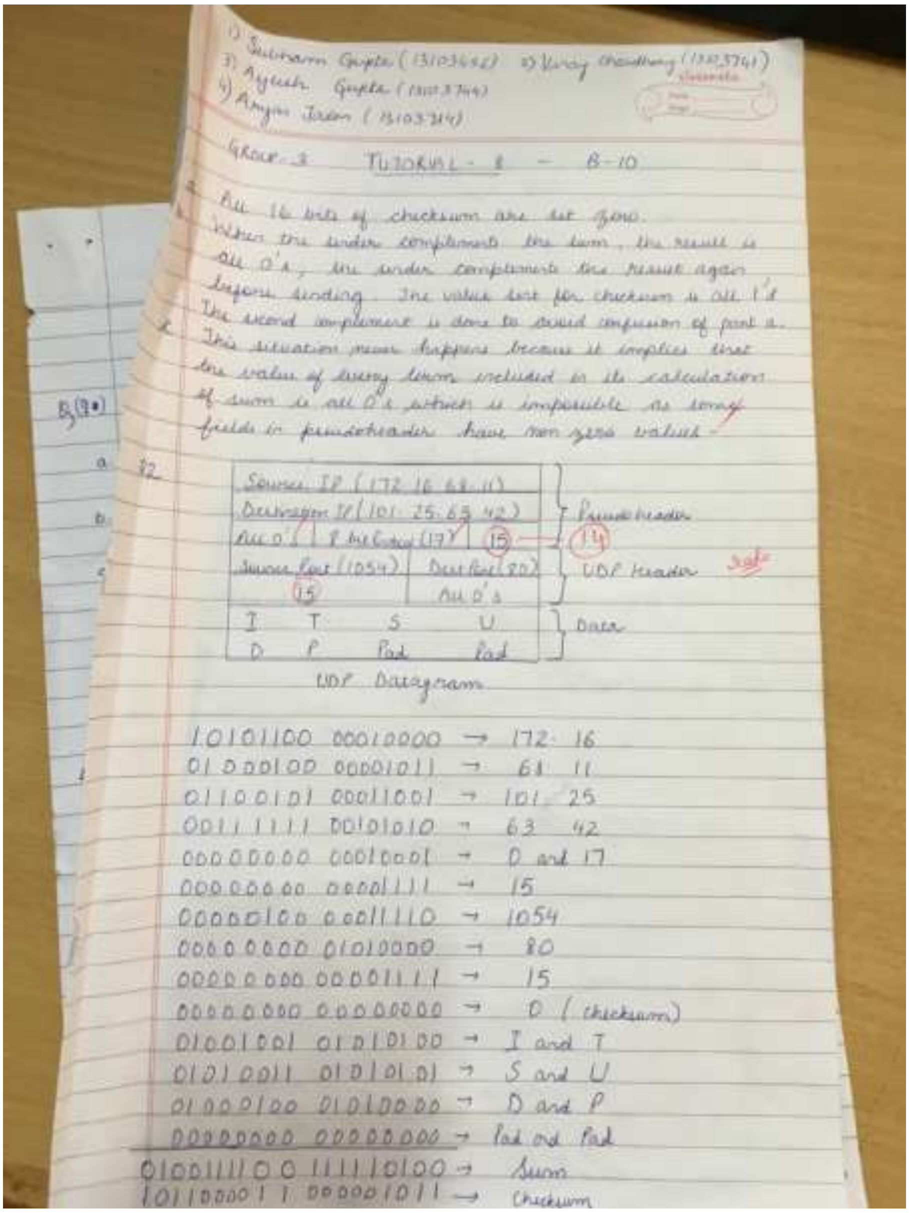


1) Shivam Gupta (13103652) 2) Vinay Chaudhary (13103741)
3) Ayush Gupta (13103744)
4) Anyes Jain (13103714)

GROUP - 3 TUTORIAL - 8 - B-10

a. All 16 bits of checksum are set zero.

b. When the sender complements the sum, the result is all 0's, the sender complements the result again before sending. The value sent for checksum is all 1's. The second complement is done to avoid confusion of part a.

c. This situation never happens because it implies that the value of every term included in the calculation of sum is all 0's which is impossible as some fields in pseudoheader have non zero values.

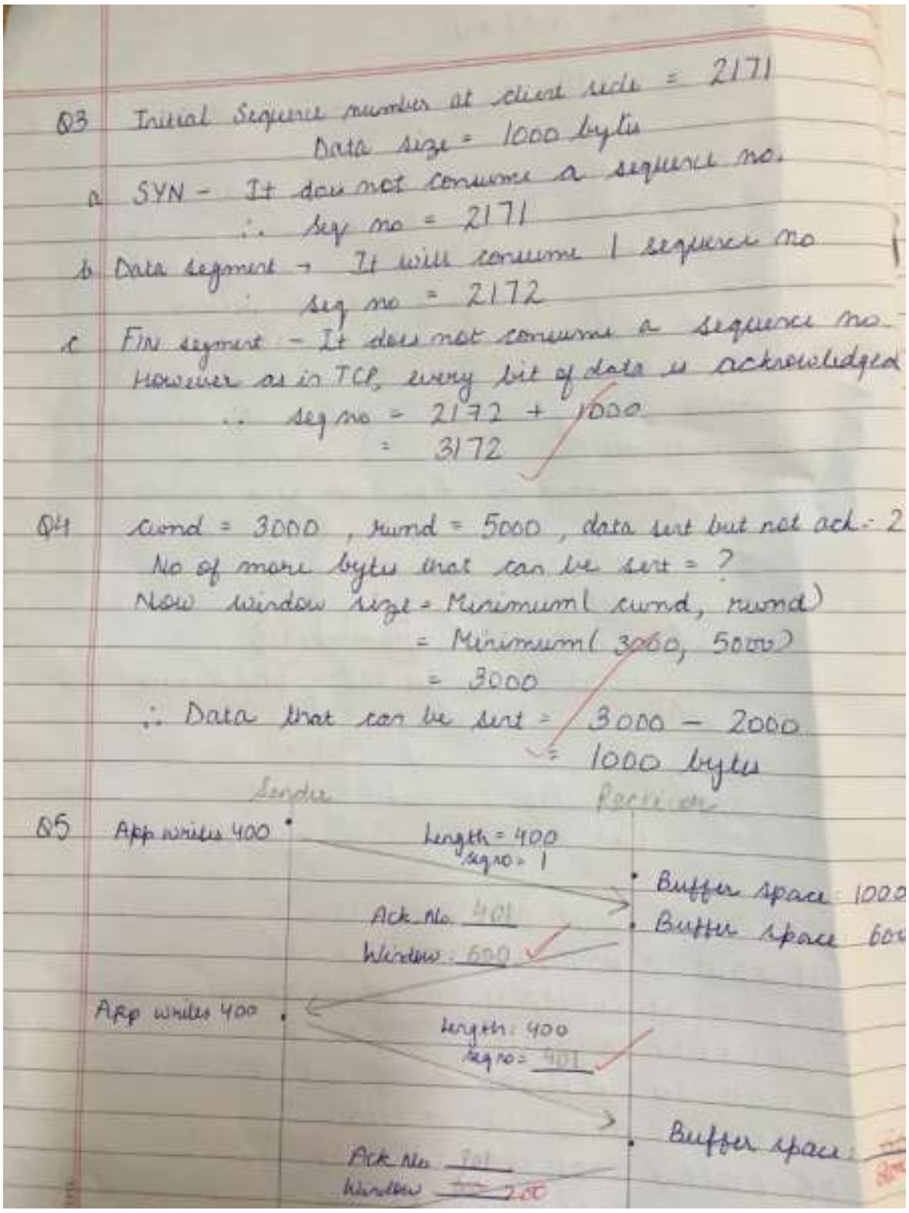
12.

Source IP (172.16.68.11)			Pseudoheader
Destination IP (101.25.63.42)			
All 0's	8 bit Protocol (17)	15	14
Source Port (1054)		Dest Port (80)	UDP Header
15		All 0's	
I T		S U	Data
D P		Pad Pad	

UDP Datagram

3.5/5

10101100 00010000 → 172 16
01000100 00001011 → 68 11
01100101 00011001 → 101 25
00111111 00101010 → 63 42
00000000 00010001 → 0 and 17
00000000 00001111 → 15
00000100 00011110 → 1054
00000000 01010000 → 80
00000000 00001111 → 15
00000000 00000000 → 0 (checksum)
01001001 01010100 → I and T
01010011 01010101 → S and U
01000100 01010000 → D and P
00000000 00000000 → Pad and Pad
0100111100 11110100 → Sum
1011000011 00001011 → Checksum



Q3 Initial Sequence number at client side = 2171
Data size = 1000 bytes

a SYN - It does not consume a sequence no.
∴ seq no = 2171

b Data segment → It will consume 1 sequence no
seq no = 2172

c FIN segment - It does not consume a sequence no.
However as in TCP, every bit of data is acknowledged
∴ seq no = 2172 + 1000
= 3172

Q4 cwnd = 3000, rwnd = 5000, data sent but not ack - 2
No of more bytes that can be sent = ?
Now window size = Minimum(cwnd, rwnd)
= Minimum(3000, 5000)
= 3000
∴ Data that can be sent = 3000 - 2000
= 1000 bytes

Q5
Sender
Receiver

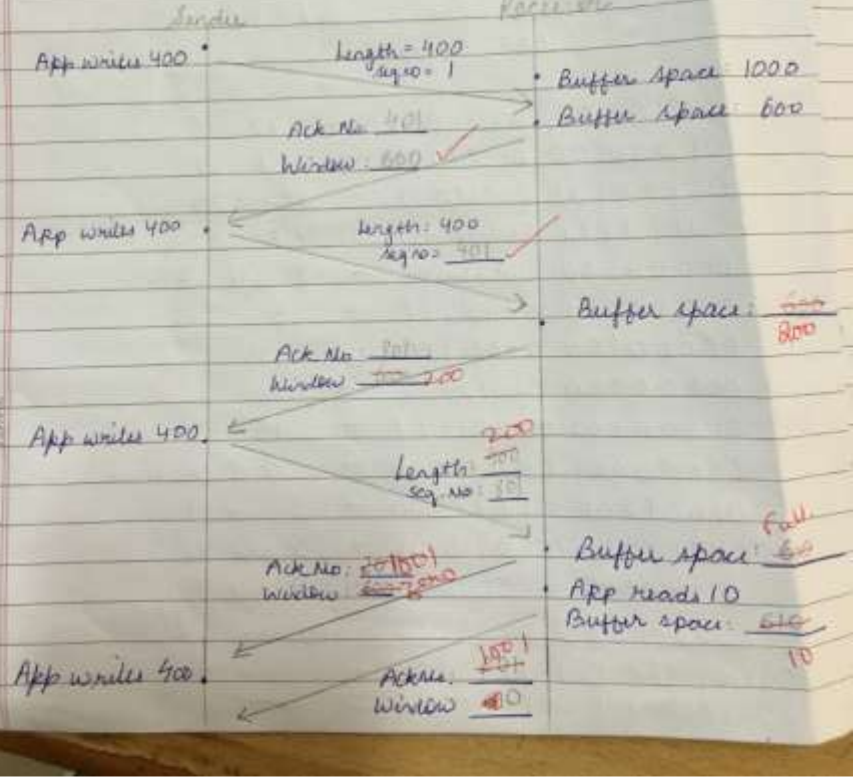
App writes 400
Length = 400
seq no = 1
Buffer space 1000
Ack No 401
Window: 600
Buffer space 600

App writes 400
Length: 400
seq no = 401
Buffer space: ~~[illegible]~~ 200
Ack No 801
Window ~~[illegible]~~ 200

c FIN segment - It does not consu[illegible] ... is acknowledg[illegible]
However as in TCP, every bit of data
∴ seq no = 2172 + 1000
= 3172 ✓

Q4 cwnd = 3000, rwnd = 5000, data sent but not ack - 2[illegible]
No of more bytes that can be sent = ?
Now window size = Minimum(cwnd, rwnd)
= Minimum(3000, 5000)
= 3000
∴ Data that can be sent = 3000 - 2000
= 1000 bytes ✓

Q5



Q6 0045 DF00 0058 FE20

From UDP Header

a. Source Port No - (0045)H = $(69)_{10}$

b. Destination Port No - (DF00)H = $(57088)_{10}$

c. Total length of User datagram = (0058)H = $(88)_{10}$ bytes

d. Length of data = 88 - 8
= 80 bytes

e. Since the source Port 69 is a well known port, the packet is from server to client — can be unreliable (Loss)

f. Trivial File Transfer Protocol (TFTP) is the client process.

Q7 (0532 0017 00000001 00000000 5002 07FF 00000000)

Source Port | Seq No | Ack No | HLEN | Window | Checksum | Urgent Ptr

reserved 000000 000010 Flags

a. Source Port No : $(0532)_{16}$ = $(1330)_{10}$

b. Destination Port No : $(0017)_{16}$ = $(23)_{10}$

c. Sequence No : $(00000001)_{16}$ = $(1)_{10}$

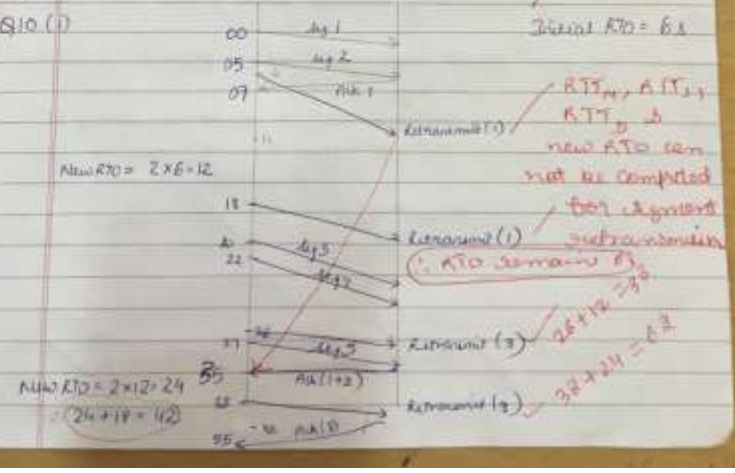
d. Acknowledgement No : $(00000000)_{16}$ = $(0)_{10}$

e. Length of header : $(5)_{16}$ = $(5)_{10}$ = 5 x 4 byte = 20 bytes

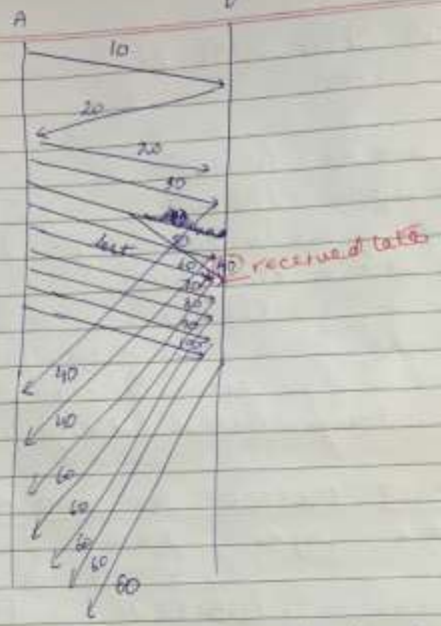
f. Segment Type : $(02)_{16}$ = SYN Flag

g. Window size : $(07FF)_{16}$ = 2047 bytes

Q10 (i)



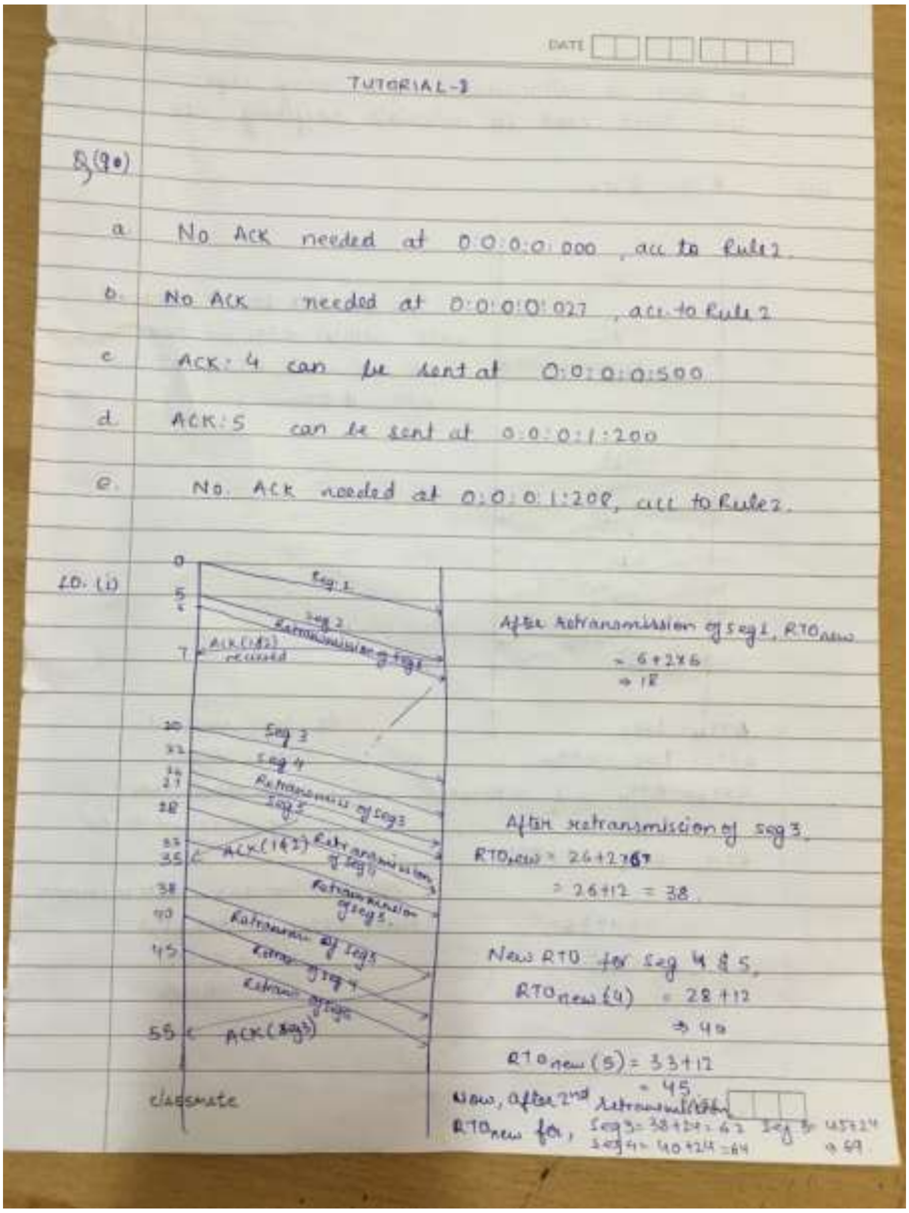
Q15



Here packet with sequence no. 40 reaches after packet
has reached. ∴ When 20 and 30 are received, ack
<u>40 is sent</u>, after that packet 50 reaches but expe
was 40. ∴ ack of <u>40 is sent</u>. Now packet with seq
40 reaches and since packet with ~~ed~~ seq 50 was pres
cumulative ack is 60. Now packet with 60 is los
∴ all succeding packets send packet with ack 6

For next packet sent by A, ack is still 60 while seq
must be 110 idealy.
If there was no more data, it would see ack of 60.
∴ retransmit 60 packet and then ack would be <u>110</u>.

43,44 45,46,47 $\boxed{48}$ 49



DATE

TUTORIAL-3

Q(9a)

a. No ACK needed at 0:0:0:0:000, acc to Rule2.

b. No ACK needed at 0:0:0:0:027, acc to Rule 2

c. ACK: 4 can be sent at 0:0:0:0:500

d. ACK: 5 can be sent at 0:0:0:1:200

e. No. ACK needed at 0:0:0:1:208, acc to Rule2.

10. (i)

0
5
6
7
20
32
34
27
28
33
35
38
40
45
55
Seg 1
Seg 2
Retransmission of seg1
ACK(1&2) received
Seg 3
Seg 4
Retransmission of seg3
Seg 5
ACK(1&2)
Retransmission of seg4
Retransmission of seg5.
Retransmission of seg3
Retrans. of seg 4
Retrans. of seg5
ACK(seg3)

After Retransmission of seg1, RTO_{new}
$= 6+2\times6$
$\Rightarrow 18$

After retransmission of seg 3.
$RTO_{new} = 26+2\times6$
$= 26+12 = 38$.

New RTO for seg 4 & 5,
$RTO_{new}(4) = 28+12$
$\Rightarrow 40$

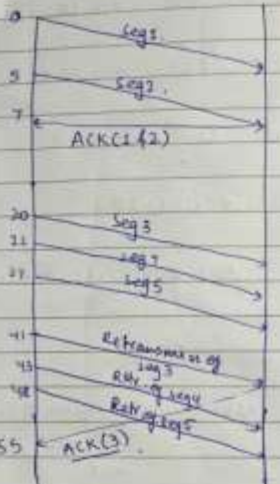
$RTO_{new}(5) = 33+12$
$= 45$

classmate

Now, after 2nd retransmission
RTO_{new} for, seg3 = 38+24 = 62 seg 5 = 45+24
seg4 = 40+24 = 64 ⇒ 69.

As, there is retransmission for every segment, we don't need to calculate anything else.

(ii) RTO = 8 sec.



As the ACK for 1st recieved well within RTO, we have to calculate RTT_M, RTT_S, RTT_D & RTO_{new}.

$RTT_M = 7 sec$

$RTT_S = 7 sec = RTT_M$

$RTT_D = \frac{RTT_M}{2} = \frac{7}{2} = 3.5 sec$

$RTO = RTT_S + 4 \times RTT_D$

$= 7 + 4 \times 3.5$

$= 21$ sec.

As, Seg3 was sent at 20, ~~then~~ ACK must be by 20+21 = 41sec

No ACK means retransmission

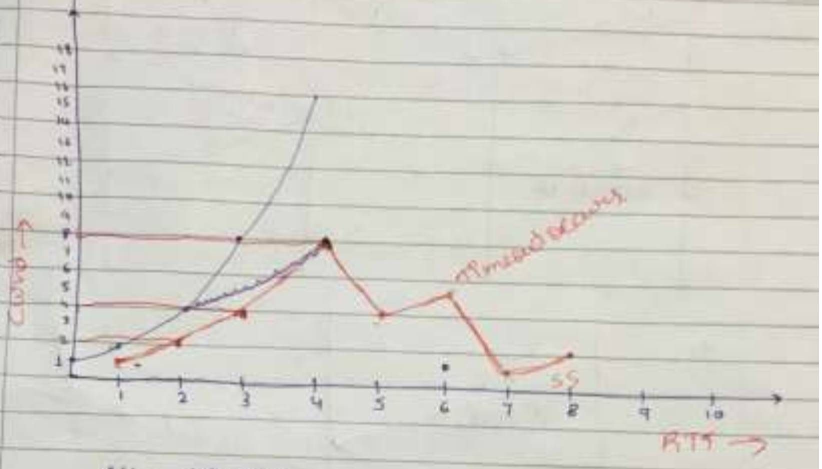
Now, $RTO_{new} = 41 + 2 \times 21$

$\Rightarrow 41 + 42$

$= 83 sec.$

DATE

(ii)



~~Ans~~ After 4th RTT,

3 duplicate ACKs are recieved

~~so~~, it creates congestion.

$cwnd = \frac{1}{2} cwnd$

$= \frac{1}{2} \times 8 = 4$

goes to AIMD

Threshold = 32

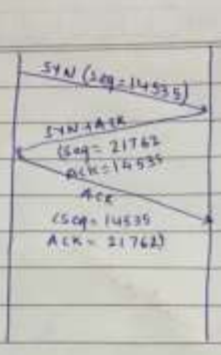
Maximum window size = 64

threshold = $\frac{64}{2}$ = 32.

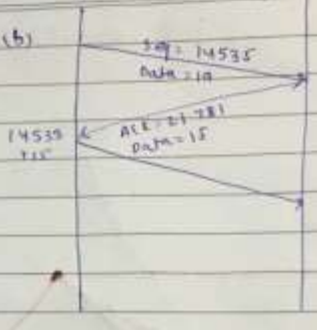
threshold

$= \frac{32}{2} =$

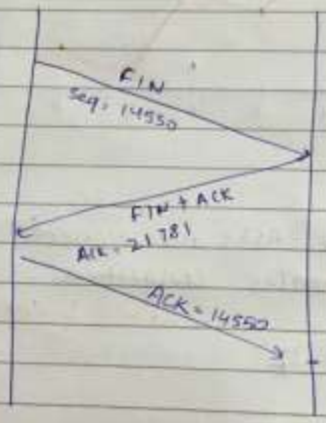
(8) (a)


SYN (seq=14535)
SYN+ACK
(seq= 21762
ACK=14535
ACK
(seq= 14535
ACK= 21762)

(b)


seq= 14535
Data =10
ACK=21781
Data=15
14535
+15

(c)


FIN
seq= 14550
FIN + ACK
ACK= 21781
ACK= 14550