Congestion Control

- **Congestion:** Congestion refers to a network state where, the message traffic becomes so heavy that it slows down the network response time.

- Congestion control refers to techniques and mechanisms that can: Either prevent congestion before it happens or remove congestion after it has happened
  - TCP reacts to Congestion by reducing the sender window size.
  - TCP uses a combination of GBN and SR protocols to provide reliability.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Windows in TCP

- TCP uses two windows (send window and receive window) for each direction of data transfer, i.e. four windows for a bidirectional communication.

- ***Send Window***
  - The size of the sender window is determined by the following two factors
  - **Receiver window size and Congestion window size.**

To access all paid content get KG Prime at ₹25/day CLICK HERE

- ***Receive Window***
  - Sender should not send data greater than receiver window size. Otherwise, it leads to dropping the TCP segments which causes TCP Retransmission.
  - So, sender should always send data less than or equal to receiver window size. Receiver dictates its window size to the sender through TCP Header.

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Congestion Window**
  - Sender should not send data greater than congestion window size. Otherwise, it leads to dropping the TCP segments which causes TCP Retransmission.
  - So, sender should always send data less than or equal to congestion window size.
  - Different variants of TCP use different approaches to calculate the size of congestion window. Congestion window is known only to the sender and is not sent over the links.

- **In general, Sender window size = Minimum (Receiver window size, Congestion window size)**

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

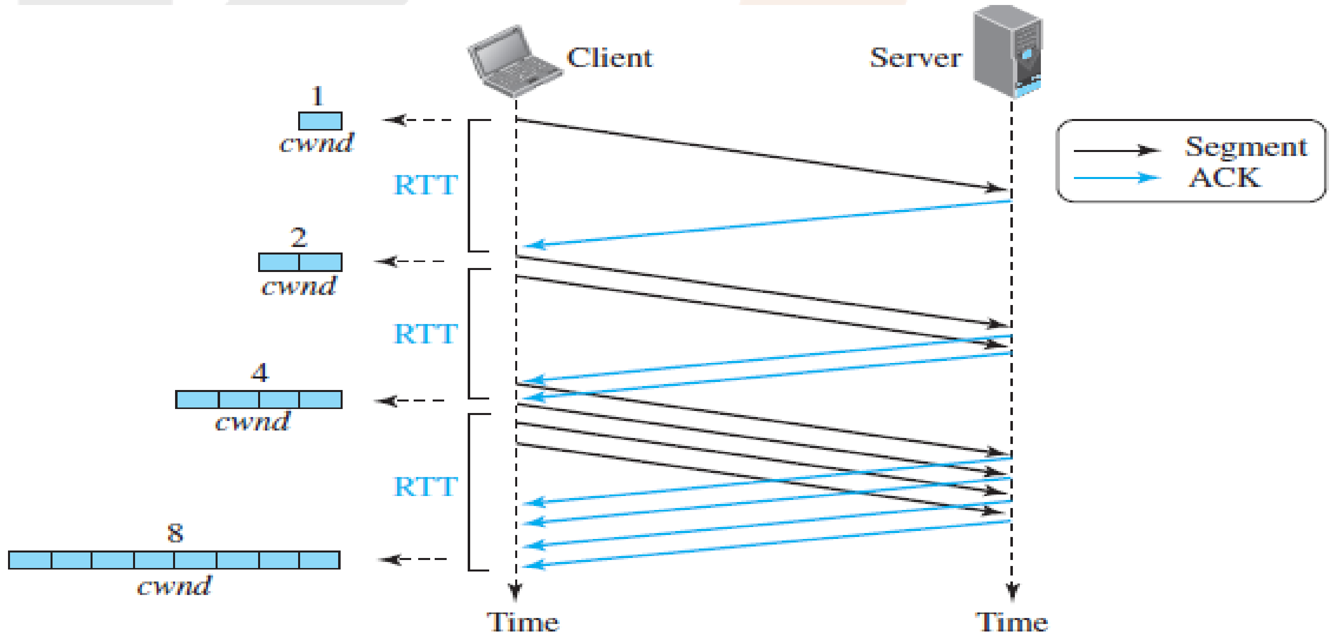
TCP Congestion Policy

- TCP’s general policy for handling congestion consists of following three phases
  - Slow Start (Exponential Increase)
  - Congestion Avoidance (Additive Increase)
  - Congestion Detection

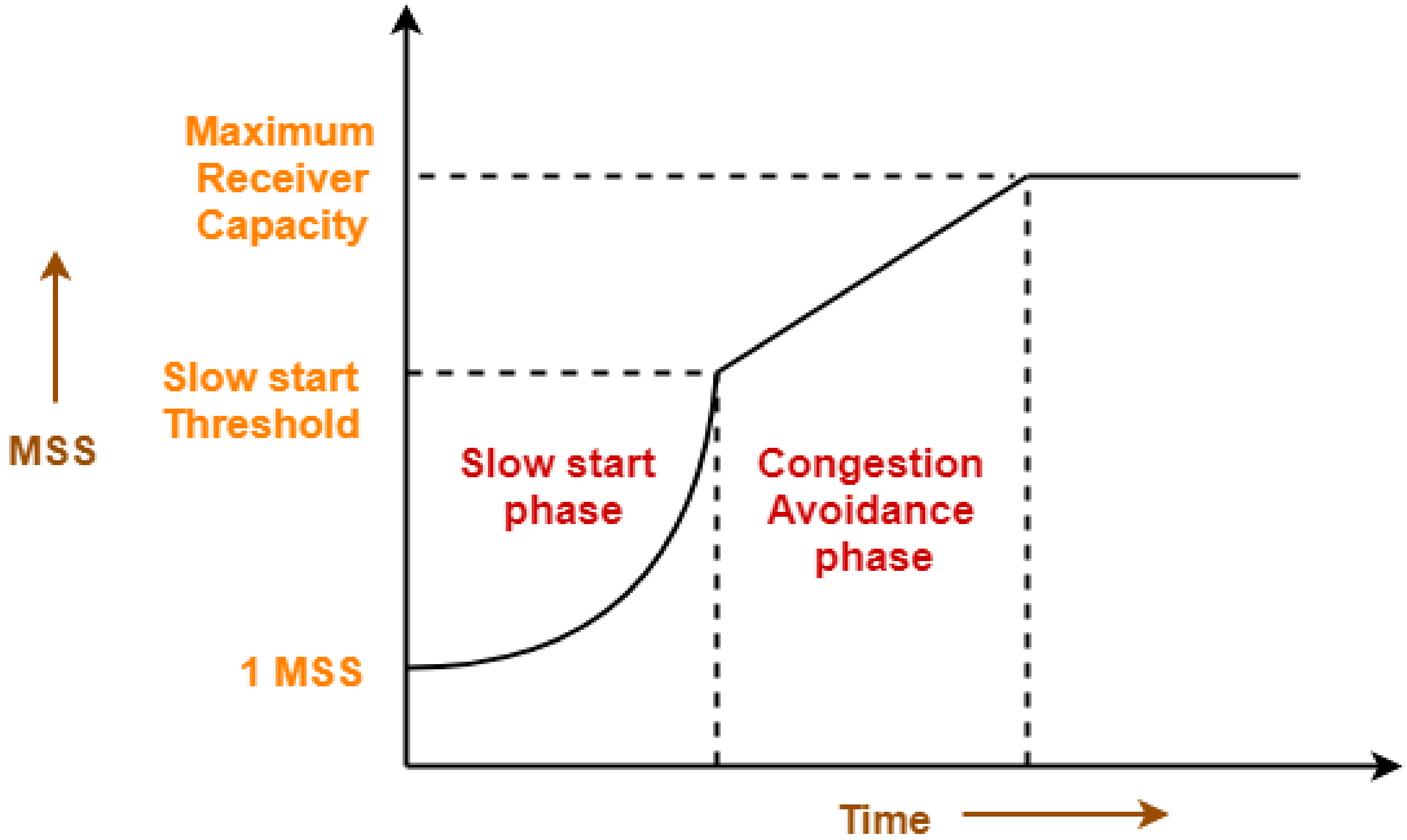
To access all paid content get KG Prime at ₹25/day CLICK HERE

Slow Start Phase (Exponential Increase)

- Initially, sender sets congestion window size = Maximum Segment Size (1 MSS).
- After receiving each acknowledgment, the size of congestion window increases exponentially.
- After 1 round trip time, congestion window size = $(2)^1$ = 2 MSS
- After 2 round trip time, congestion window size = $(2)^2$ = 4 MSS
- After 3 round trip time, congestion window size = $(2)^3$ = 8 MSS and so on.
- This phase continues until the congestion window size reaches the slow start threshold.
- **Threshold = Maximum number of TCP segments that receiver window can accommodate / 2 = (Receiver window size / Maximum Segment Size) / 2**



To access all paid content get KG Prime at ₹25/day CLICK HERE


Maximum Receiver Capacity
Slow start Threshold
MSS
1 MSS
Slow start phase
Congestion Avoidance phase
Time

ERE

Q Consider the following statements regarding the slow start phase of the TCP congestion control algorithm. Note that *cwnd* stands for the TCP congestion window and MSS denotes the Maximum Segment Size.

(i) The *cwnd* increase by 2 MSS on every successful acknowledgement.

(ii) The *cwnd* approximately doubles on every successful acknowledgement.

(iii) The *cwnd* increase by 1 MSS every round-trip time.

(iv) The *cwnd* approximately doubles every round-trip time.

Which one of the following is correct? **(GATE-2018) (1 Marks)**

(a) Only (ii) and (iii) are true

(b) Only (i) and (iii) are true

(c) Only (iv) is true

(d) Only(i) and (iv) is true

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q In the slow start phase of the TCP congestion algorithm, the size of the congestion window:

(Gate-2008) (2 Marks)

a) does not increase

b) increase linearly

c) increases quadratically

d) increases exponentially

To access all paid content get KG Prime at ₹25/day CLICK HERE

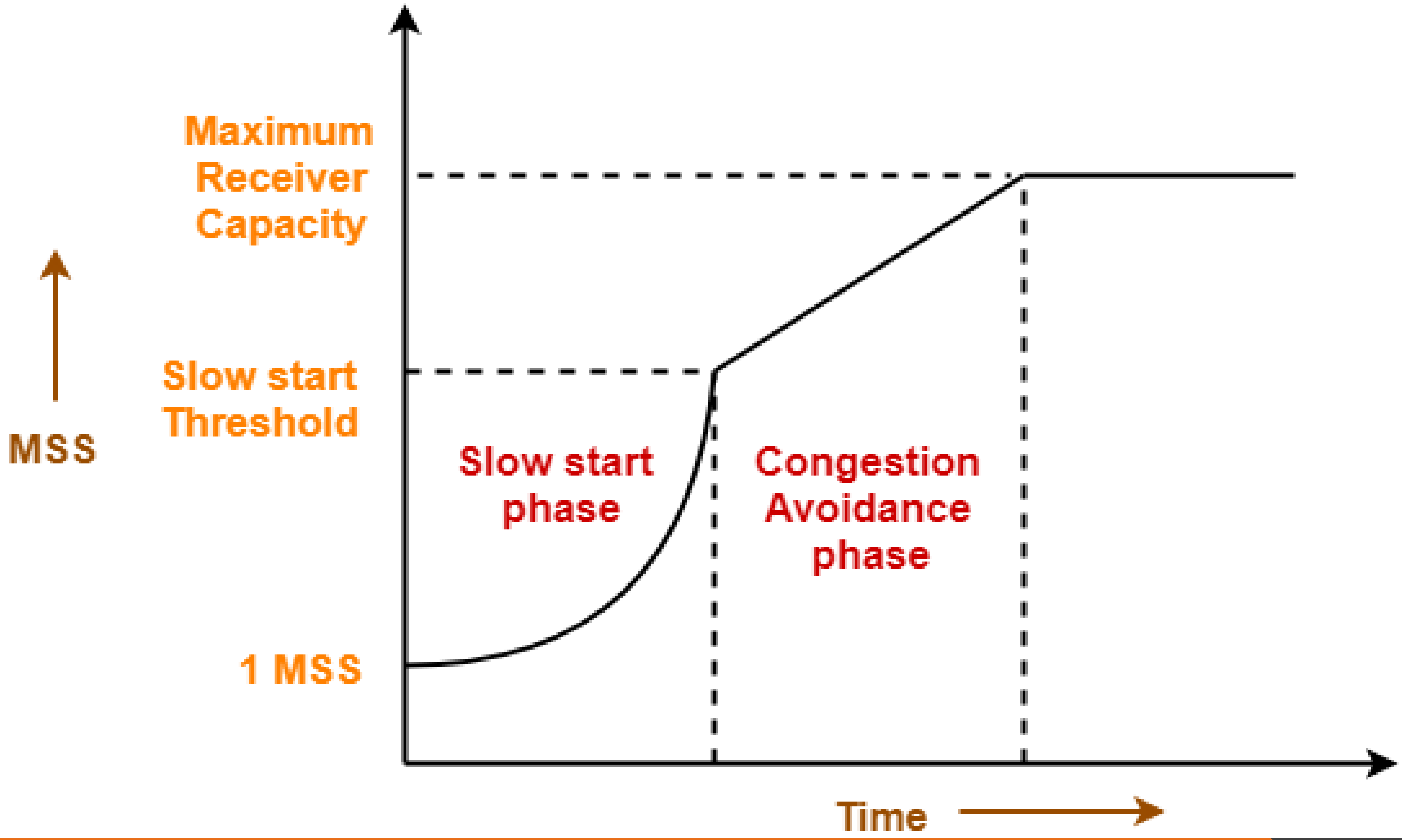
Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

Congestion Avoidance Phase

- After reaching the threshold, Sender increases the congestion window size linearly to avoid the congestion.
- On receiving each acknowledgement, sender increments the congestion window size by 1.
- **Congestion window size = Congestion window size + 1,** This phase continues until the congestion window size becomes equal to the receiver window size.

To access all paid content get KG Prime at ₹25/day CLICK HERE


Maximum Receiver Capacity
Slow start Threshold
MSS
1 MSS
Slow start phase
Congestion Avoidance phase
Time

ERE

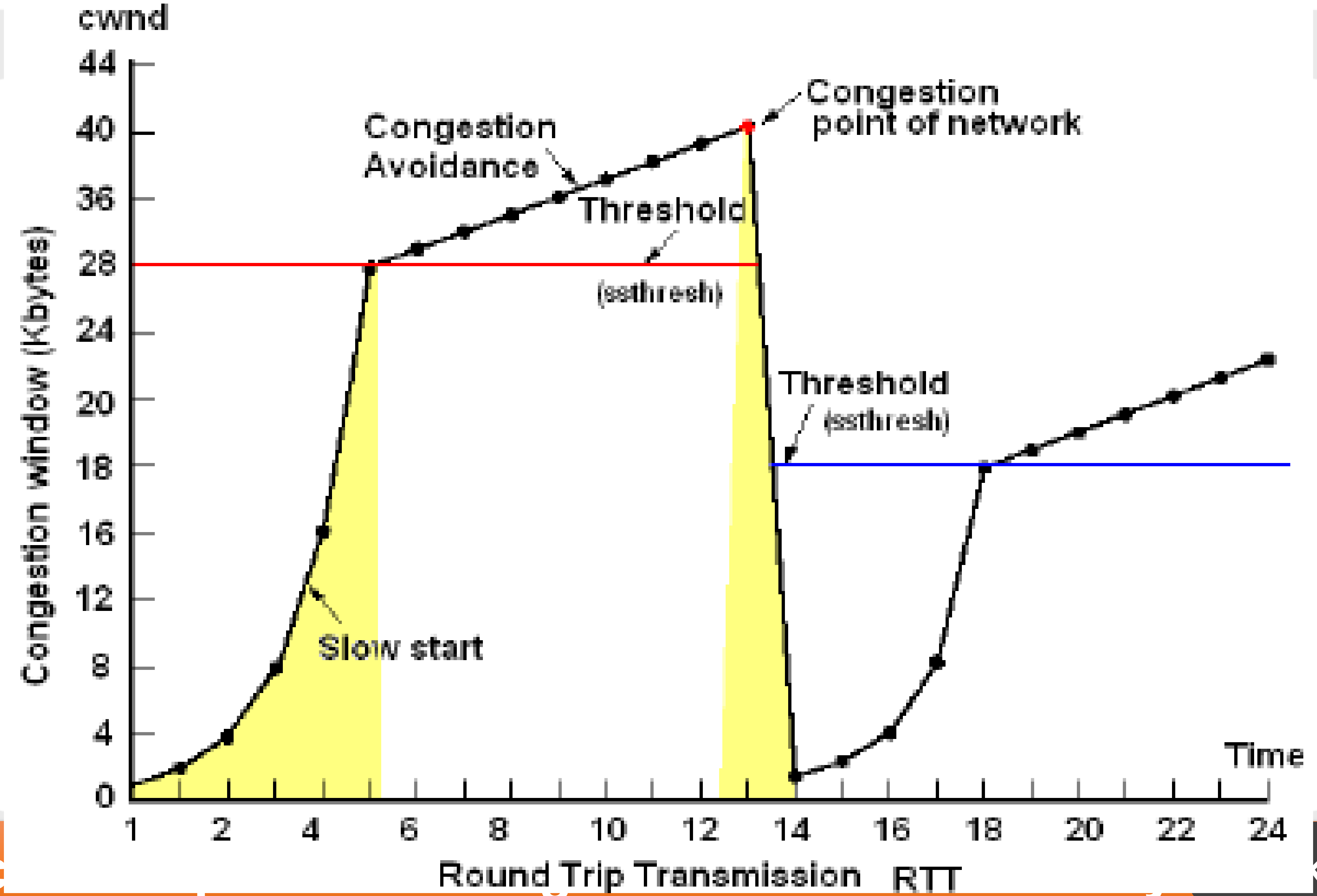
Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

<u>Congestion Detection Phase</u>

- When sender detects the loss of segments, it reacts in different ways depending on how the loss is detected
- **<u>Detection On Time Out</u>**
  - Time Out Timer expires before receiving the acknowledgement for a segment. It suggests the strong possibility of congestion in the network. There are chances that a segment has been dropped in the network.
  - **<u>Reaction:</u>** In this case, sender reacts by
    - Setting the slow start threshold to half of the current congestion window size.
    - Decreasing the congestion window size to 1 MSS.
    - Resuming the slow start phase.

To access all paid content get KG Prime at ₹25/day CLICK HERE

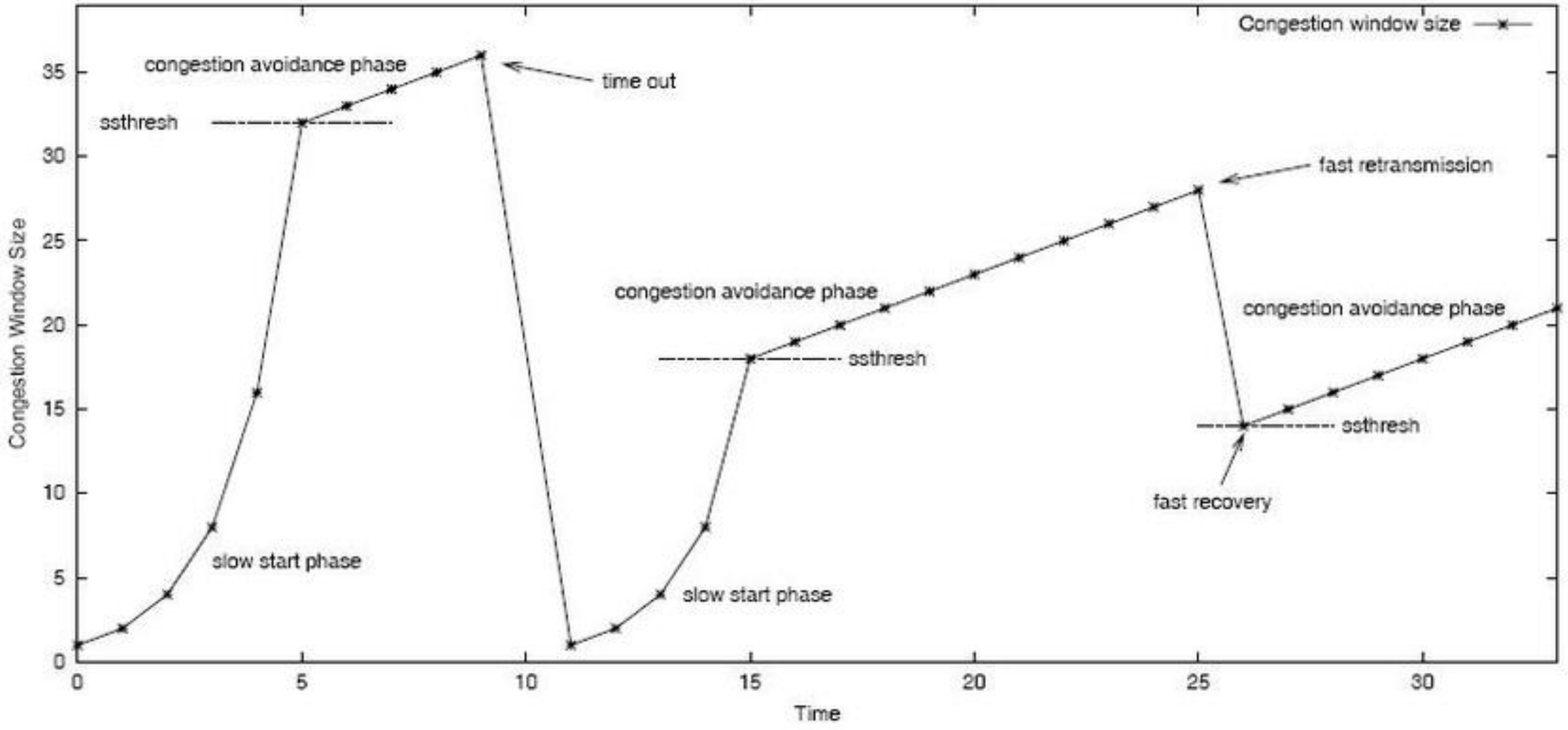

cwnd
Congestion window (Kbytes)
44
40
36
28
24
20
18
16
12
8
4
0
Congestion Avoidance
Congestion point of network
Threshold
(ssthresh)
Threshold
(ssthresh)
Slow start
Time
1
2
4
6
8
10
12
14
16
18
20
22
24
Round Trip Transmission RTT

To acc

K HERE

- **<u>Detection On Receiving 3 Duplicate Acknowledgements</u>**
  - Sender receives 3 duplicate acknowledgements for a segment. This case suggests the weaker possibility of congestion in the network. There are chances that a segment has been dropped but few segments sent later may have reached.
  - **<u>Reaction</u>**
    - In this case, sender reacts by setting the slow start threshold to half of the current congestion window size.
    - Decreasing the congestion window size to slow start threshold.
    - Resuming the congestion avoidance phase.

To access all paid content get KG Prime at ₹25/day CLICK HERE



To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Let the size of congestion window of a TCP connection be 32 KB when a timeout occurs. The round-trip time of the connection is 100 msec and the maximum segment size used is 2 KB. The time taken (in msec) by the TCP connection to get back to 32 KB congestion window is __________. **(Gate-2014) (2 Marks)**

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Consider an instance of TCP's Additive Increase Multiplicative Decrease (AIMD) algorithm where the window size at the start of the slow start phase is 2 MSS and the threshold at the start of the first transmission is 8 MSS. Assume that a timeout occurs during the fifth transmission. Find the congestion window size at the end of the tenth transmission. **(Gate-2012) (2 Marks)**

a) 8 MSS **b)** 14 MSS **c)** 7 MSS **d)** 12 MSS

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q On a TCP connection, current congestion window size is Congestion Window = 4 KB. The window size advertised by the receiver is Advertise Window = 6 KB. The last byte sent by the sender is LastByteSent = 10240 and the last byte acknowledged by the receiver is LastByteAcked = 8192. The current window size at the sender is **(Gate-2005) (2 Marks)**

(A) 2048 bytes **(B)** 4096 bytes **(C)** 6144 bytes **(D)** 8192 bytes

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Suppose that the maximum transmit window size for a TCP connection is 12000 bytes. Each packet consists of 2000 bytes. At some point of time, the connection is in slow-start phase with a current transmit window of 4000 bytes. Subsequently, the transmitter receives two acknowledgements. Assume that no packets are lost and there are no time-outs. What is the maximum possible value of the current transmit window? **(Gate-2004) (2 Marks)**

(A) 4000 bytes **(B)** 8000 bytes **(C)** 10000 bytes **(D)** 12000 bytes

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Which one of the following statements is FALSE? **(Gate-2004) (1 Marks)**

(A) TCP guarantees a minimum communication rate

(B) TCP ensures in-order delivery

(C) TCP reacts to congestion by reducing sender window size

(D) TCP employs retransmission to compensate for packet loss

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

Timers

- **Time-wait timer (Take care of late packets)**
  - never close connection immediately, other wise the port no will be available for some other process, generally we wait for 2*LT. If some packet arrives late then there will a problem.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Keep-alive timer(Close idle connections)

- Sever periodically check connection and close them.
- After keep-alive time duration, server sends 10 probe messages with a gap of 75 seconds and in case of no reply, will close the connection.

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Persistent timer**
  - Window size zero advertise

- **Acknowledgement time**
  - ack timer is used to generate cumulative ack mostly in piggybacking mode, when one segment arrives station starts ack timer, and whatever segments are received with in that time are acknowledges using a single cumulative ack.

- **Time-out timer**
  - will be discussed in detail in next section.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

Timer

- Timer setting in data link layer was very simple, as it was concerned only with adjacent nodes.
  - T_p = d/v
  - Rtt = 2 X T_p
  - Tot = 2 X Rtt

- In Transport layer handling all this is a complex task, as multiple layers, networks and even different paths are involved, so calculating correct tot is difficult task, which may leads to congestion or inefficient use of resources.

- One important conclusion that the value of time out timer should be static it should change with situation

To access all paid content get KG Prime at ₹25/day CLICK HERE

Network Traffic And Time Out Timer

- TCP uses a time out timer for retransmission of lost segments.
- The value of time out timer is dynamic and changes with the amount of traffic in the network.
- Consider Receiver has sent the ACK to the sender and the ACK is on its way through the network. Now, following two cases are possible

  - **High traffic**: If there is high traffic in the network, the time taken by the ACK to reach the sender will be more. So, as per the high traffic, the value of time out timer should be kept large.
    - ***If the value is kept small***, then timer will time out soon. It causes the sender to assume that the segment is lost before reaching the receiver. However, in actual the ACK is delayed due to high traffic. Sender keeps retransmitting the same segment. This overburdens the network and might lead to congestion.

  - **Low traffic**: If there is low traffic in the network, the time taken by the ACK to reach the sender will be less. So, as per the low traffic, the value of time out timer should be kept small.
    - ***If the value is kept large***, Timer will not time out soon. Sender keeps waiting for the ACK even when it is actually lost. This causes excessive delay.

- **Conclusion:** The setting of the time-out timer is very important if we want to use the network efficiently, and the value of the timer must change based on the change in the network scenario.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

Algorithms for Computing Time Out Timer Value

- The algorithms used for computing the value of time out timer dynamically are-
  - Basic Algorithm
  - Jacobson’s Algorithm
  - Karn’s modification

To access all paid content get KG Prime at ₹25/day CLICK HERE

General Rules for Algorithms (Basic algorithm)

- **Rule-01**
  - The value of time out timer for the next segment is increased when Actual round-trip time for the previous segment is found to be increased indicating there is high traffic in the network.

- **Rule-02**
  - The value of time out timer for the next segment is decreased when Actual round-trip time for the previous segment is found to be decreased indicating there is low traffic in the network.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Basic Algorithm The steps followed under Basic Algorithm are-

- **Step-01: Sending 1st Segment**
  - Sender assumes any random value of initial RTT say $IRTT_1$.
  - So, after sending the 1st segment, sender expects its ACK to arrive in time $IRTT_1$.
  - Sender sets time out timer value (TOT) for the 1st segment to be- $\mathbf{TOT_1 = 2 \times IRTT_1}$
  - Suppose ACK for the 1st segment arrives in time $ARTT_1$. Here, $ARTT_1$ = Actual Round-Trip Time for the 1st segment.

To access all paid content get KG Prime at ₹25/day CLICK HERE

- <u>**Step-02: Sending 2nd Segment**</u>
  - Sender computes the value of initial RTT for the 2nd segment using the relation
    - $\mathbf{IRTT_{n+1} = \alpha\ IRTT_n + (1 - \alpha)\ ARTT_n}$
  - Here, α is called smoothing factor where 0 <= α <= 1 (Its value will be given in questions)
  - Substituting n=1, sender gets $IRTT_2 = \alpha\ IRTT_1 + (1 - \alpha)\ ARTT_1$.
  - So, after sending the 2nd segment, sender expects its ACK to arrive in time $IRTT_2$.
  - Sender sets time out timer value (TOT) for the 2nd segment to be- $\mathbf{TOT_2 = 2 \times IRTT_2}$
  - Suppose ACK for the 2nd segment arrives in time $ARTT_2$.
  - Here, $ARTT_2$ = Actual Round-Trip Time for the 2nd segment.

- In the similar manner, algorithm computes the time out timer value for all the further segments.

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Advantages**
  - Time out timer value is flexible to dynamic round trip time.
  - It takes into consideration all the previously sent segments to derive the initial RTT for the current segment.

- **Disadvantage**
  - It always considers Time out timer value = 2 x Initial round trip time.
  - There is no logic behind using the number 2.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Van Jacobson** (born 1950) is an American computer scientist, renowned for his work on TCP/IP network performance and scaling.
- He is one of the primary contributors to the TCP/IP protocol stack—the technological foundation of today's Internet.
- Since 2013, Jacobson is an adjunct professor at the University of California, Los Angeles (UCLA) working on Named Data Networking.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Jacobson’s Algorithm

- Jacobson’s Algorithm is a modified version of the basic algorithm.
- It gives better performance than Basic Algorithm.
- The steps involved in Jacobson’s Algorithm are

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Step-01: Sending 1st Segment-**
  - Sender assumes any random value of initial RTT say $IRTT_1$.
  - So, after sending the 1st segment, sender expects its ACK to arrive in time $IRTT_1$.
  - Sender assumes any random value of initial deviation say ID_1.
  - So, after sending the 1st segment, sender expects there will be a deviation of ID_1 time from $IRTT_1$.
  - Sender sets time out timer value (TOT) for the 1st segment to be-
    - $\mathbf{TOT_1 = 4 \times ID_1 + IRTT_1}$
  - Suppose ACK for the 1st segment arrives in time $ARTT_1$. Here, $ARTT_1$ = Actual Round-Trip Time for the 1st segment.
  - Then, Actual deviation from $IRTT_1$ is given by-
  - $AD_1 = | IRTT_1 – ARTT_1 |$

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Step-02: Sending 2nd Segment-**
  - Sender computes the value of initial RTT for the 2nd segment using the relation-
    - **$IRTT_{n+1} = \alpha\ IRTT_n + (1 – \alpha)\ ARTT_n$**
    - Here, α is called smoothing factor where 0 <= α <= 1 (Its value will be given in questions)
  - Sender computes the value of initial deviation for the 2nd segment using the relation
    - **$ID_{n+1} = \alpha\ ID_n + (1 – \alpha)\ AD_n$**
    - Here, α is called smoothing factor where 0 <= α <= 1 (Its value will be given in questions)
  - Substituting n=1, sender gets
    - $IRTT_2 = \alpha\ IRTT_1 + (1 – \alpha)\ ARTT_1$
    - $ID_2 = \alpha\ ID_1 + (1 – \alpha)\ AD_1$
  - So after sending the 2nd segment, sender expects its ACK to arrive in time $IRTT_2$ with deviation of ID_2 time.
  - Sender sets time out timer value (TOT) for the 2nd segment to be-
    - **$TOT_2 = 4 \times ID_2 + IRTT_2$**
  - Suppose ACK for the 2nd segment arrives in time $ARTT_2$. Here, $ARTT_2$ = Actual Round Trip Time for the 2nd segment.
  - Then, Actual deviation from $IRTT_2$ is given by-
    - $AD_2 = | IRTT_2 – ARTT_2 |$
- In the similar manner, algorithm computes the time out timer value for all the further segments.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Problems with Basic Algorithm and Jacobson's Algorithm

- To calculate initial round trip time, both the algorithms depend on the actual round-trip time of the previous segment through the relation
  - $\mathbf{IRTT_{n+1} = \alpha\, IRTT_n + (1 - \alpha)\, ARTT_n}$

- Consider ACK of some segment arrives to the sender after its initial time out timer goes off. Then, sender will have to re transmit the segment.
- Now for the segment being re transmitted, what should be the initial time out timer value is the concern.
- This is because the ACK is delayed and will arrive after time out. So, ARTT is not available.
- This problem is resolved by Karn's modification.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

- **Karn's algorithm** addresses the problem of getting accurate estimates of the round-trip time for messages when using the Transmission Control Protocol (TCP) in computer networking.

- The algorithm, also sometimes termed as the Karn-Partridge algorithm was proposed in a paper by Phil Karn and Craig Partridge in 1987.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Karn’s Modification

- Karn’s modification states:
  - Whenever a segment has to be re transmitted, do not apply either of Basic or Jacobson’s algorithm since actual RTT is not available.
  - Instead, double the time out timer (TOT) whenever the timer times out and make a retransmission.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Consider the following statements about the timeout value used in TCP.

i. The timeout value is set to the RTT (Round Trip Time) measured during TCP connection establishment for the entire duration of the connection.

ii. Appropriate RTT estimation algorithm is used to set the timeout value of a TCP connection.

iii. Timeout value is set to twice the propagation delay from the sender to the receiver.

Which of the following choices hold? **(Gate-2007) (1 Marks)**

(A) (i) is false, but (ii) and (iii) are true

(B) (i) and (iii) are false, but (ii) is true

(C) (i) and (ii) are false, but (iii) is true

(D) (i), (ii) and (iii) are false

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

Silly Window Syndrome

- Silly Window Syndrome is a problem that arises due to the poor implementation of TCP.
- It degrades the TCP performance and makes the data transmission extremely inefficient.
- The problem is called so because
  - It causes the sender window size to shrink to a silly value.
  - The window size shrinks to such an extent where the data being transmitted is smaller than TCP Header.

- **The problem arises due to following causes**
  - Sender transmitting data in small segments repeatedly
  - Receiver accepting only few bytes at a time repeatedly

- This problem is solved using Nagle’s Algorithm.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Nagle’s Algorithm

- Nagle’s Algorithm tries to solve the problem caused by the sender delivering 1 data byte at a time. Nagle’s algorithm suggests
  - Sender should send only the first byte on receiving one-byte data from the application.
  - Sender should buffer all the rest bytes until the outstanding byte gets acknowledged. In other words, sender should wait for 1 RTT.
  - After receiving the acknowledgement, sender should send the buffered data in one TCP segment.
  - Then, sender should buffer the data again until the previously sent data gets acknowledged.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Clark's Solution

- **Receiver Accepting Only Few Bytes Repeatedly**
  - Consider the receiver continues to be unable to process all the incoming data.
  - In such a case, its window size becomes smaller and smaller.
  - A stage arrives when it repeatedly sends the window size of 1 byte to the sender.
  - **This problem is solved using Clark's Solution.**

To access all paid content get KG Prime at ₹25/day CLICK HERE

- Clark’s Solution tries to solve the problem caused by the receiver sucking up one data byte at a time. **Clark’s solution suggests-**
  - Receiver should not send a window update for 1 byte.
  - Receiver should wait until it has a decent amount of space available.
  - Receiver should then advertise that window size to the sender.
  - **Specifically, the receiver should not send a window update**
    - Until it can handle the MSS it advertised during Three Way Handshake
    - Or until its buffer is half empty, whichever is smaller.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Important Notes

- **Nagle's algorithm is turned off for the applications that require data to be sent immediately.** This is because Nagle's algorithm sends only one segment per round trip time. This impacts the latency by introducing a delay.

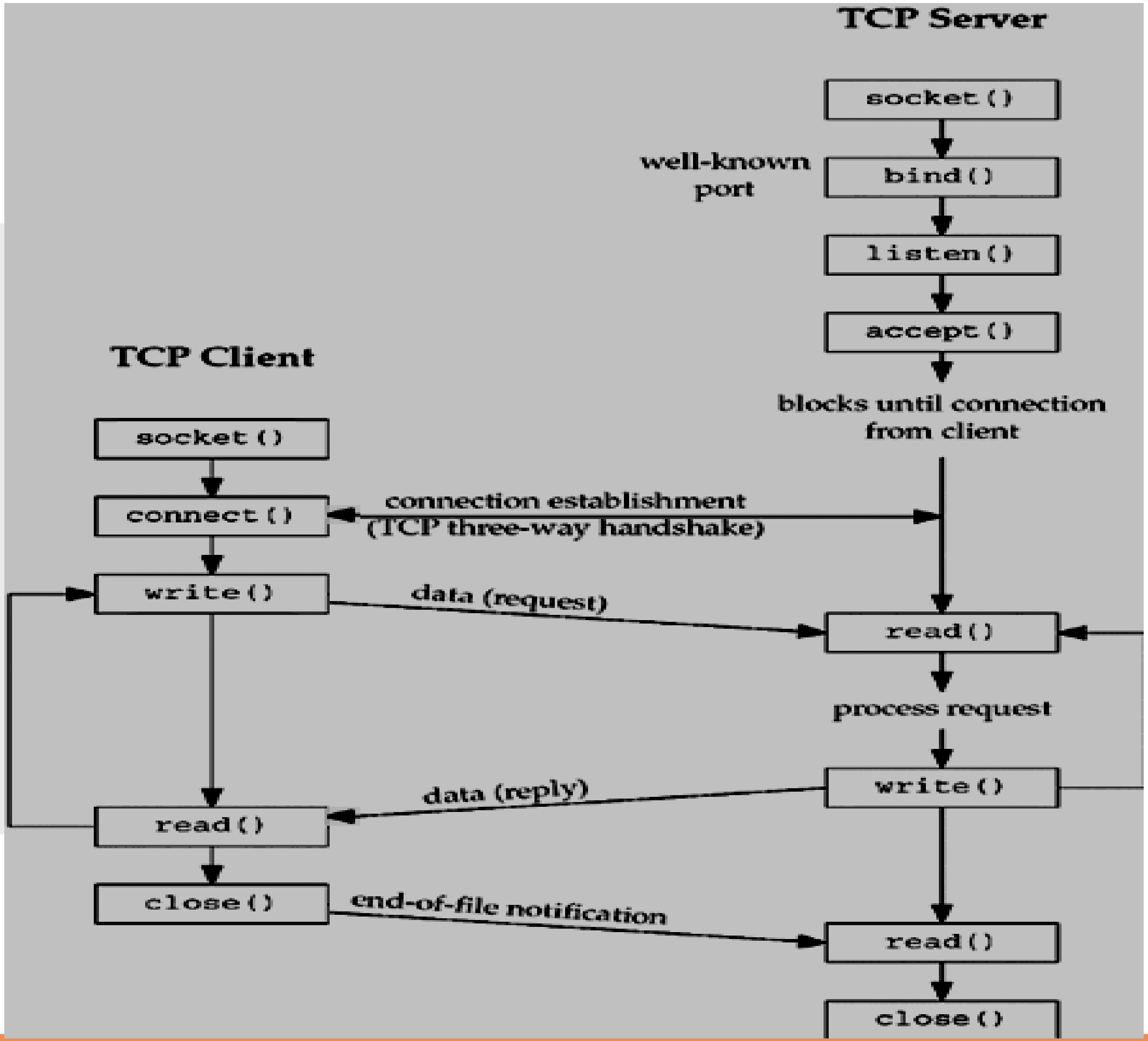
- **Nagle's algorithm and Clark's solution are complementary.** Both Nagle's solution and Clark's solution can work together. The ultimate goal is sender should not send the small segments and receiver should not ask for them.

To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

UNIX socket API



To access all paid content get KG Prime at ₹25/day CLICK HERE

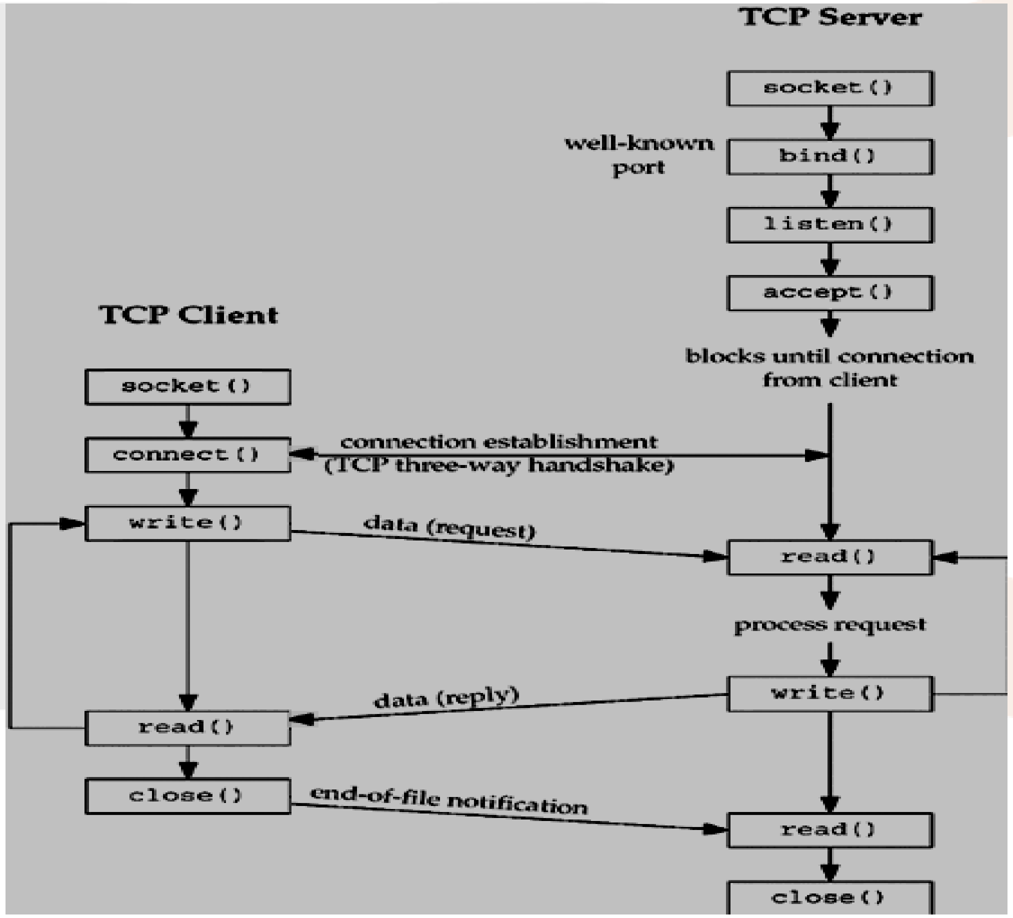
Q Identify the correct order in which a server process must invoke the function calls accept, bind, listen, and recv according to UNIX socket API. **(Gate-2015) (1 Marks)**

(A) listen, accept, bind recv

(B) bind, listen, accept, recv

(C) bind, accept, listen, recv

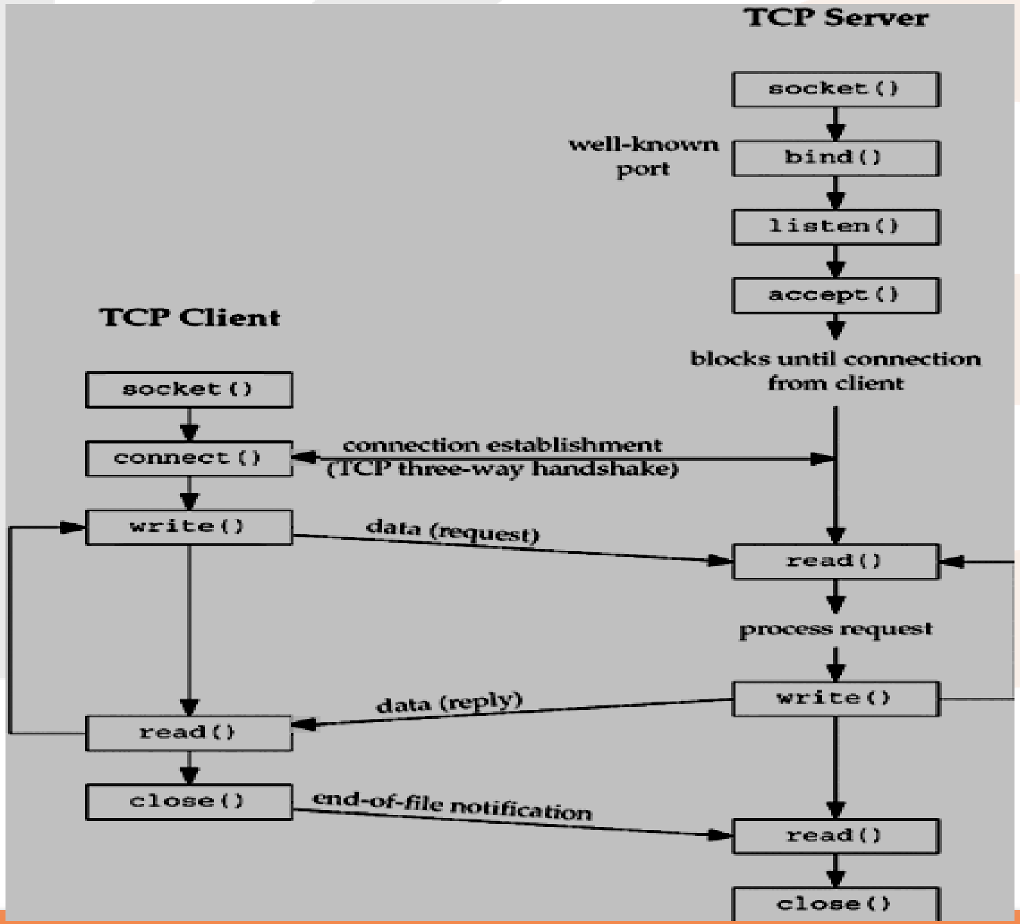
(D) accept, listen, bind, recv



To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Which one of the following socket API functions converts an unconnected active TCP socket into a passive socket? **(Gate-2014) (1 Marks)**

(a) CONNECT **(b)** BIND **(c)** LISTEN **(d)** ACCEPT



To access all paid content get KG Prime at ₹25/day CLICK HERE

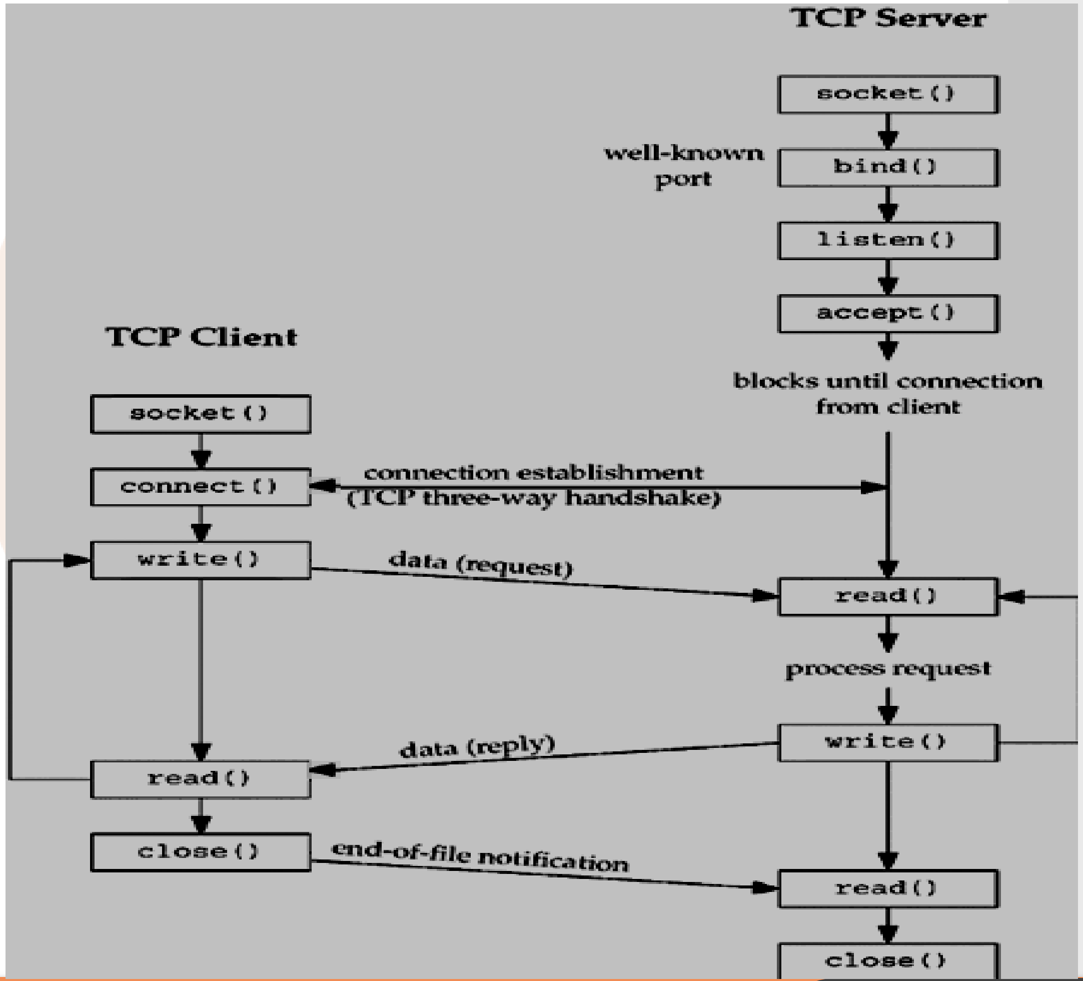
Q A client process P needs to make a TCP connection to a server process S. Consider the following situation: the server process S executes a socket (), a bind () and a listen () system call in that order, following which it is pre-empted. Subsequently, the client process P executes a socket () system call followed by connect () system call to connect to the server process S. The server process has not executed any accept () system call. Which one of the following events could take place? **(Gate-2008) (2 Marks)**

(A) connect () system call returns successfully

(B) connect () system call blocks

(C) connect () system call returns an error

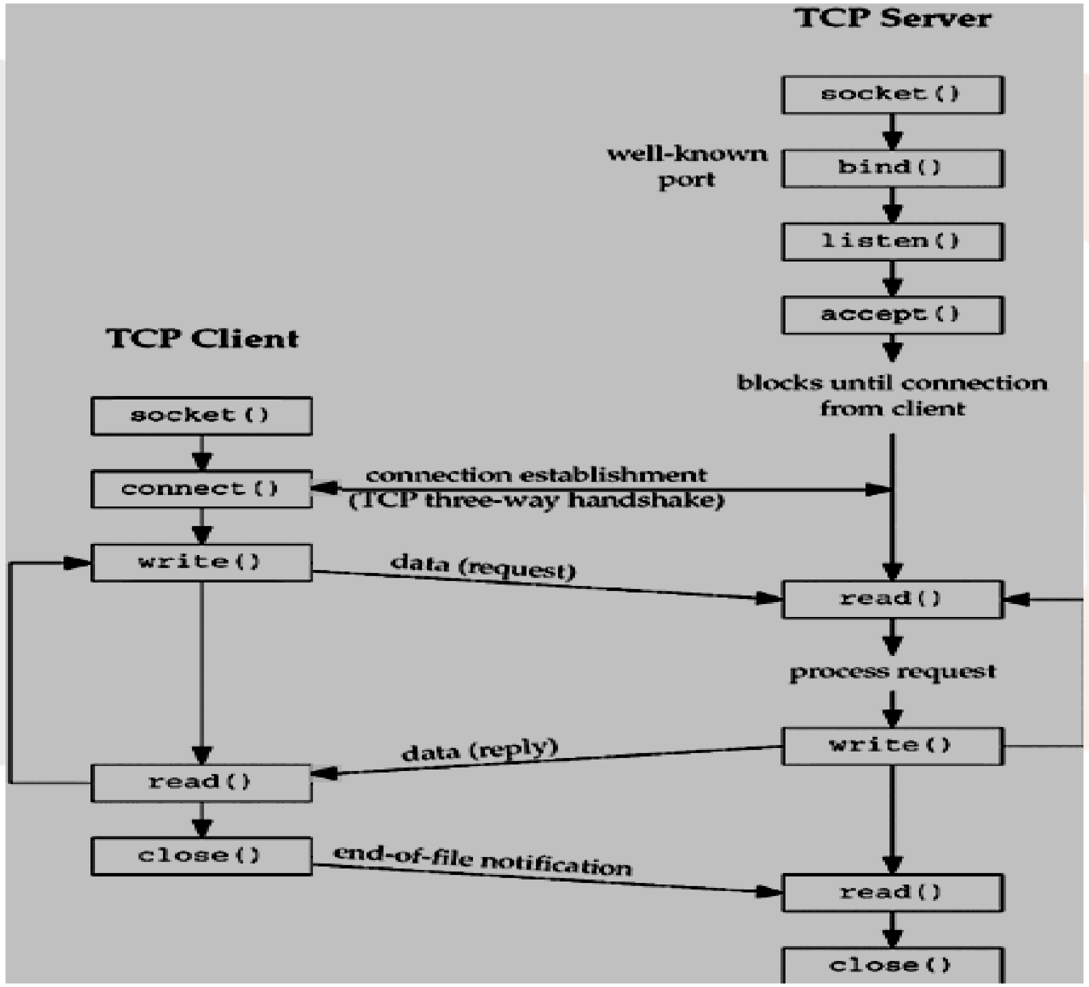
(D) connect () system call results in a core dump



To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Which of the following system calls results in the sending of SYN packets? **(Gate-2008) (1 Marks)**

(A) socket **(B)** bind **(C)** listen **(D)** connect



To access all paid content get KG Prime at ₹25/day CLICK HERE

Break

To access all paid content get KG Prime at ₹25/day CLICK HERE

USER DATAGRAM PROTOCOL (UDP)

- The **User Datagram Protocol (UDP)** is a connectionless, unreliable transport protocol.
- It does not add anything to the services of IP except for providing process-to-process communication instead of host-to-host communication.

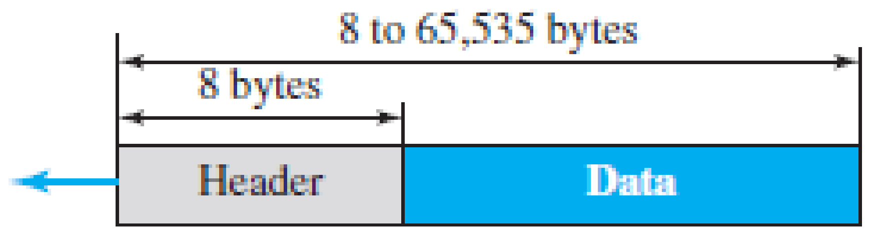
To access all paid content get KG Prime at ₹25/day CLICK HERE

- **<u>Why to use UDP</u>**
  - UDP is a very simple protocol using a minimum of overhead.
  - If a process wants to send a small message and does not care much about reliability, it can use UDP.
  - Sending a small message using UDP takes much less interaction between the sender and receiver than using TCP.

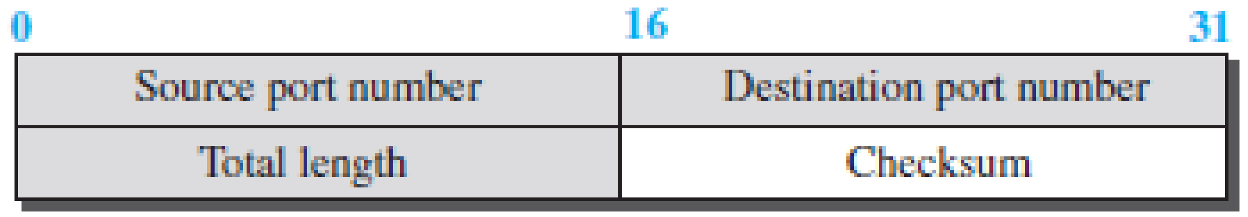
To access all paid content get KG Prime at ₹25/day CLICK HERE

User Datagram

- UDP packets, called ***user datagrams,*** have a fixed-size header of 8 bytes made of four fields, each of 2 bytes (16 bits).



a. UDP user datagram



b. Header format

- The first two fields define the source and destination port numbers.
- The third field defines the total length of the user datagram, header plus data.
- The 16 bits can define a total length of 0 to 65,535 bytes. However, the total length needs to be less because a UDP user datagram is stored in an IP datagram with the total length of 65,535 bytes.
- The last field can carry the optional checksum

To access all paid content get KG Prime at ₹25/day CLICK HERE

UDP Services

- ***Process-to-Process Communication***
  - UDP provides process-to-process communication using **socket addresses,** a combination of IP addresses and port numbers.

- ***Connectionless Services***
  - Each user datagram sent by UDP is an independent datagram. There is no relationship between the different user datagrams.
  - The user datagrams are not numbered.
  - There is no connection establishment and no connection termination unlike TCP.

- Only those processes sending short messages, messages less than 65,507 bytes (65,535 minus 8 bytes for the UDP header and minus 20 bytes for the IP header), can use UDP.

To access all paid content get KG Prime at ₹25/day CLICK HERE

- ***Flow Control***
  - There is no *flow control,* and hence no window mechanism.

- ***Error Control***
  - There is no *error control* mechanism in UDP except for the checksum.

- ***Congestion Control***
  - It does not provide congestion control.

To access all paid content get KG Prime at ₹25/day CLICK HERE

UDP Applications

- UDP is suitable for a process that requires simple request-response communication with little concern for flow and error control. It is not usually used for a process such as ***FTP*** that needs to send bulk data.

- UDP is suitable for a process with internal flow- and error-control mechanisms. For example, the ***Trivial File Transfer Protocol (TFTP)*** process includes flow and error control. It can easily use UDP.

- UDP is a suitable transport protocol for multicasting. Multicasting capability is embedded in the UDP software but not in the TCP software.

- UDP is used for management processes such as SNMP

- UDP is used for some route updating protocols such as Routing Information Protocol (RIP)

- UDP is normally used for interactive real-time applications that cannot tolerate uneven delay between sections of a received message.

- DNS,

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Match the following: **(GATE-2018) (1 Marks)**

Field	Length in bits
P. UDP Header's Port Number	I. 48
Q. Ethernet MAC Address	II. 8
R. IPv6 Next Header	III.32
S. TCP Header's Sequence Number	IV. 16

(a) P-III, Q-IV, R-II, S-I

(b) P-II, Q-I, R-IV, S-III

(c) P-IV, Q-I, R-II, S-III

(d) P-IV, Q-I, R-III, S-II

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Which of the following statements are TRUE? **(Gate-2008) (2 Marks)**

(S_1) TCP handles both congestion and flow control

(S_2) UDP handles congestion but not flow control

(S_3) Fast retransmit deals with congestion but not flow control

(S_4) Slow start mechanism deals with both congestion and flow control

(A) S_1, S_2 and S_3 only

(B) S_1 and S_3 only

(C) S_3 and S_4 only

(D) S_1, S_3 and S_4 only

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q A program on machine X attempts to open a UDP connection to port 5376 on a machine Y, and a TCP connection to port 8632 on machine Z. However, there are no applications listening at the corresponding ports on Y and Z. An ICMP Port Unreachable error will be generated by **(Gate-2006) (2 Marks)**

(A) Y but not Z

(B) Z but not Y

(C) Neither Y nor Z

(D) Both Y and Z

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Packets of the same session may be routed through different paths in: **(Gate-2005) (1 Marks)**

(a) TCP, but not UDP

(b) TCP and UDP

(c) UDP, but not TCP

(d) Neither TCP nor UDP

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q A firewall is to be configured to allow hosts in a private network to freely open TCP connections and send packets on open connections. However, it will only allow external hosts to send packets on existing open TCP connections or connections that are being opened (by internal hosts) but not allow them to open TCP connections to hosts in the private network. To achieve this the minimum capability of the firewall should be that of **(GATE-2007) (1 Marks)**

(A) A combinational circuit

(B) A finite automaton

(C) A pushdown automaton with one stack

(D) A pushdown automaton with two stacks

To access all paid content get KG Prime at ₹25/day CLICK HERE

Q Which one of the following statements is FALSE? **(Gate-2004) (1 Marks)**

(A) TCP guarantees a minimum communication rate

(B) TCP ensures in-order delivery

(C) TCP reacts to congestion by reducing sender window size

(D) TCP employs retransmission to compensate for packet loss

To access all paid content get KG Prime at ₹25/day CLICK HERE