National University
Of Computer and Emerging Sciences

Assignment-02

In partial
fulfillment of the requirements
for the course of

FA2024-CS3001

Computer Networks

By:

Maryam Masood
22i-1169
BSCS-C

Problem:01

~~$u_s = 30 Mbps$~~

$F = 15\ Gbits = 15 \times 1024\ Mbps = 15360\ Mbps$

$d_i = 2Mbps$

$N = 10, 100, 1{,}000 \Rightarrow N_1 = 10 \quad N_2 = 100 \quad N_3 = 1000$

$u_1 = 300\ Kbps$

$u_2 = 700\ Kbps$

$u_3 = 2\ Mbps = 2048\ Kbps$

upload rate $u_s = 22 + (1169 \% 15)\ Mbps = 22 + 14 = 36\ Mbps$

⇒ Client Server:

N=10

$$D_{cs} = max \left\{ \frac{NF}{us}, \frac{F}{d_{min}} \right\}$$

$$\frac{NF}{u_s} = \frac{10 \times 15360\ Mbps}{36\ Mbps} = 4266.667$$

$$\frac{F}{d_{min}} = \frac{15360}{2} = 7680\ sec$$

$$D_{cs} = max\{4266.667, 7680\} = 7680\ sec$$

N=100

$$\frac{NF}{u_s} = \frac{100 \times 15360}{36} = 42666.667$$

$$D_{cs} = max(42666.667, 7680) = 42666.667\ sec.$$

N=1000

$$\cancel{D_{cs} =} \frac{NF}{U_s} = \frac{1000 \times 15360}{36} = 426666.667\ sec.$$

$$D_{cs} = max(426666.667, 7680) = 426666.667\ sec.$$

⇒ Peer to Peer:

$$D_{P2P} = max \left\{ \frac{F}{U_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i} \right\}$$

N=10, U = 300 Kbps

$F/u_s = 426.667 \qquad F/d_{min} = 7680$

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{10\times15360}{36+10\times(300/1024)} = \frac{153600}{38.93} = 3945.57$$

$D_{p2p} = 7680$

N=10 u=700 kbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{10\times15360}{36+10\times(700/1024)} = \frac{153600}{42.83} = 3585.77$$

$D_{p2p} = \max\{426.6, 7680, 3585.77\} = 7680$

N=10 u=2 Mbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{10\times15360}{36+10\times2} = 2742.86$$

$D_{p2p} = 7680$

N=100 u=300 Kbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{100\times15360}{36+(100\times(300/1024))} = \frac{1536000}{65.297} = 23523.33$$

$D_{p2p} = 23523.33$

N=100 u=700 Kbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{1536000}{36+(100\times\frac{700}{1024})} = 14718.37$$

$D_{p2p} = 14718.37$

N=100 u=2 Mbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{1536000}{36+(100\times2)} = 6508.47$$

$D_{p2p} = \max\{426.667, 7680, 6508\} = 7680$

N=1000 u=300 Kbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{1000\times15360}{36+1000\times(300/1024)} = 46691.36$$

$D_{p2p} = 46691.36$

N=1000 u=700 Kbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{15360000}{36+1000\times(700/1024)} = 21345.38$$

$D_{p2p} = 21345.38$

N=1000 u=2 Mbps

$$\frac{NF}{u_s+\sum_{i=1}^{N} u_i} = \frac{15360000}{36+1000\times2} = 7544.2$$

$D_{p2p} = max\{426.6, 7680, 7544.2\} = 7680$

~~Problem:02~~

Client Server

u	N = 10	100	1000
300 Kbps	7680	42666.67	426666.67
700 Kbps	7680	42666.67	426666.67
2 Mbps	7680	42666.67	426666.67

Peer to Peer

u	N = 10	100	1000
300 Kbps	7680	23523.3	46691.36
700 Kbps	7680	14718.37	21345.38
2 Mbps	7680	7680	~~7544.2~~ 7680

Problem:02

Apache Web Server

The Apache HTTP Server is a widely used open-source web server developed by the Apache Software Foundation. It's free to use and supports various features like HTTP/HTTPS, modularity, virtual hosting, and customization for different programming languages.

Other commonly used Web Servers

1. Nginx:

- Known for high performance and scalability.

Functions:
reverse proxy, load balancing, static content serving

2. Microsoft IIS:

- Microsoft's web server for Windows.

Functions:
ASP.NET support, SSL, integration with Windows services.

3- Lite Speed.

- High-performance commercial server, Apache alternative.

Functions:

caching, DDoS protection, Apache config compatability.

4. Apache Tomcat.

- Serves Java-based web applications.

Functions:

supports Java Servlets, JSP, WebSockets.

5. Node.js:

- Javascript runtime for server-side applications.

Functions:

handles real-time, scalable web apps like chats and APIs.

Problem: 03

$$\text{Estimated RTT} = \alpha * \text{Sample RTT} + (1-\alpha) * \text{Estimated RTT}$$

$$\text{DevRTT} = \beta * |\text{SampleRTT} - \text{Estimated RTT}| + (1-\beta) * \text{DevRTT}$$

$$\text{TimeoutInterval} = \text{Estimated RTT} + 4 * \text{DevRTT}$$

⇒ After obtaining first sample RTT 106 ms,

$$\alpha = (100 + 1169 \% 32)/1000 = (100+17)/1000 = 0.117 \text{ ms}$$

$$\beta = (200 + 1169 \% 45)/1000 = (200+44)/1000 = 0.244 \text{ ms}$$

$$\text{EstimatedRTT} = (0.117)(106) + (1-0.117)(100) = 100.702 \text{ ms}$$

$$\text{DevRTT} = (0.244)|106-100| + (1-0.244)(5) = 5.244 \text{ ms}$$

$$\text{Timeout Interval} = 100 + (4 \times 5.244) = 120.976 \text{ ms}$$

⇒ After obtaining second sample RTT 120 ms

$$\text{EstimatedRTT} = (0.117)(120) + (1-0.117)(100.702) = 102.96 \text{ ms}$$

$$\text{DevRTT} = 0.244|120 - \overset{102.96}{\cancel{100.702}}| + (1-0.244)(5.244) = 8.12 \text{ ms}$$

$$\text{Timeout Interval} = 102.96 + (4 \times 8.12) = 135.44 \text{ ms}$$

⇒ After obtaining third sample RTT 140 ms

$EstimatedRTT = 0.117 \times 140 + (1-0.117)(102.96) = 107.29\ ms$

$DevRTT = (0.244)|140 - 107.29| + (1-0.244)(8.12) = 14.12\ ms$

$Timeout\ Interval = 107.29 + (4 \times 14.12) = 163.77\ ms$

⇒ After obtaining fourth sample RTT 90 ms

$EstimatedRTT = \cancel{(0.244)}\ (0.117)(90) + (1-0.117)(107.29) = 105.27\ ms$

$DevRTT = (0.244)|90 - 105.27| + (1-0.244)(14.12) = 14.4006\ ms$

$Timeout\ Interval = 105.27 + (4 \times 14.4006) = 162.8724\ ms$

⇒ After obtaining fifth sample RTT 115 ms

$EstimatedRTT = 0.117 \times 115 + (1-0.117)(105.27) = 106.41\ ms$

$DevRTT = (0.244)|115 - 106.41| + (1-0.244)(14.4006) = 12.98\ ms$

$Timeout\ Interval = 106.41 + (4 \times 12.98) = 158.33\ ms$

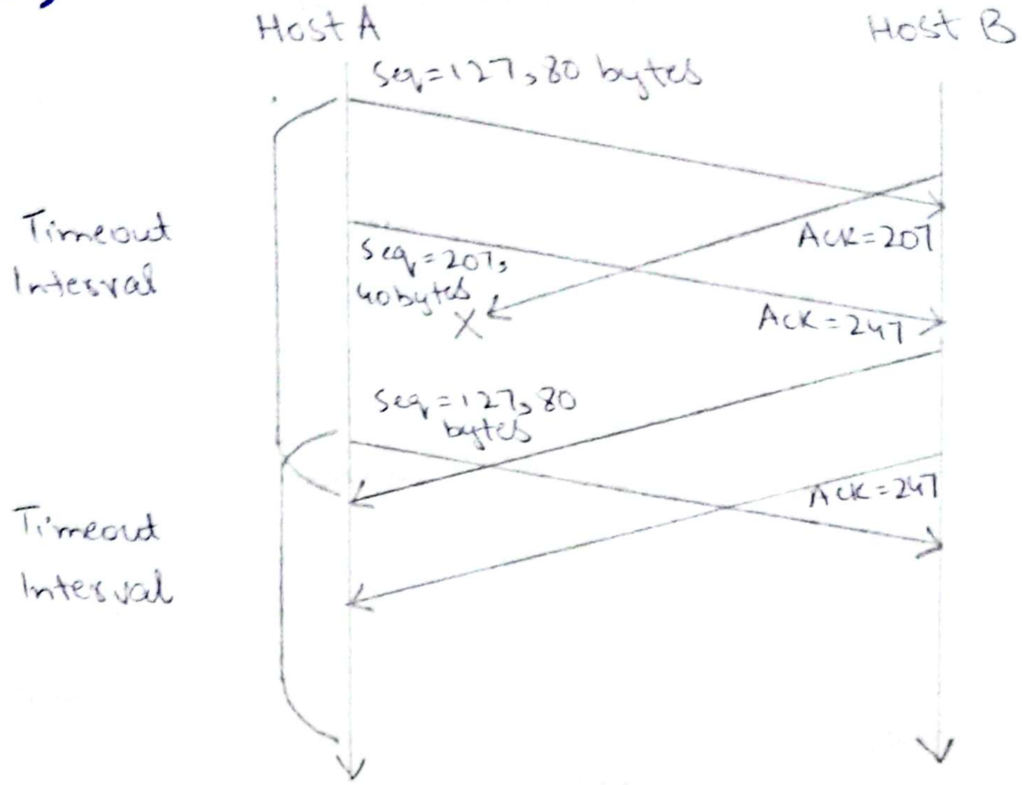
Problem: 04

(a) In the second segment from Host A to B, the sequence number is 207, source port is 302 and destination port number is 80.

(b) If the first segment arrives before the second, in the acknowledgement of the first arriving segment, the ack no. is 207, the source port no. is 80 and the destination port no. is 302.

(c) If the second segment arrives before the first segment, in the ack of the first arriving segment, the ack number is 127, indicating that it is still waiting for bytes 127 and onwards.

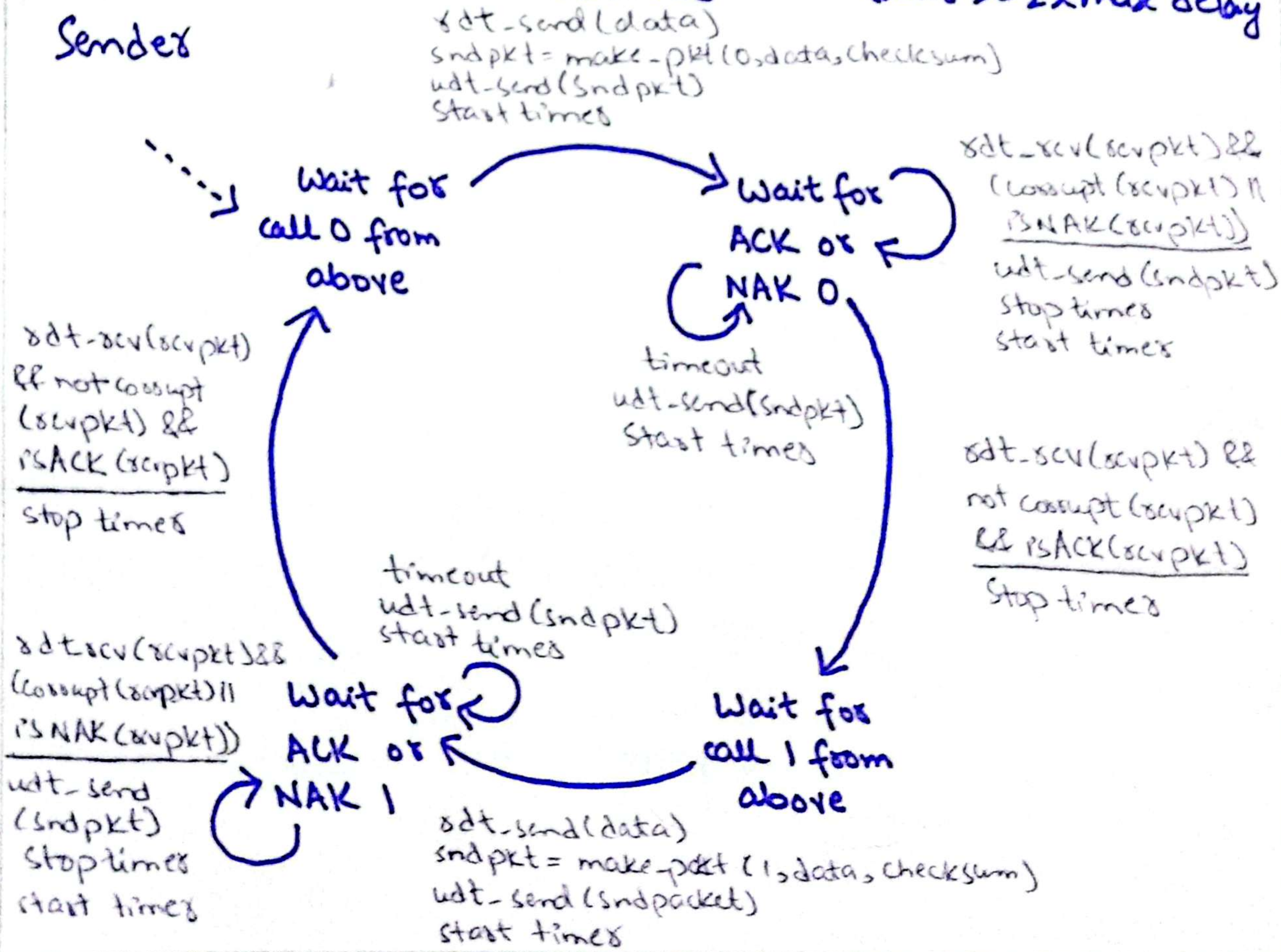
(d)



Problem : 05

timer >= 2 x max delay

Sender



Receiver:
No changes

Here, ~~we added~~ is added a timer whose value is greater than the known round-trip propagation delay. ~~We~~ A timeout event is added to the "Wait for ACK or NAK 0" and "Wait for ACK or NAK 1" states. If the timeout event occurs, the most recently transmitted packet is retransmitted. This protocol will still work with rdt 2.1 receiver

- Suppose the timeout is caused by a lost data packet i.e a packet on the sender-to-receiver channel. In this case, the receiver never received the previous transmission and, from the receiver's viewpoint, if the timeout retransmission is received, it looks exactly the same as if the original transmission is being received.
- Suppose now that an ACK is lost. The receiver will eventually retransmit the packet on a timeout. But a retransmission is exactly the same action that if an ACK is garbled. Thus the sender's reaction is the same with a loss, as with a garbled ACK. The rdt 2.1 receiver can already handle the case of a garbled ACK.
