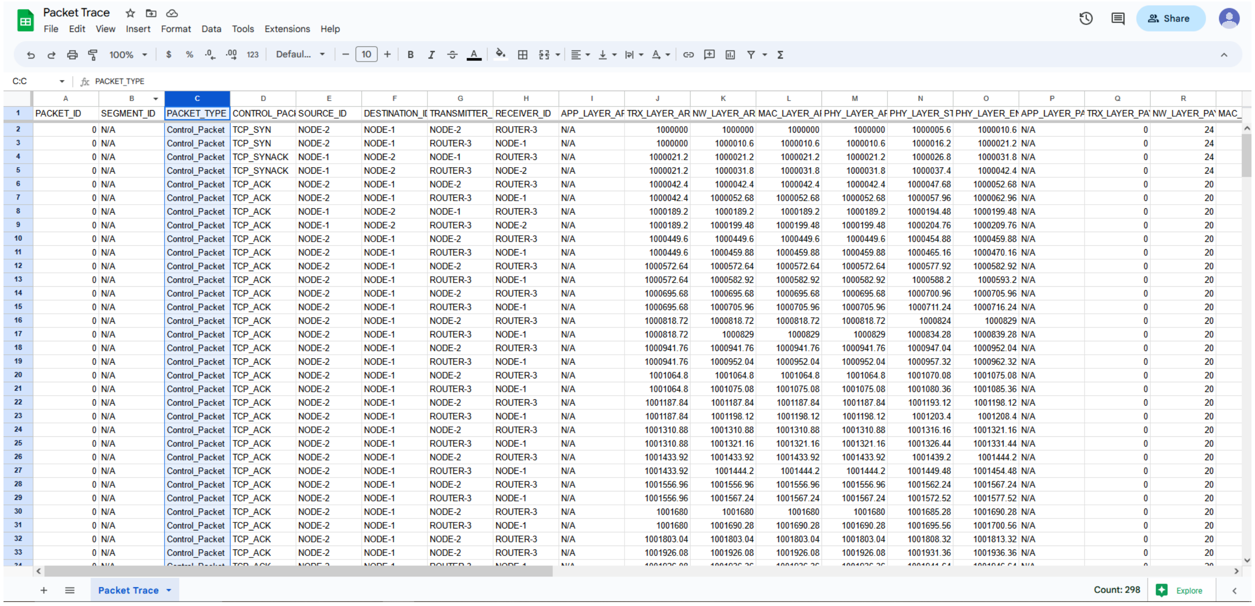
Student Name	Student ID
Dev Adnani	202212012
Saif Saiyed	202212083

2.1.1 Experiment

To encounter three way handshaking perform the steps mention below.

1. Establish topology shown in figure 2, with two wired node, one router with HTTP application between
two nodes.
2. Set parameters for application: Application Method: Unicast and END Time(s): 10
3. Run simulation for 5 second.
4. Open packet trace file and Consider following Columns: PACKET ID, PACKET TYPE, CONTROL PACKET TYPE, DESTINATION ID, TRANSMITTER ID,RECEIVER ID, SEQ NO, is Syn,is Ack,is Fin, SEGMENT lEN, Remove rest of the columns.
5. Filter PACKET TYPE by selecting only Control Packets.
6. After filter, you can see the SYN flag transmitted from Node 2 to Node 1 through Router, analyze the value of is Syn and is Ack columns.|





	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	
1	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACI	SOURCE_ID	DESTINATION_I	TRANSMITTER_	RECEIVER_ID	APP_LAYER_AF	TRX_LAYER_AR	NW_LAYER_AR	MAC_LAYER_AI	PHY_LAYER_AF	PHY_LAYER_ST	PHY_LAYER_EN	APP_LAYER_PA	TRX_LAYER_PA	NW_LAYER_PA	MAC_
2	0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000000	1000000	1000000	1000000	1000005.6	1000010.6	N/A	0	24	
3	0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000000	1000010.6	1000010.6	1000010.6	1000016.2	1000021.2	N/A	0	24	
4	0	N/A	Control_Packet	TCP_SYNACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	1000021.2	1000021.2	1000021.2	1000021.2	1000026.8	1000031.8	N/A	0	24	
5	0	N/A	Control_Packet	TCP_SYNACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	1000021.2	1000031.8	1000031.8	1000031.8	1000037.4	1000042.4	N/A	0	24	
6	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000042.4	1000042.4	1000042.4	1000042.4	1000047.68	1000052.68	N/A	0	20	
7	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000042.4	1000052.68	1000052.68	1000052.68	1000057.96	1000062.96	N/A	0	20	
8	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	1000189.2	1000189.2	1000189.2	1000189.2	1000194.48	1000199.48	N/A	0	20	
9	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	1000189.2	1000199.48	1000199.48	1000199.48	1000204.76	1000209.76	N/A	0	20	
10	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000449.6	1000449.6	1000449.6	1000449.6	1000454.88	1000459.88	N/A	0	20	
11	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000449.6	1000459.88	1000459.88	1000459.88	1000465.16	1000470.16	N/A	0	20	
12	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000572.64	1000572.64	1000572.64	1000572.64	1000577.92	1000582.92	N/A	0	20	
13	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000572.64	1000582.92	1000582.92	1000582.92	1000588.2	1000593.2	N/A	0	20	
14	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000695.68	1000695.68	1000695.68	1000695.68	1000700.96	1000705.96	N/A	0	20	
15	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000695.68	1000705.96	1000705.96	1000705.96	1000711.24	1000716.24	N/A	0	20	
16	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000818.72	1000818.72	1000818.72	1000818.72	1000824	1000829	N/A	0	20	
17	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000818.72	1000829	1000829	1000829	1000834.28	1000839.28	N/A	0	20	
18	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000941.76	1000941.76	1000941.76	1000941.76	1000947.04	1000952.04	N/A	0	20	
19	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000941.76	1000952.04	1000952.04	1000952.04	1000957.32	1000962.32	N/A	0	20	
20	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001064.8	1001064.8	1001064.8	1001064.8	1001070.08	1001075.08	N/A	0	20	
21	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001064.8	1001075.08	1001075.08	1001075.08	1001080.36	1001085.36	N/A	0	20	
22	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001187.84	1001187.84	1001187.84	1001187.84	1001193.12	1001198.12	N/A	0	20	
23	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001187.84	1001198.12	1001198.12	1001198.12	1001203.4	1001208.4	N/A	0	20	
24	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001310.88	1001310.88	1001310.88	1001310.88	1001316.16	1001321.16	N/A	0	20	
25	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001310.88	1001321.16	1001321.16	1001321.16	1001326.44	1001331.44	N/A	0	20	
26	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001433.92	1001433.92	1001433.92	1001433.92	1001439.2	1001444.2	N/A	0	20	
27	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001433.92	1001444.2	1001444.2	1001444.2	1001449.48	1001454.48	N/A	0	20	
28	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001556.96	1001556.96	1001556.96	1001556.96	1001562.24	1001567.24	N/A	0	20	
29	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001556.96	1001567.24	1001567.24	1001567.24	1001572.52	1001577.52	N/A	0	20	
30	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001680	1001680	1001680	1001680	1001685.28	1001690.28	N/A	0	20	
31	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001680	1001690.28	1001690.28	1001690.28	1001695.56	1001700.56	N/A	0	20	
32	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001803.04	1001803.04	1001803.04	1001803.04	1001808.32	1001813.32	N/A	0	20	
33	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001926.08	1001926.08	1001926.08	1001926.08	1001931.36	1001936.36	N/A	0	20	

2.1.1 Experiment

1. After connection establishment, continue with the same packet trace file, Add one more filter, CONTROL PACKET TYPE: TCP ACK and TCP FIN

	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACI	SOURCE_ID	DESTINATION_I	TRANSMITTER_	RECEIVER_ID	APP_LAYER_AF	TRX_LAYER_AR	NW_LAYER_AR	MAC_LAYER_AI	PHY_LAYER_AF	PHY_LAYER_ST	PHY_LAYER_EI	APP_LAYER_PA	TRX_LAYER_PA	NW_LAYER_PA	MAC_
2	0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000000	1000000	1000000	1000000	1000005.6	1000010.6	N/A	0	24	
3	0	N/A	Control_Packet	TCP_SYN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000000	1000010.6	1000010.6	1000010.6	1000016.2	1000021.2	N/A	0	24	
4	0	N/A	Control_Packet	TCP_SYNACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	1000021.2	1000021.2	1000021.2	1000021.2	1000026.8	1000031.8	N/A	0	24	
5	0	N/A	Control_Packet	TCP_SYNACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	1000021.2	1000031.8	1000031.8	1000031.8	1000037.4	1000042.4	N/A	0	24	
6	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000042.4	1000042.4	1000042.4	1000042.4	1000047.68	1000052.68	N/A	0	20	
7	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000042.4	1000052.68	1000052.68	1000052.68	1000057.96	1000062.96	N/A	0	20	
8	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	1000189.2	1000189.2	1000189.2	1000189.2	1000194.48	1000199.48	N/A	0	20	
9	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	1000189.2	1000199.48	1000199.48	1000199.48	1000204.76	1000209.76	N/A	0	20	
10	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000449.6	1000449.6	1000449.6	1000449.6	1000454.88	1000459.88	N/A	0	20	
11	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000449.6	1000459.88	1000459.88	1000459.88	1000465.16	1000470.16	N/A	0	20	
12	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000572.64	1000572.64	1000572.64	1000572.64	1000577.92	1000582.92	N/A	0	20	
13	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000572.64	1000582.92	1000582.92	1000582.92	1000588.2	1000593.2	N/A	0	20	
14	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000695.68	1000695.68	1000695.68	1000695.68	1000700.96	1000705.96	N/A	0	20	
15	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000695.68	1000705.96	1000705.96	1000705.96	1000711.24	1000716.24	N/A	0	20	
16	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000818.72	1000818.72	1000818.72	1000818.72	1000824	1000829	N/A	0	20	
17	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000818.72	1000829	1000829	1000829	1000834.28	1000839.28	N/A	0	20	
18	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1000941.76	1000941.76	1000941.76	1000941.76	1000947.04	1000952.04	N/A	0	20	
19	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1000941.76	1000952.04	1000952.04	1000952.04	1000957.32	1000962.32	N/A	0	20	
20	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001064.8	1001064.8	1001064.8	1001064.8	1001070.08	1001075.08	N/A	0	20	
21	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001064.8	1001075.08	1001075.08	1001075.08	1001080.36	1001085.36	N/A	0	20	
22	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001187.84	1001187.84	1001187.84	1001187.84	1001193.12	1001198.12	N/A	0	20	
23	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001187.84	1001198.12	1001198.12	1001198.12	1001203.4	1001208.4	N/A	0	20	
24	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001310.88	1001310.88	1001310.88	1001310.88	1001316.16	1001321.16	N/A	0	20	
25	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001310.88	1001321.16	1001321.16	1001321.16	1001326.44	1001331.44	N/A	0	20	
26	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001433.92	1001433.92	1001433.92	1001433.92	1001439.2	1001444.2	N/A	0	20	
27	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001433.92	1001444.2	1001444.2	1001444.2	1001449.48	1001454.48	N/A	0	20	
28	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001556.96	1001556.96	1001556.96	1001556.96	1001562.24	1001567.24	N/A	0	20	
29	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001556.96	1001567.24	1001567.24	1001567.24	1001572.52	1001577.52	N/A	0	20	
30	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001680	1001680	1001680	1001680	1001685.28	1001690.28	N/A	0	20	
31	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1001680	1001690.28	1001690.28	1001690.28	1001695.56	1001700.56	N/A	0	20	
32	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001803.04	1001803.04	1001803.04	1001803.04	1001808.32	1001813.32	N/A	0	20	
33	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1001926.08	1001926.08	1001926.08	1001926.08	1001931.36	1001936.36	N/A	0	20	

Count: 298

2. You can see the FIN flag transmitted over destination through router, analyze the value of is Fin and is Ack

	PACKET_ID	SEGMENT_ID	PACKET_TYI	CONTROL_P	SOURCE_ID	DESTINATION_I	TRANSMITTER_	RECEIVER_ID	APP_LAYER_AF	TRX_LAYER_AR	NW_LAYER_AR	MAC_LAYER_AI	PHY_LAYER_AF	PHY_LAYER_ST	PHY_LAYER_EI	APP_LAYER_PA	TRX_LAYER_PA	NW_LAYER_PA	MAC_
253	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002124.56	4002124.56	4002124.56	4002124.56	4002129.84	4002134.84	N/A	0	20	
254	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002124.56	4002134.84	4002134.84	4002134.84	4002140.12	4002145.12	N/A	0	20	
255	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002247.6	4002247.6	4002247.6	4002247.6	4002252.88	4002257.88	N/A	0	20	
256	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002247.6	4002257.88	4002257.88	4002257.88	4002263.16	4002268.16	N/A	0	20	
257	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002370.64	4002370.64	4002370.64	4002370.64	4002375.92	4002380.92	N/A	0	20	
258	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002370.64	4002380.92	4002380.92	4002380.92	4002386.2	4002391.2	N/A	0	20	
259	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002493.68	4002493.68	4002493.68	4002493.68	4002498.96	4002503.96	N/A	0	20	
260	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002493.68	4002503.96	4002503.96	4002503.96	4002509.24	4002514.24	N/A	0	20	
261	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002581.52	4002581.52	4002581.52	4002581.52	4002586.8	4002591.8	N/A	0	20	
262	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002581.52	4002591.8	4002591.8	4002591.8	4002597.08	4002602.08	N/A	0	20	
263	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002622.64	4002622.64	4002622.64	4002622.64	4002627.92	4002632.92	N/A	0	20	
264	0	N/A	Control_Packet	TCP_ACK	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002622.64	4002632.92	4002632.92	4002632.92	4002638.2	4002643.2	N/A	0	20	
265	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	4002649.44	4002649.44	4002649.44	4002649.44	4002654.72	4002659.72	N/A	0	20	
266	0	N/A	Control_Packet	TCP_ACK	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	4002649.44	4002659.72	4002659.72	4002659.72	4002665	4002670	N/A	0	20	
267	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	1002034.48	1002034.48	1002034.48	1002034.48	1002039.76	1002044.76	N/A	0	20	
268	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	1002034.48	1002044.76	1002044.76	1002044.76	1002050.04	1002055.04	N/A	0	20	
269	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	1002055.04	1002055.04	1002055.04	1002061.28	1002066.56	1002071.56	N/A	0	20	
270	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	1002055.04	1002071.56	1002071.56	1002071.56	1002076.84	1002081.84	N/A	0	20	
271	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	2002223.68	2002223.68	2002223.68	2002223.68	2002228.96	2002233.96	N/A	0	20	
272	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	2002223.68	2002233.96	2002233.96	2002233.96	2002239.24	2002244.24	N/A	0	20	
273	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	2002244.24	2002244.24	2002244.24	2002250.48	2002255.76	2002260.76	N/A	0	20	
274	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	2002244.24	2002260.76	2002260.76	2002260.76	2002266.04	2002271.04	N/A	0	20	
275	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	3002535.92	3002535.92	3002535.92	3002535.92	3002541.2	3002546.2	N/A	0	20	
276	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	3002535.92	3002546.2	3002546.2	3002546.2	3002551.48	3002556.48	N/A	0	20	
277	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	3002556.48	3002556.48	3002556.48	3002562.72	3002568	3002573	N/A	0	20	
278	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	3002556.48	3002573	3002573	3002573	3002578.28	3002583.28	N/A	0	20	
279	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	ROUTER-3	N/A	4002602.08	4002602.08	4002602.08	4002602.08	4002607.36	4002612.36	N/A	0	20	
280	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	NODE-2	N/A	4002602.08	4002612.36	4002612.36	4002612.36	4002617.64	4002622.64	N/A	0	20	
281	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	ROUTER-3	N/A	4002622.64	4002622.64	4002622.64	4002628.88	4002634.16	4002639.16	N/A	0	20	
282	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	NODE-1	N/A	4002622.64	4002639.16	4002639.16	4002639.16	4002644.44	4002649.44	N/A	0	20	

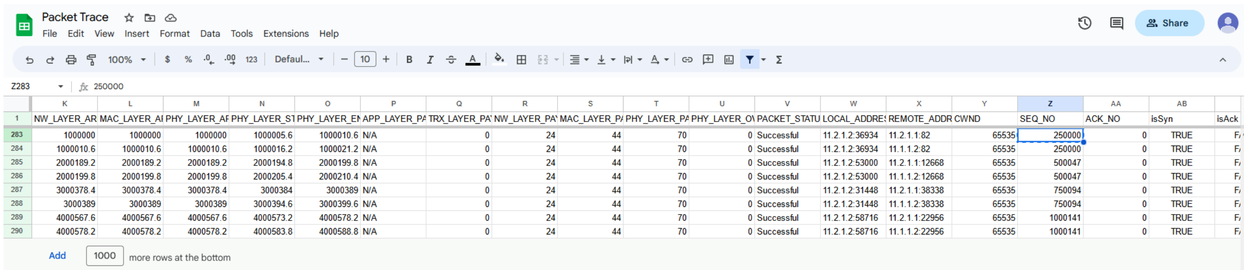
Add 1000 more rows at the bottom

isSyn: True
isAck: False

2.3 Exercise

1. What is the Sequence number of the 1st SYN control packet and its acknowledgment?

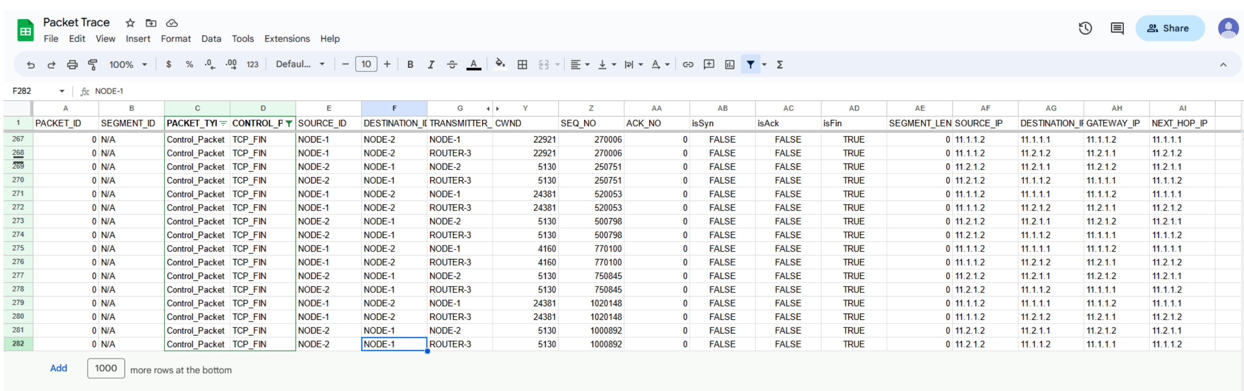
Answer : Sequence number is 250000 and acknowledgment is false



	NW_LAYER_AR	MAC_LAYER_AI	PHY_LAYER_AF	PHY_LAYER_ST	PHY_LAYER_EN	APP_LAYER_PA	TRX_LAYER_PA	NW_LAYER_PA	MAC_LAYER_PA	PHY_LAYER_PA	PHY_LAYER_OV	PACKET_STATU	LOCAL_ADDRES	REMOTE_ADDR	CWND	SEQ_NO	ACK_NO	isSyn	isAck
283	1000000	1000000	1000000	1000005.6	1000010.6	N/A	0	24	44	70	0	Successful	11.2.1.2:36934	11.2.1.1:82	65535	250000	0	TRUE	F
284	1000010.6	1000010.6	1000010.6	1000016.2	1000021.2	N/A	0	24	44	70	0	Successful	11.2.1.2:36934	11.1.1.2:82	65535	250000	0	TRUE	F
285	2000189.2	2000189.2	2000189.2	2000194.8	2000199.8	N/A	0	24	44	70	0	Successful	11.2.1.2:53000	11.2.1.1:12668	65535	500047	0	TRUE	F
286	2000199.8	2000199.8	2000199.8	2000205.4	2000210.4	N/A	0	24	44	70	0	Successful	11.2.1.2:53000	11.1.1.2:12668	65535	500047	0	TRUE	F
287	3000378.4	3000378.4	3000378.4	3000384	3000389	N/A	0	24	44	70	0	Successful	11.2.1.2:31448	11.2.1.1:38338	65535	750094	0	TRUE	F
288	3000389	3000389	3000389	3000394.6	3000399.6	N/A	0	24	44	70	0	Successful	11.2.1.2:31448	11.1.1.2:38338	65535	750094	0	TRUE	F
289	4000567.6	4000567.6	4000567.6	4000573.2	4000578.2	N/A	0	24	44	70	0	Successful	11.2.1.2:58716	11.2.1.1:22956	65535	1000141	0	TRUE	F
290	4000578.2	4000578.2	4000578.2	4000583.8	4000588.8	N/A	0	24	44	70	0	Successful	11.2.1.2:58716	11.1.1.2:22956	65535	1000141	0	TRUE	F

2. What is the sequence number of the 1st FIN control packet and its acknowledgement?

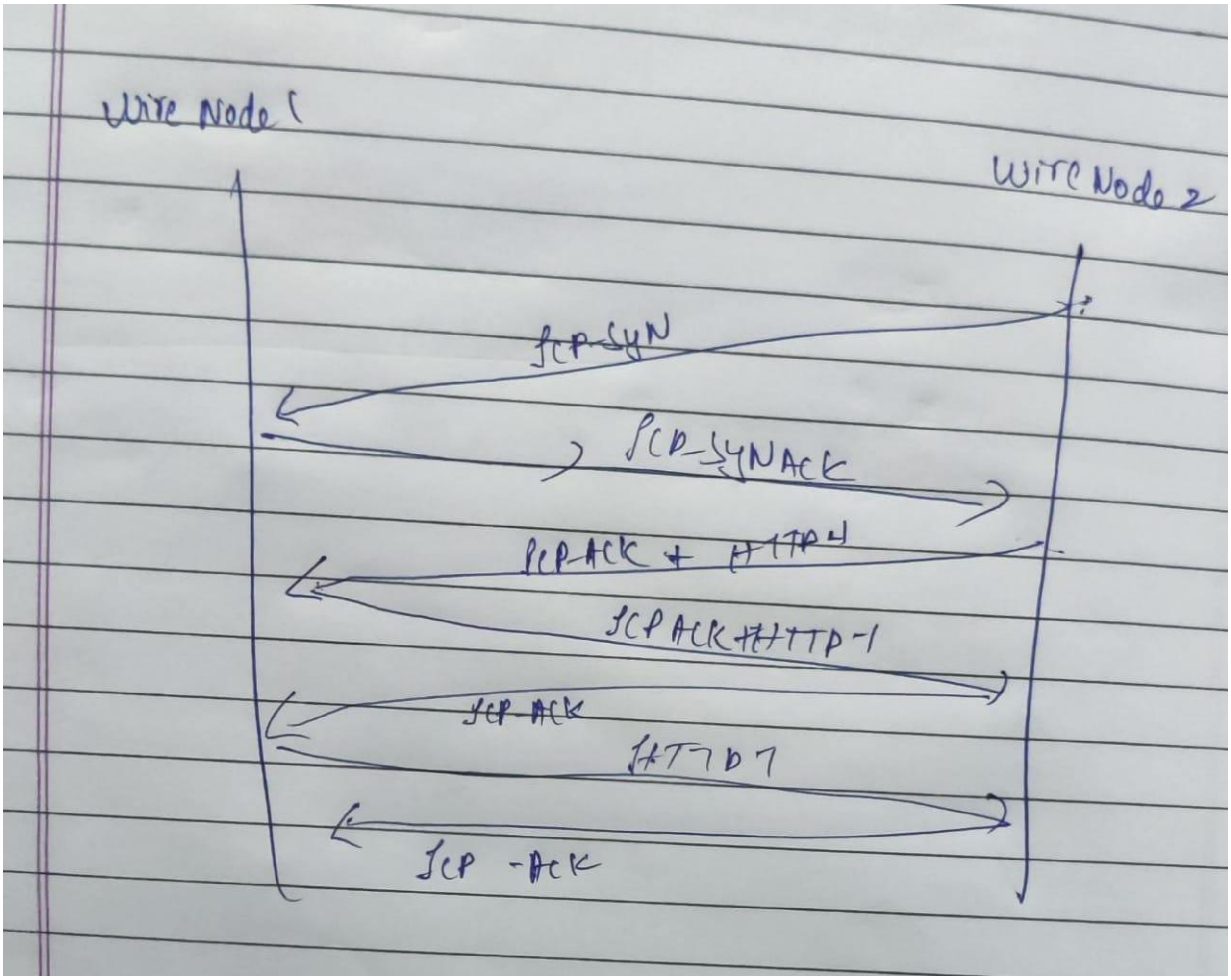
Answer : sequence number is 270006 and acknowledgement is false



	PACKET_ID	SEGMENT_ID	PACKET_TYI	CONTROL_P	SOURCE_ID	DESTINATION_I	TRANSMITTER_	CWND	SEQ_NO	ACK_NO	isSyn	isAck	isFin	SEGMENT_LEN	SOURCE_IP	DESTINATION_I	GATEWAY_IP	NEXT_HOP_IP
267	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	22921	270006	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.1.1.1	11.1.1.2	11.1.1.1
268	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	22921	270006	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.2.1.2	11.2.1.1	11.2.1.2
269	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	5130	250751	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.2.1.1	11.2.1.2	11.2.1.1
270	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	5130	250751	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.1.1.2	11.1.1.1	11.1.1.2
271	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	24381	520053	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.1.1.1	11.1.1.2	11.1.1.1
272	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	24381	520053	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.2.1.2	11.2.1.1	11.2.1.2
273	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	5130	500798	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.2.1.1	11.2.1.2	11.2.1.1
274	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	5130	500798	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.1.1.2	11.1.1.1	11.1.1.2
275	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	4160	770100	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.1.1.1	11.1.1.2	11.1.1.1
276	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	4160	770100	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.2.1.2	11.2.1.1	11.2.1.2
277	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	5130	750845	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.2.1.1	11.2.1.2	11.2.1.1
278	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	5130	750845	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.1.1.2	11.1.1.1	11.1.1.2
279	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	NODE-1	24381	1020148	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.1.1.1	11.1.1.2	11.1.1.1
280	0	N/A	Control_Packet	TCP_FIN	NODE-1	NODE-2	ROUTER-3	24381	1020148	0	FALSE	FALSE	TRUE	0	11.1.1.2	11.2.1.2	11.2.1.1	11.2.1.2
281	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	NODE-2	5130	1000892	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.2.1.1	11.2.1.2	11.2.1.1
282	0	N/A	Control_Packet	TCP_FIN	NODE-2	NODE-1	ROUTER-3	5130	1000892	0	FALSE	FALSE	TRUE	0	11.2.1.2	11.1.1.2	11.1.1.1	11.1.1.2

3. Draw the Diagram of Connection establishment and termination as shown in figure 1 and 3 only with sequence number of each packet in you log book.

Answer :



4. Why TCP uses 4 way finishing for connection termination instead of 3way like connection establishment?

Answer :

TCP uses a 4-way handshake for connection termination to ensure that both the client and server have completely closed the connection and that all data has been successfully transferred.

In a 3-way handshake for connection establishment, the client sends a SYN packet to the server, the server responds with a SYN-ACK packet, and the client sends an ACK packet to confirm the connection.

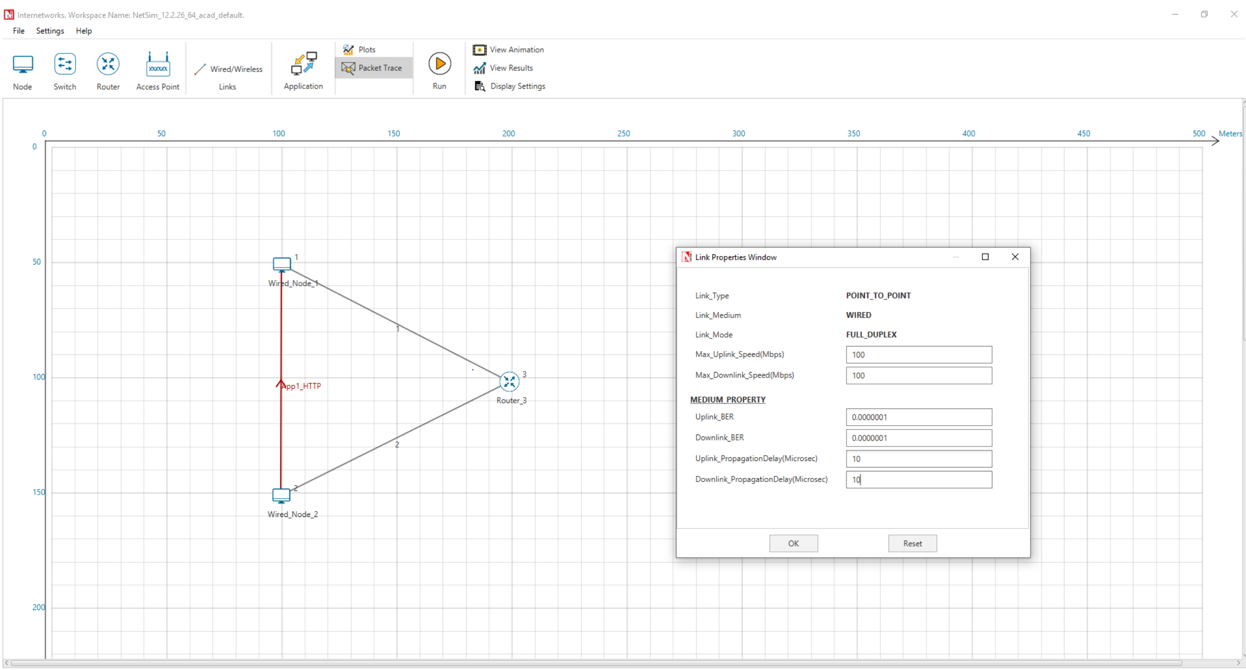
During a connection termination, the client sends a FIN packet to the server to indicate that it has no more data to send. The server then responds with an ACK packet to confirm that it has received the FIN packet. However, the server may still have data to send to the client, so it sends a FIN packet to the client to indicate that it has no more data to send. Finally, the client responds with an ACK packet to confirm that it has received the FIN packet.

By using a 4-way handshake, TCP ensures that all data has been successfully transmitted and acknowledged by both parties before the connection is fully terminated.
This helps to avoid potential data loss or corruption that could occur if the connection was terminated abruptly with a 3-way handshake.

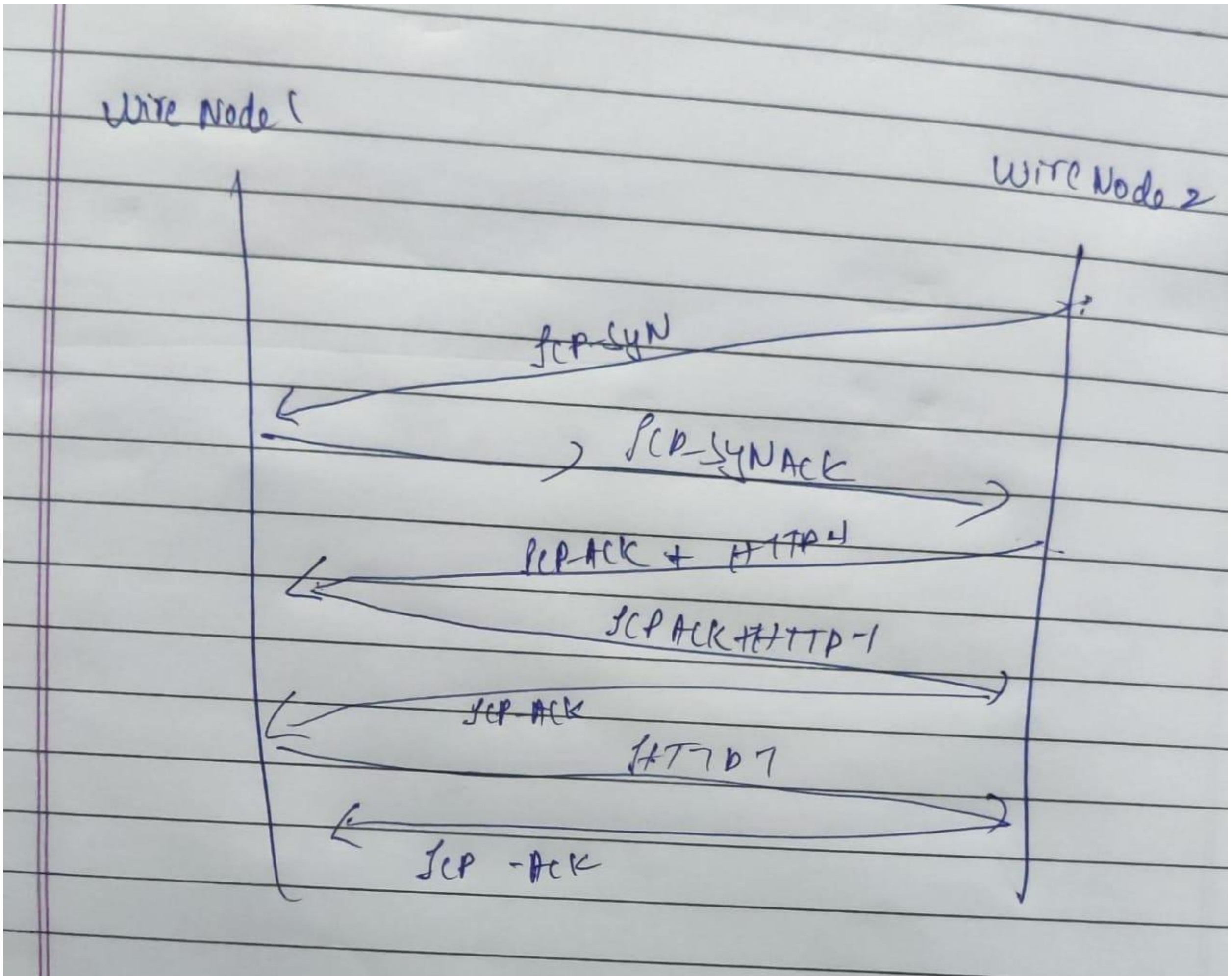
5. How many sessions it takes to transfer all data in this application?
Answer : Total 130 http sessions

6. Save this experiment as EX:1 for further lab session.
Saved

7. Start new experiment. Consider the same topology and configuration of EX: 1. Modify the property of link 1 according to this, Set Uplink Bit Error Rate and Download Bit Error Rate: 0.00001, Uplink Propagation Delay and Download Propagation Delay: 10 microsecond. Run the simulation for 10 seconds. Observe all rows of PACKET ID:1 with SEGMENT ID 1,2. Draw the diagram of transmission of packet id 1 for segment no 1 and 2 until successfully received with sequence number and acknowledgement number.



	PACKET_ID	SEGMENT_ID	PACKET_TYPE	CONTROL_PACKET_TYPE/APP_NAME	SOURCE_ID	DESTINATION_ID	TRANSMITTER_ID	RECEIVER_ID	SEQ_NO	ACK_NO	isSyn	isAck	isFin	SEGMENT_LEN
12	1	1	HTTP	App1_HTTP	NODE-1	NODE-2	NODE-1	ROUTER-3	250008	0	FALSE	FALSE	FALSE	1460
14	1	1	HTTP	App1_HTTP	NODE-1	NODE-2	ROUTER-3	NODE-2	250008	0	FALSE	FALSE	FALSE	1460
22	1	1	HTTP	App1_HTTP	NODE-1	NODE-2	NODE-1	ROUTER-3	250008	0	FALSE	FALSE	FALSE	1460
24	1	1	HTTP	App1_HTTP	NODE-1	NODE-2	ROUTER-3	NODE-2	250008	0	FALSE	FALSE	FALSE	1460


Wire Node 1
Wire Node 2
TCP-SYN
TCP-SYNACK
TCP-ACK + HTTP-1
TCP ACK+HTTP-1
TCP-ACK
HTTP 7
TCP -ACK

8. The data transfer initiated by from node 1 to node 2. If SYN packet have sequence number 5460, there were 5000 bytes of total data transmitted through network in one session, maximum segment size were 1500 bytes then what will be the sequence number of last packet and FIN packet?

SYN packet sequence number = 5460
Total data transmitted = 5000 bytes
Maximum segment size = 1500 bytes

To calculate the sequence number of the last packet, we need to divide the total data by the maximum segment size and round up to the nearest integer to determine the total number of packets needed for transmission:

Total number of packets = ceil(5000/1500) = 4
To calculate the sequence number of the last packet, we need to add the sequence number of the first packet (SYN packet) and the cumulative size of all packets except the last one (3 packets x 1500 bytes/packet = 4500 bytes), and then add the size of the last packet: Sequence number of last packet = 5460 + 4500 + 500 = 10460

To calculate the sequence number of the FIN packet, we need to add 1 to the sequence Number of the last packet:

Sequence number of FIN packet = 10460 + 1 = 10461
Therefore, the sequence number of the last packet is 10460 and the sequence number of The FIN packet is 10461.

3.1 Experiment

1. Open Experiment Ex:1

2. Run simulation for 2 seconds.

3. Open packet trace file. filter the field CONTROL PACKET TYPE: APP1 HTTP, HTTP REQUEST

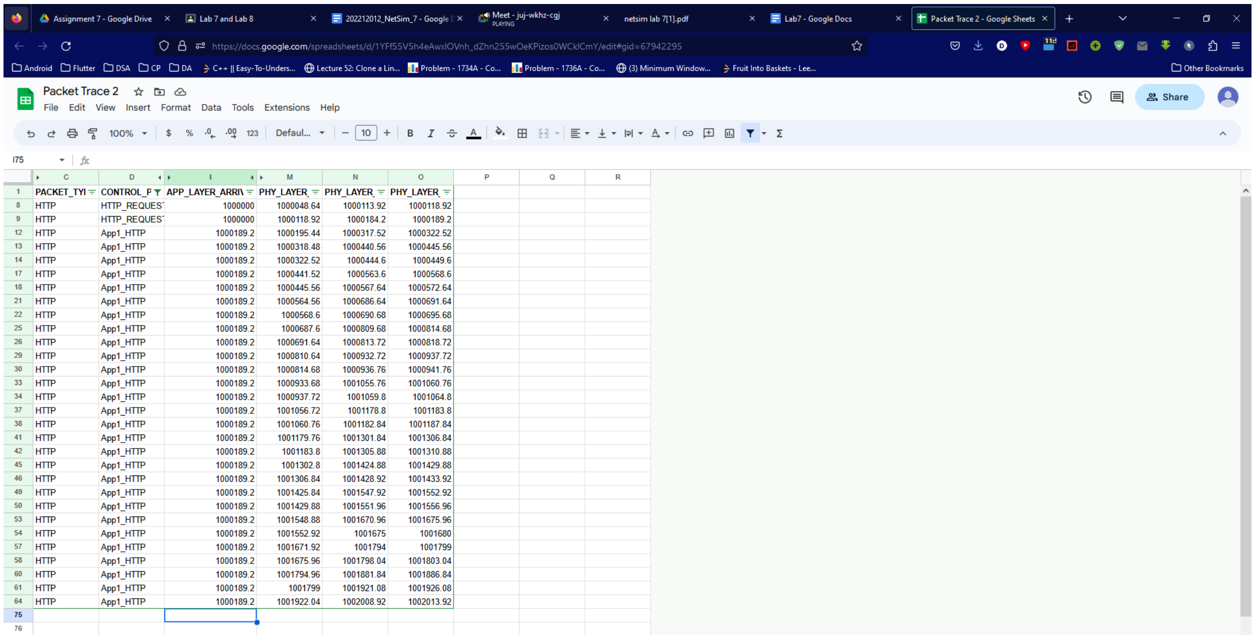
4. Consider PHYSICAL LAYER END TIME, APPLICATION LAYER ARRIVAL TIME, PHY LAYER PAYLOAD

5. Calculate throughput:

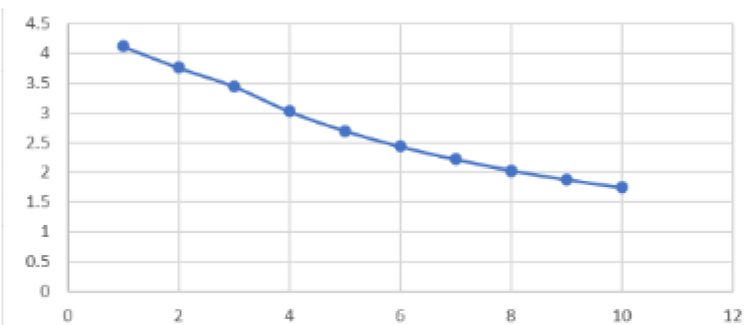
APPLICATION LAYER ARRIVAL TIME - PHYSICAL LAYER END TIME = 17003288-21509980.08

Fix throughput = 24522 / 4506692.08
=0.0054412415

• Fix throughput: Calculate total payload and divide by total difference of time.
(APPLICATION LAYER ARRIVAL TIME - PHYSICAL LAYER END TIME) .
• Moving average throughput
(a) Except APPLICATION LAYER ARRIVAL TIME, PHYSICAL LAYER END TIME, PHY LAYER PAYLOAD, you can clear all other columns for convince.
(b) Calculate time difference.
(APPLICATION LAYER ARRIVAL TIME - PHYSICAL LAYER END TIME) of 1st 10 rows and respectively sum of total payload of 1st 10 rows and copy both values in two separate columns A and B.
(c) Similarly calculate time difference for 2nd to 11th rows and total payload for same rows, then for 3rd to 12th, 4th to 13th, 5th to 14th, 6th to 15th up to 11th to 20th rows respectively.
(d) In third column C calculate the throughput by dividing total payload bytes (column B) by time difference (column A).
(e) Select Column C and select the scatter graph with smooth lines and markers.



	C	D	I	M	N	O
1	PACKET_TYI	CONTROL_P	APP_LAYER_ARRI\	PHY_LAYER_	PHY_LAYER_	PHY_LAYER_
8	HTTP	HTTP_REQUES	1000000	1000048.64	1000113.92	1000118.92
9	HTTP	HTTP_REQUES	1000000	1000118.92	1000184.2	1000189.2
12	HTTP	App1_HTTP	1000189.2	1000195.44	1000317.52	1000322.52
13	HTTP	App1_HTTP	1000189.2	1000318.48	1000440.56	1000445.56
14	HTTP	App1_HTTP	1000189.2	1000322.52	1000444.6	1000449.6
17	HTTP	App1_HTTP	1000189.2	1000441.52	1000563.6	1000568.6
18	HTTP	App1_HTTP	1000189.2	1000445.56	1000567.64	1000572.64
21	HTTP	App1_HTTP	1000189.2	1000564.56	1000686.64	1000691.64
22	HTTP	App1_HTTP	1000189.2	1000568.6	1000690.68	1000695.68
25	HTTP	App1_HTTP	1000189.2	1000687.6	1000809.68	1000814.68
26	HTTP	App1_HTTP	1000189.2	1000691.64	1000813.72	1000818.72
29	HTTP	App1_HTTP	1000189.2	1000810.64	1000932.72	1000937.72
30	HTTP	App1_HTTP	1000189.2	1000814.68	1000936.76	1000941.76
33	HTTP	App1_HTTP	1000189.2	1000933.68	1001055.76	1001060.76
34	HTTP	App1_HTTP	1000189.2	1000937.72	1001059.8	1001064.8
37	HTTP	App1_HTTP	1000189.2	1001056.72	1001178.8	1001183.8
38	HTTP	App1_HTTP	1000189.2	1001060.76	1001182.84	1001187.84
41	HTTP	App1_HTTP	1000189.2	1001179.76	1001301.84	1001306.84
42	HTTP	App1_HTTP	1000189.2	1001183.8	1001305.88	1001310.88
45	HTTP	App1_HTTP	1000189.2	1001302.8	1001424.88	1001429.88
46	HTTP	App1_HTTP	1000189.2	1001306.84	1001428.92	1001433.92
49	HTTP	App1_HTTP	1000189.2	1001425.84	1001547.92	1001552.92
50	HTTP	App1_HTTP	1000189.2	1001429.88	1001551.96	1001556.96
53	HTTP	App1_HTTP	1000189.2	1001548.88	1001670.96	1001675.96
54	HTTP	App1_HTTP	1000189.2	1001552.92	1001675	1001680
57	HTTP	App1_HTTP	1000189.2	1001671.92	1001794	1001799
58	HTTP	App1_HTTP	1000189.2	1001675.96	1001798.04	1001803.04
60	HTTP	App1_HTTP	1000189.2	1001794.96	1001881.84	1001886.84
61	HTTP	App1_HTTP	1000189.2	1001799	1001921.08	1001926.08
64	HTTP	App1_HTTP	1000189.2	1001922.04	1002008.92	1002013.92



3.2 Exercise

1. What is the maximum throughput value, consider the graph.
Answer : 4.124645352

2. Calculate average throughput for the same experiment with simulation time 10 seconds. Is there any difference? Why?

Answer :
If the size and number of packets transmitted increases with longer simulation times, the average throughput may increase as well. This is because more data is transmitted in a longer simulation period, increasing the overall throughput.

= 0.0015376023721994 - 0.000216319722
= 0.0013212826501994

Difference is because with more simulation time then the previous experiment more amount of material will be passed between the two wired nodes resulting in greater throughput.

3. Consider data transmission between 2 device A and B. A have sent a total 1000 bytes of data, Maximum segment size will be 150 bytes. Sending rate of packet 10 bytes/second will be Packet number 2, 4, and 5 got errored. But before termination, Device B have received all 1000 bytes. Calculate the average throughput in unit bits/second.

First, we need to calculate the total number of packets sent by device A:
Total number of packets = Total bytes sent / Maximum segment size
Total number of packets = 1000 bytes / 150 bytes per packet
Total number of packets = 6.67, which we round up to 7 packets
Since packets 2, 4, and 5 were errored, only 4 packets were successfully received by device B.

The time it took for device A to send all 1000 bytes is:
Time = Total bytes sent / Sending rate per second
Time = 1000 bytes / 10 bytes per second
Time = 100 seconds

The throughput is the amount of data transmitted per unit time, and can be calculated as:

Throughput = Total data transmitted / Total time taken
Throughput = 1000 bytes / 100 seconds
Throughput = 10 bytes per second

To convert this to bits per second, we multiply by 8:
Throughput = 10 bytes per second * 8 bits per byte
Throughput = 80 bits per second
Therefore, the average throughput between devices A and B is 80 bits per second

Lab - 08

Analyzing Congestion Policy, RTT Of TCP And Working Of UDP Using NetSim And Wireshark.

Program: MScIT

Sem-2

Group ID : 28

Student Name	Student ID
Dev Adnani	202212012
Saif Saiyed	202212083

(1) Introduction Of Congestion Policy Of TCP

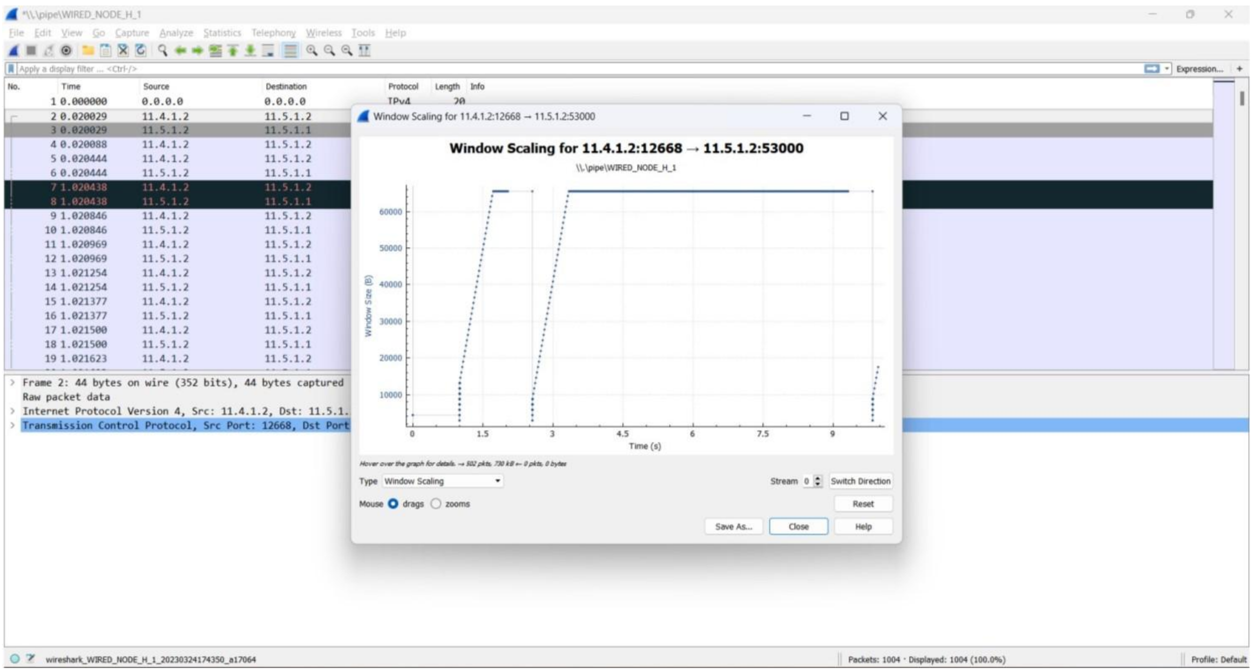
1.2 Exercise

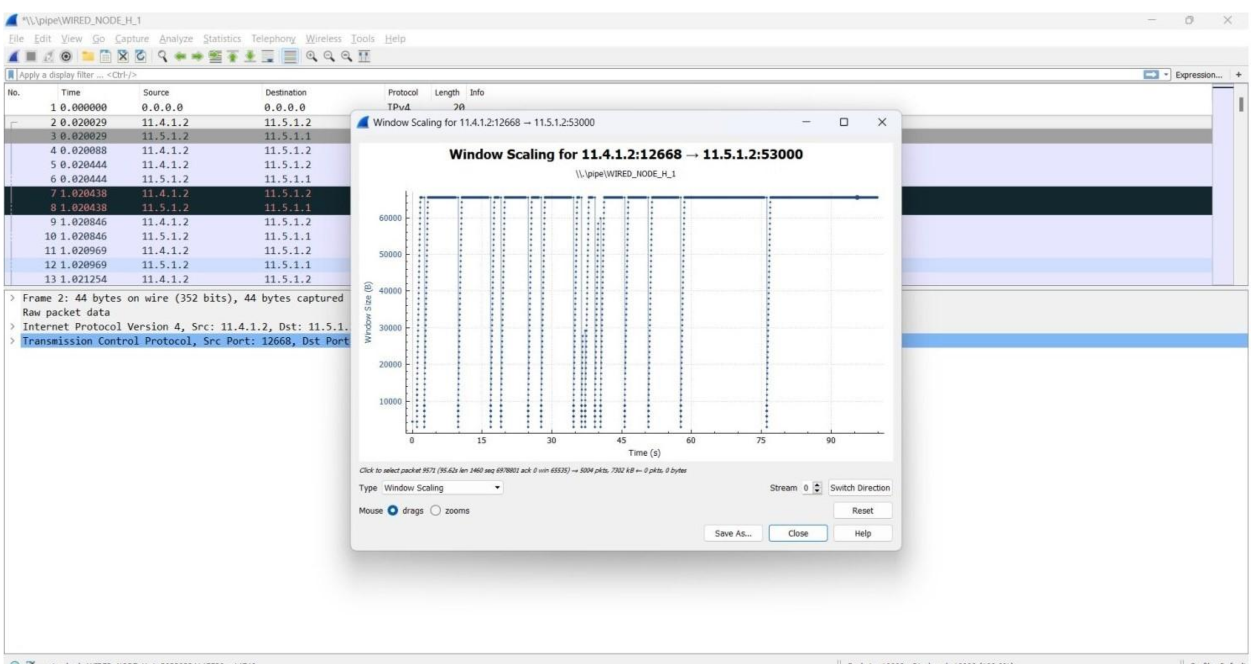
1. For both the variant, analyze graph of congestion window, answer the following by marking in the graph.

(a) Identify the event of TCP slow start.

(b) Identify the event of packet loss and time out.

(c) Identify the intervals of time when TCP congestion avoidance is operating.





2. What is the difference in congestion control policy of Tahoe and Reno, with respect to congestion avoidance and two events of congestion avoidance phase. Explain briefly in your log book.

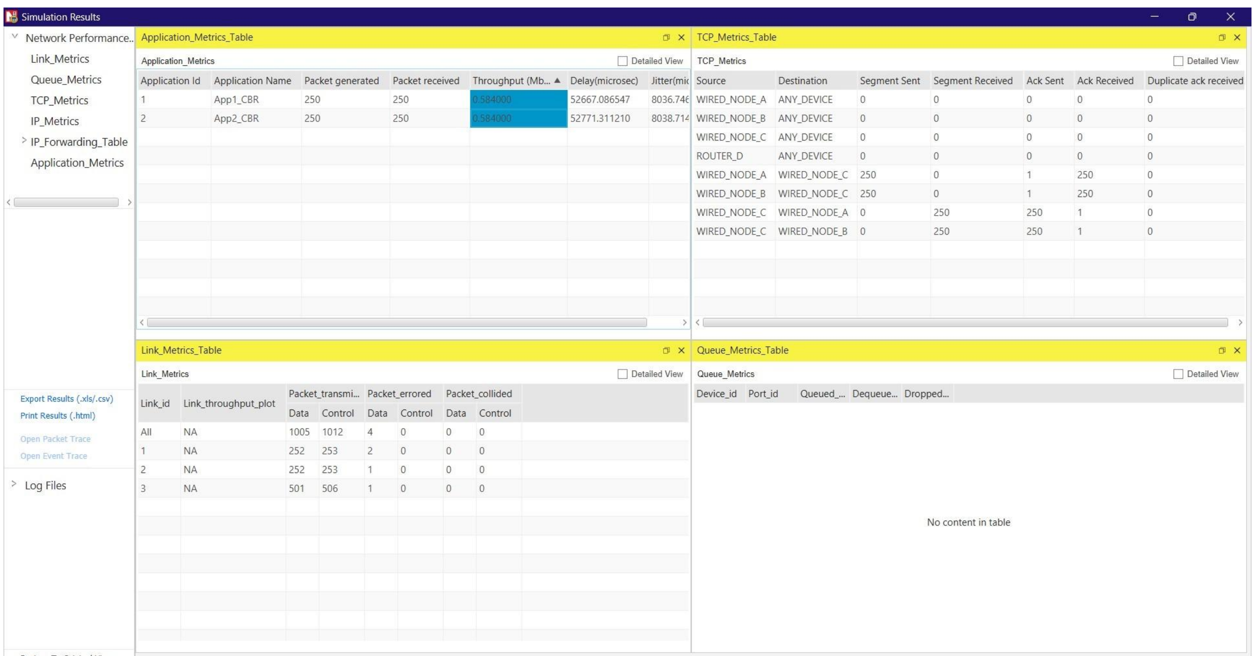
Ans:

Reno is more aggressive and adds a fast-recovery phase to avoid resetting the congestion window size to initial size in case of packet loss. Tahoe takes a more conservative strategy to congestion avoidance by resetting to slow-start phase after detecting congestion.

(2) Analyzing Fairness Of TCP

2.1 Experiment

1. Take 3 wired nodes and one router, configure 2 identical CBR applications with default app specification between them as shown in the figure4.
2. Keep link properties as default.
3. Run simulation for 5 seconds.
4. Check throughput for both the applications and write down your observation.



Application Id	Application Name	Packet generated	Packet received	Throughput (Mb...	Delay(microsec)	Jitter(mic
1	App1_CBR	250	250	0.584000	52667.086547	8036.746
2	App2_CBR	250	250	0.584000	52771.311210	8038.714

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
WIRED_NODE_A	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_B	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_C	ANY_DEVICE	0	0	0	0	0
ROUTER_D	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_A	WIRED_NODE_C	250	0	1	250	0
WIRED_NODE_B	WIRED_NODE_C	250	0	1	250	0
WIRED_NODE_C	WIRED_NODE_A	0	250	250	1	0
WIRED_NODE_C	WIRED_NODE_B	0	250	250	1	0

Link_id	Link_throughput_plot	Packet_transmi...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	1005	1012	4	0	0	0
1	NA	252	253	2	0	0	0
2	NA	252	253	1	0	0	0
3	NA	501	506	1	0	0	0

Device_id	Port_id	Queued_...	Dequeue...	Dropped...

(3) Analyzing Throughput

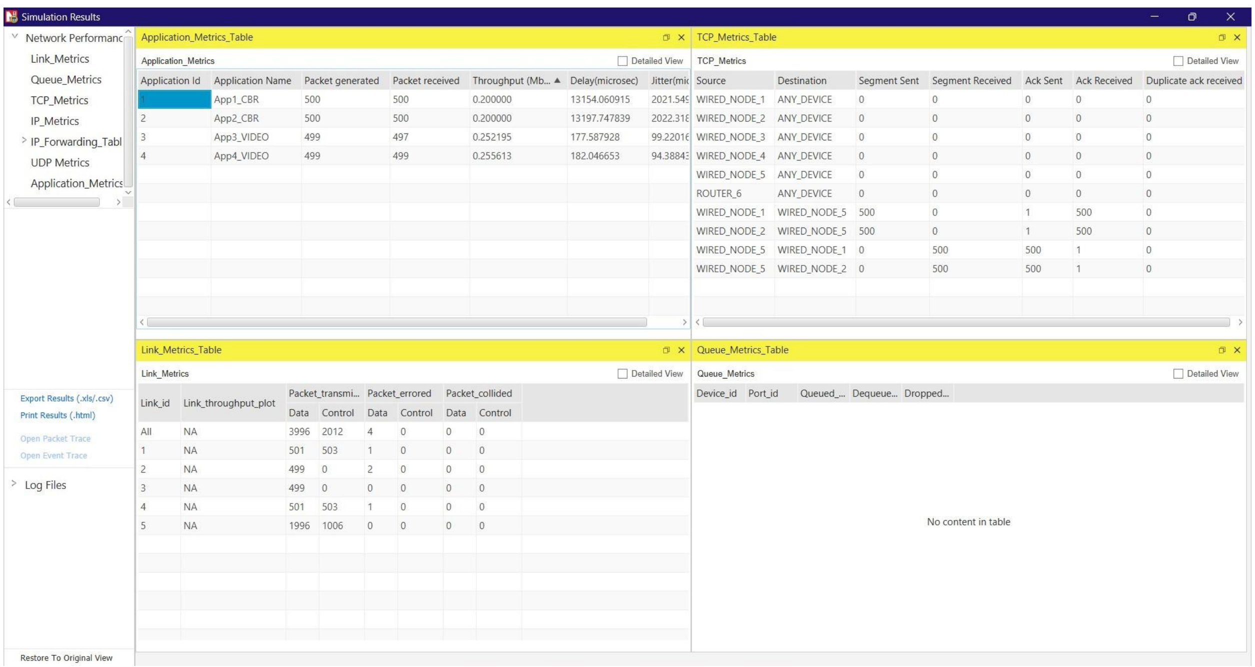
3.1 Experiment

1. Configure a new network as shown in the figure 5 with 4 wired node, 1 router.
2. Configure two CBR application between node B and F and between C and F with packet size of 500 bytes.
3. Configure two video application between node D and node F and between node E and F with Frame per second 50.
4. Keep all properties of all nodes, router and links as default values.
5. Run the simulation for 10 second.

3.2 Exercise

1. Calculate and Observe average throughput of both the applications (CBR and VIDEO).

Here are the notations, 1=A; B=2; C=3; D=4; E=5; 6=A.


Simulation Results

Application_Metrics_Table

Application Id	Application Name	Packet generated	Packet received	Throughput (Mb...	Delay(microsec)	Jitter(mic
1	App1_CBR	500	500	0.200000	13154.060915	2021.549
2	App2_CBR	500	500	0.200000	13197.747839	2022.318
3	App3_VIDEO	499	497	0.252195	177.587928	99.22016
4	App4_VIDEO	499	499	0.255613	182.046653	94.38843

TCP_Metrics_Table

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
WIRED_NODE_1	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_3	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_4	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	ANY_DEVICE	0	0	0	0	0
ROUTER_6	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_1	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_2	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_5	WIRED_NODE_1	0	500	500	1	0
WIRED_NODE_5	WIRED_NODE_2	0	500	500	1	0

Link_Metrics_Table

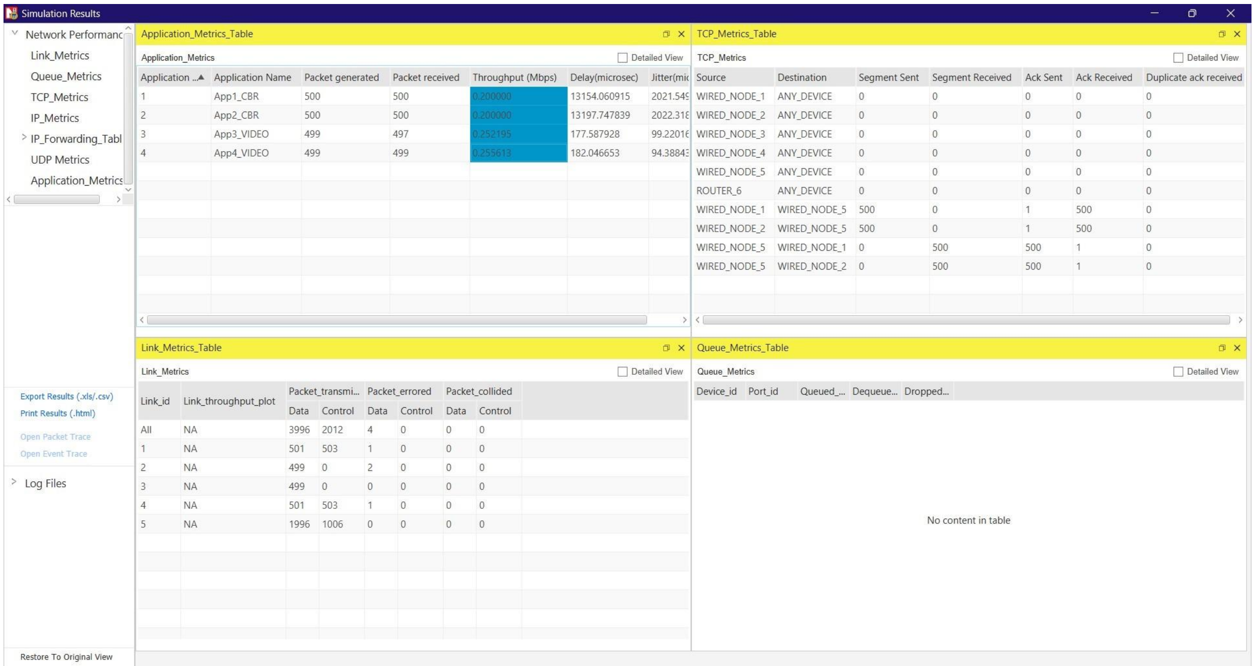
Link_id	Link_throughput_plot	Packet_transmi...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	3996	2012	4	0	0	0
1	NA	501	503	1	0	0	0
2	NA	499	0	2	0	0	0
3	NA	499	0	0	0	0	0
4	NA	501	503	1	0	0	0
5	NA	1996	1006	0	0	0	0

Queue_Metrics_Table

Device_id	Port_id	Queued_...	Dequeue...	Dropped...

No content in table

2. Observe the delay and throughput metrics in the simulation window and write down your observation.

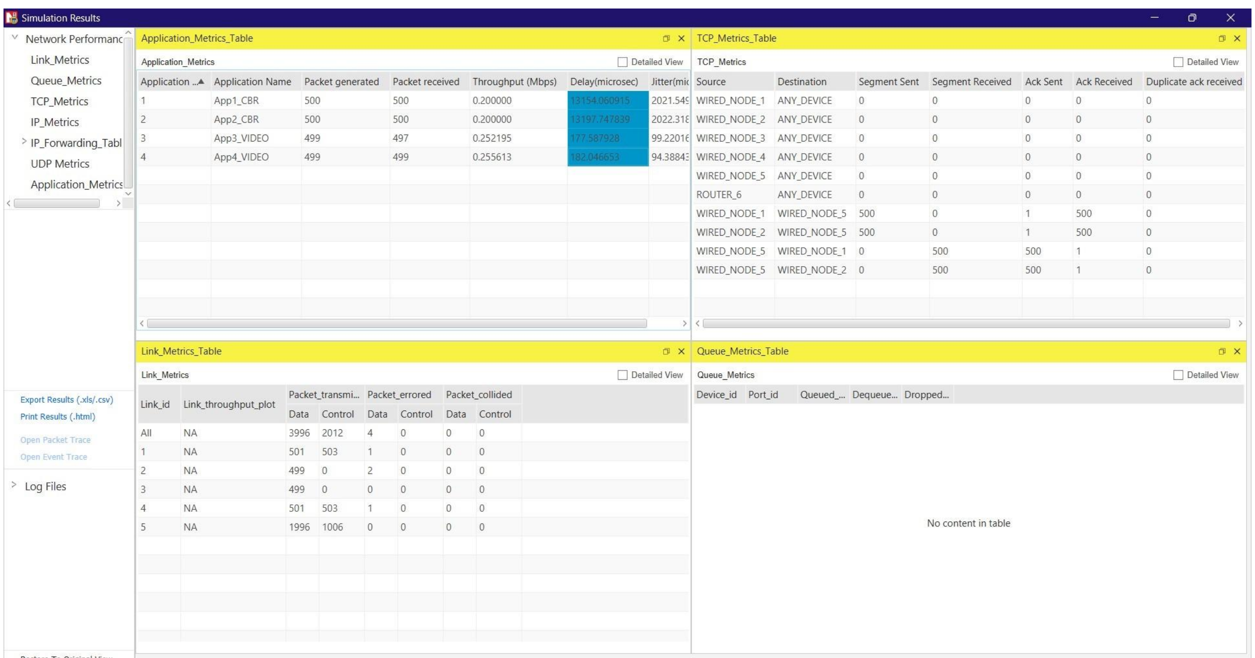


Application ...	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(mic
1	App1_CBR	500	500	0.200000	13154.060915	2021.549
2	App2_CBR	500	500	0.200000	13197.747839	2022.318
3	App3_VIDEO	499	497	0.252195	177.587928	99.22016
4	App4_VIDEO	499	499	0.255613	182.046653	94.38843

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
WIRED_NODE_1	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_3	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_4	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	ANY_DEVICE	0	0	0	0	0
ROUTER_6	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_1	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_2	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_5	WIRED_NODE_1	0	500	500	1	0
WIRED_NODE_5	WIRED_NODE_2	0	500	500	1	0

Link_id	Link_throughput_plot	Packet_transmi...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	3996	2012	4	0	0	0
1	NA	501	503	1	0	0	0
2	NA	499	0	2	0	0	0
3	NA	499	0	0	0	0	0
4	NA	501	503	1	0	0	0
5	NA	1996	1006	0	0	0	0

Device_id	Port_id	Queued_...	Dequeue...	Dropped...



Application ...	Application Name	Packet generated	Packet received	Throughput (Mbps)	Delay(microsec)	Jitter(mic
1	App1_CBR	500	500	0.200000	13154.060915	2021.549
2	App2_CBR	500	500	0.200000	13197.747839	2022.318
3	App3_VIDEO	499	497	0.252195	177.587928	99.22016
4	App4_VIDEO	499	499	0.255613	182.046653	94.38843

Source	Destination	Segment Sent	Segment Received	Ack Sent	Ack Received	Duplicate ack received
WIRED_NODE_1	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_2	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_3	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_4	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_5	ANY_DEVICE	0	0	0	0	0
ROUTER_6	ANY_DEVICE	0	0	0	0	0
WIRED_NODE_1	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_2	WIRED_NODE_5	500	0	1	500	0
WIRED_NODE_5	WIRED_NODE_1	0	500	500	1	0
WIRED_NODE_5	WIRED_NODE_2	0	500	500	1	0

Link_id	Link_throughput_plot	Packet_transmi...		Packet_errored		Packet_collided	
		Data	Control	Data	Control	Data	Control
All	NA	3996	2012	4	0	0	0
1	NA	501	503	1	0	0	0
2	NA	499	0	2	0	0	0
3	NA	499	0	0	0	0	0
4	NA	501	503	1	0	0	0
5	NA	1996	1006	0	0	0	0

Device_id	Port_id	Queued_...	Dequeue...	Dropped...

(4) Analysing RTT Of TCP Using Wireshark.

4.1 Exercise:
Answer the following questions, by opening the Wireshark captured packet file tcp-ethereal-trace-1 in http://gaia.cs.umass.edu/wireshark-labs/wireshark-traces.zip
(Once you have downloaded the trace, you can load it into Wireshark and view the trace using the File pull down menu, choosing Open, and then selecting the tcp-ethereal-trace-1 trace file.)

4.2 Questions:
1. Consider the TCP segment containing the HTTP POST as the first segment in the TCP connection. What are the sequence numbers of the first six segments in the TCP connection (including the segment containing the HTTP POST)? At what time was each segment sent? When was the ACK for each segment received? Given the difference between when each TCP segment was sent, and when its acknowledgement was received, what is the RTT value for each of the six segments? What is the EstimatedRTT value after the receipt of each ACK?
Note: Wireshark has a nice feature that allows you to plot the RTT for each of the TCP segments sent. Select a TCP segment in the listing of captured packets window that is being sent from the client to the gaia.cs.umass.edu server. Then select: Statistics− >TCP Stream Graph− >Round Trip Time Graph.

Ans:
Sequence Numbers Of First Six Segments:

1. Seq 1 - Sent At 0.023265; ACK Received At 0.053937.
2. Seq 1 - Sent At 0.026477; ACK Received At 0.077294.
3. Seq 1 - Sent At 0.041737; ACK Received At 0.124085.
4. Seq 1 - Sent At 0.054026; ACK Received At 0.169118.
5. Seq 1 - Sent At 0.054690; ACK Received At 0.217299.
6. Seq 1 - Sent At 0.077405; ACK Received At 0.267802.

2. What is the length of each of the first six TCP segments?

Ans:

1st TCP Segment - 565 bytes.

2nd TCP Segment - 1460 bytes.

3rd TCP Segment - 1460 bytes.

4th TCP Segment - 1460 bytes.

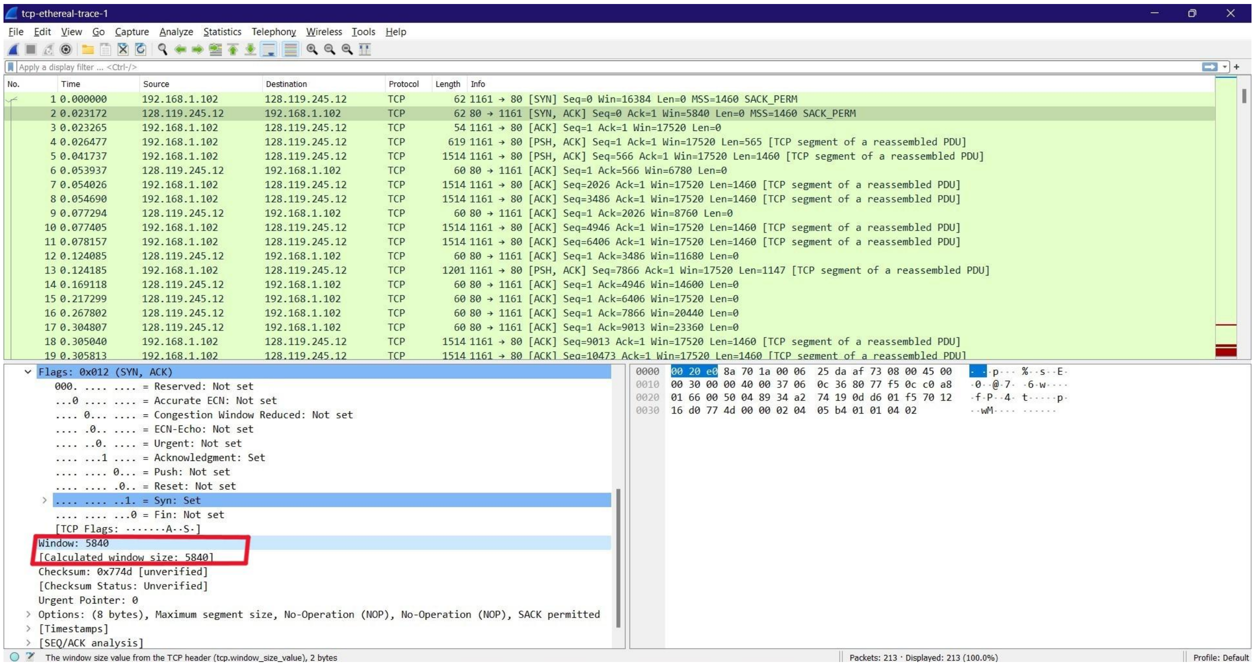
5th TCP Segment - 1460 bytes.

6th TCP Segment - 1460 bytes.

3. What is the minimum amount of available buffer space advertised at the received for the entire trace? Does the lack of receiver buffer space ever throttle the sender?

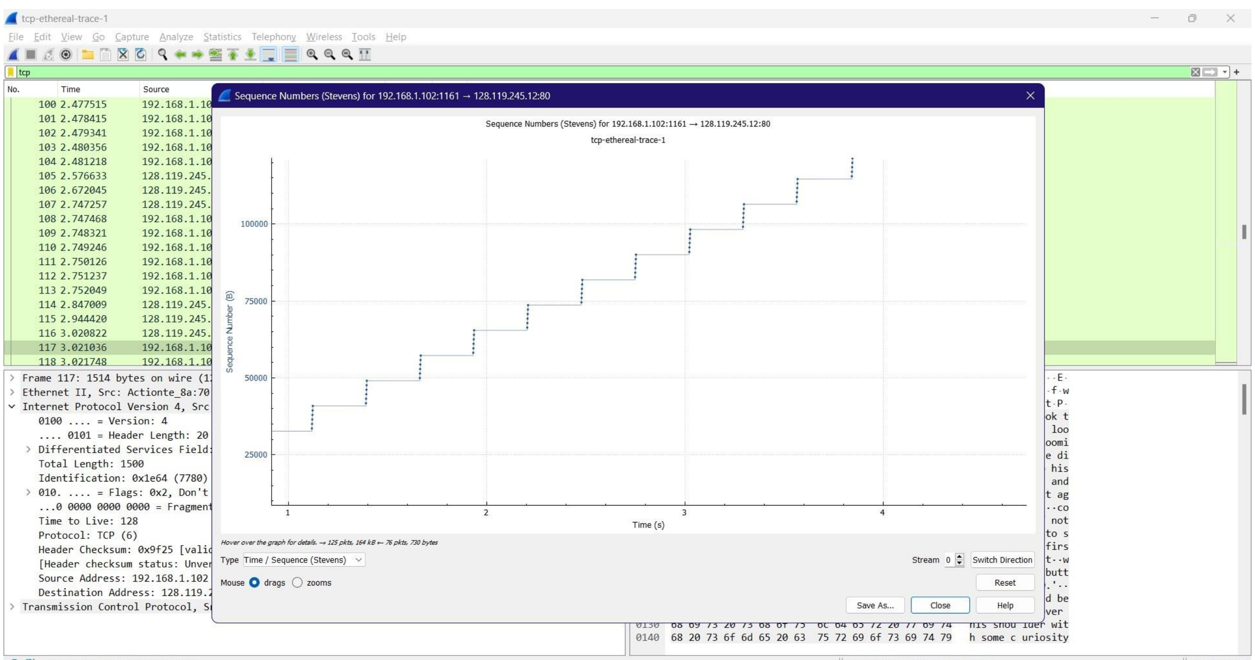
Ans: The minimum amount of available buffer space advertised at the received for the entire trace indicated first ACK from the server, its value is 5840 bytes.

The trace file contains no retransmitted segments. We can check the sequence numbers of the TCP segments in the trace file to confirm this. All sequence numbers from the source to the destination increase monotonically with respect to time in this trace's TimeSequence-Graph (Stevens).



4. Are there any retransmitted segments in the trace file? What did you check for (in the trace) in order to answer this question?

Ans:

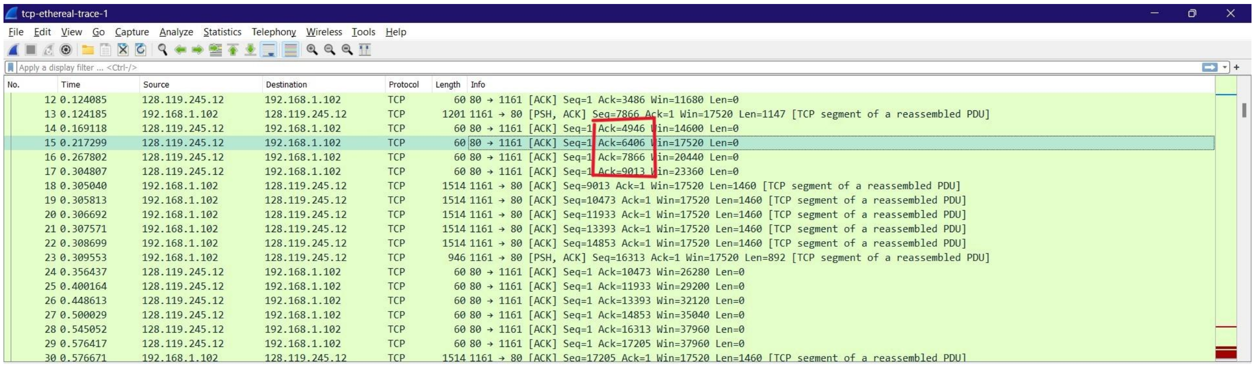


There are no retransmitted segments in the trace file because all sequence numbers in the time sequence graph (Stevens) are monotonically increasing.

5. How much data does the receiver typically acknowledge in an ACK? Can you identify cases where the receiver is ACKing every other received segment.

Ans:

The difference between two consecutive acknowledged sequence numbers indicates the data received by the server between these two ACKs.



No.	Time	Source	Destination	Protocol	Length	Info
12	0.124085	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=3486 Win=11680 Len=0
13	0.124185	192.168.1.102	128.119.245.12	TCP	1201	1161 → 80 [PSH, ACK] Seq=7866 Ack=1 Win=17520 Len=1147 [TCP segment of a reassembled PDU]
14	0.169118	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=4946 Win=14600 Len=0
15	0.217299	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=6406 Win=17520 Len=0
16	0.267802	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=7866 Win=20440 Len=0
17	0.304807	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=9013 Win=23360 Len=0
18	0.305040	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=9013 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
19	0.305813	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=10473 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
20	0.306692	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=11933 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
21	0.307571	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=13393 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
22	0.308699	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=14853 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]
23	0.309553	192.168.1.102	128.119.245.12	TCP	946	1161 → 80 [PSH, ACK] Seq=16313 Ack=1 Win=17520 Len=892 [TCP segment of a reassembled PDU]
24	0.356437	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=10473 Win=26280 Len=0
25	0.400164	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=11933 Win=29200 Len=0
26	0.448613	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=13393 Win=32120 Len=0
27	0.500029	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=14853 Win=35040 Len=0
28	0.545052	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=16313 Win=37960 Len=0
29	0.576417	128.119.245.12	192.168.1.102	TCP	60	80 → 1161 [ACK] Seq=1 Ack=17205 Win=37960 Len=0
30	0.576671	192.168.1.102	128.119.245.12	TCP	1514	1161 → 80 [ACK] Seq=17205 Ack=1 Win=17520 Len=1460 [TCP segment of a reassembled PDU]

6. What is the throughput (bytes transferred per unit time) for the TCP connection? Explain how you calculated this value.

Ans:

According to the packet's acknowledgement number, 6406 bytes were acknowledged, as can be seen when looking at it. This message was sent at 0.217299. A rough estimate of the average throughput is 6406 bytes/0.217299 seconds, or 29,480 bytes/seconds.

(5) Analysing UDP Protocol Using Wireshark

5.1 Exercise

1. Start a Wireshark capture.
2. Open a command prompt.
3. Type ipconfig /flushdns and press Enter to clear your DNS name cache.
4. Type nslookup 8.8.8.8 and press Enter to look up the hostname for IP address 8.8.8.8.

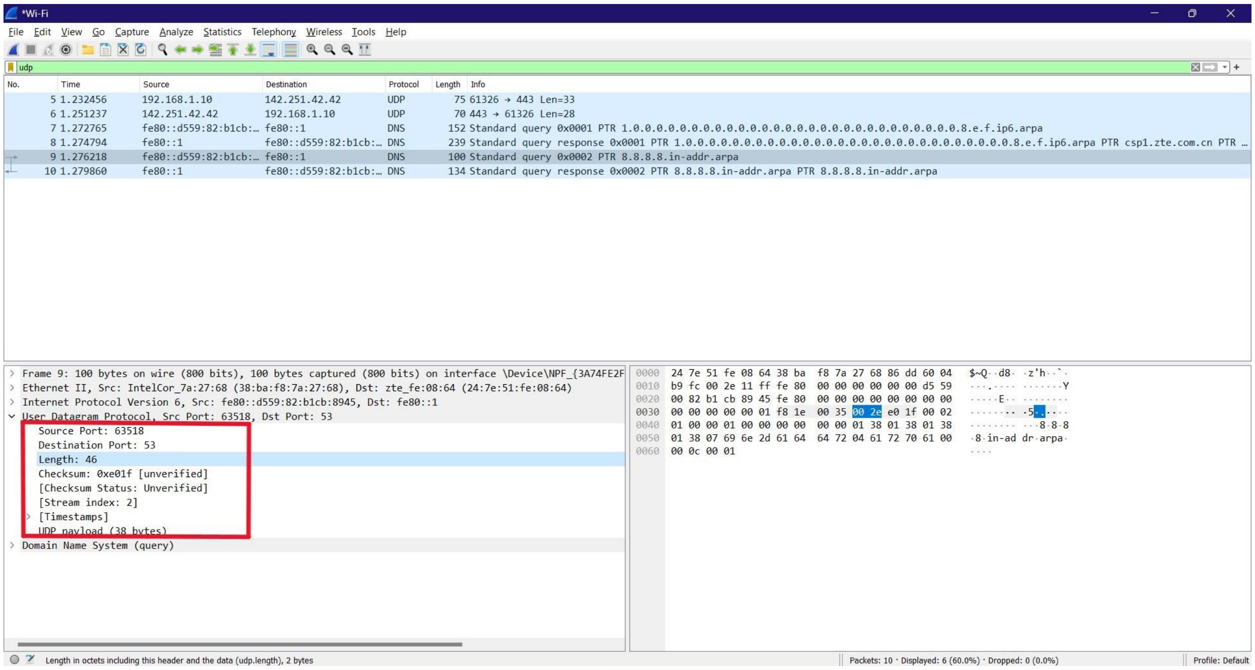
```
C:\Users\saifs>ipconfig /flushdns

Windows IP Configuration

Successfully flushed the DNS Resolver Cache.

C:\Users\saifs>nslookup 8.8.8.8
Server:  csp1.zte.com.cn
Address:  fe80::1

Name:    dns.google
Address:  8.8.8.8
```

5. Close the command prompt.
6. Stop the Wireshark capture.

5.2 Questions

1. Select one UDP packet from your trace. From this packet, determine how many fields there are in the UDP header. (You shouldnt look in the textbook! Answer these questions directly from what you observe in the packet trace.) Name these fields.

Ans: UDP Header contains 4 fields. The six fields are Source Port, Destination Port, Length & Checksum.



2. By consulting the displayed information in Wiresharks packet content field for this packet, determine the length (in bytes) of the UDP header fields.

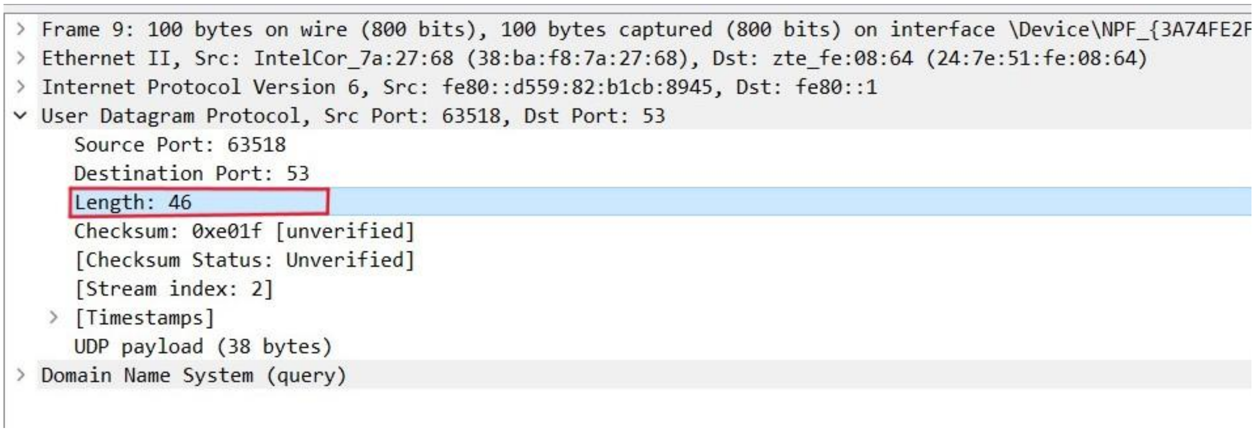
Ans: Length Of UDP Header: 8 bytes.

Header length = Length field bytes - UDP payload bytes = 46 - 38 = 8 bytes.

3. The value in the Length field is the length of what? (You can consult the text for this answer). Verify your claim with your captured UDP packet.

Ans: Value in the Length field is the length of the UDP segment (header + data).

Here the length is 46 bytes which is 8 header bytes + 38 UDP payload bytes.



4. What is the maximum number of bytes that can be included in a UDP payload? (Hint: the answer to this question can be determined by your answer to 2. above)

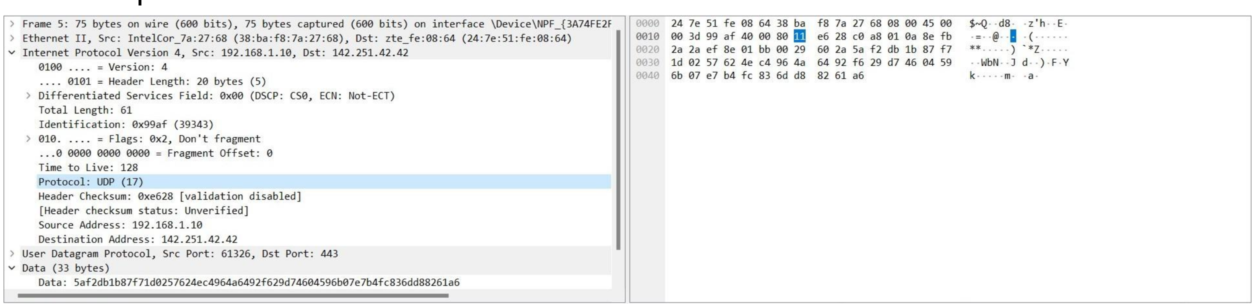
Ans: A UDP payload can contain a maximum of (2^16 - 1) bytes plus the header bytes. This results in 65535 bytes - 8 bytes = 65527 bytes.

5. What is the largest possible source port number? (Hint: see the hint in 4.)

Ans: The largest possible Source Port Number is (2^16 – 1) = 65535.

6. What is the protocol number for UDP? Give your answer in both hexadecimal and decimal notation. To answer this question, youll need to look into the Protocol field of the IP datagram containing this UDP segment.

Ans: The protocol number for UDP is 0x11 hex and it is 17 in decimal value.



7. Why we have used DNS commands to capture UDP packets? Do you know any-other method to generate UDP traffic using wireshark? Write your answer in detail.

Ans: We use DNS commands to capture UDP packets because DNS uses UDP as the transport protocol for the majority of its queries and responses.

Other ways to generate UDP traffic are:
-Using a basic UDP client-server application: We can write a basic UDP client-server application that sends and receives UDP packets over a specific port. Wireshark can then be used to capture the traffic and analyze the packets.