SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with 'A' Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Information Technology

Course Name – 19IT401 Computer Networks

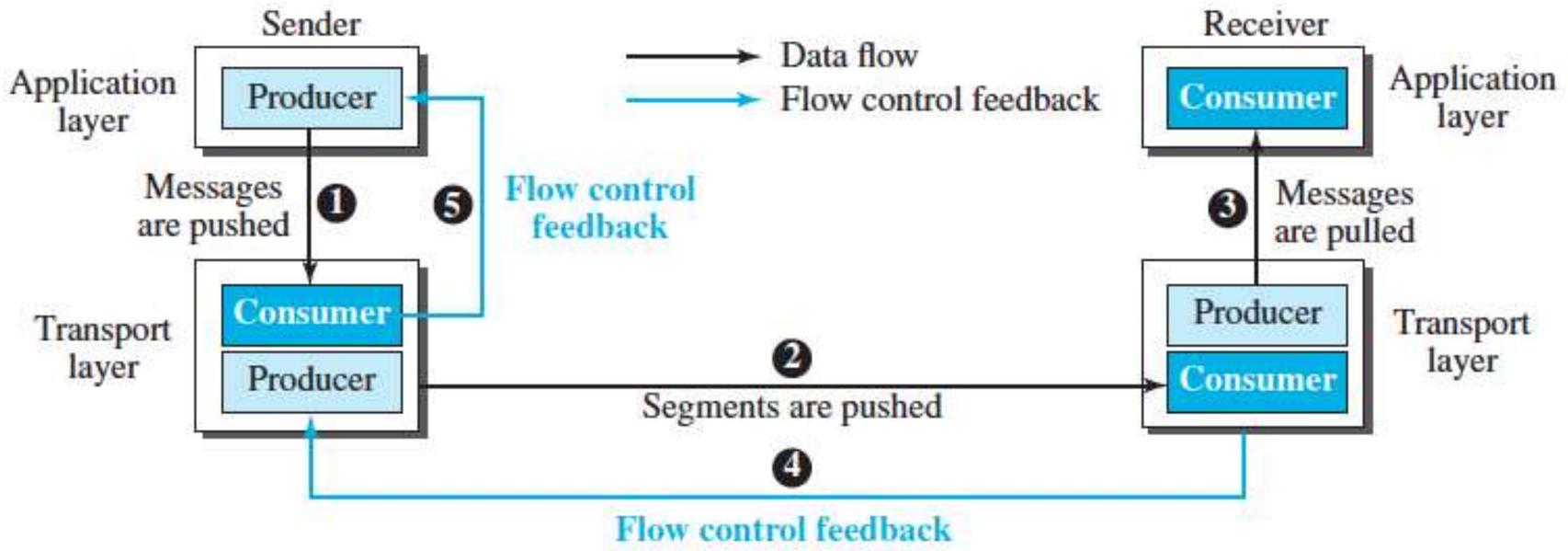
II Year / IV Semester

Unit 4 – Transport Layer

Topic 3 – Flow Control

Flow Control

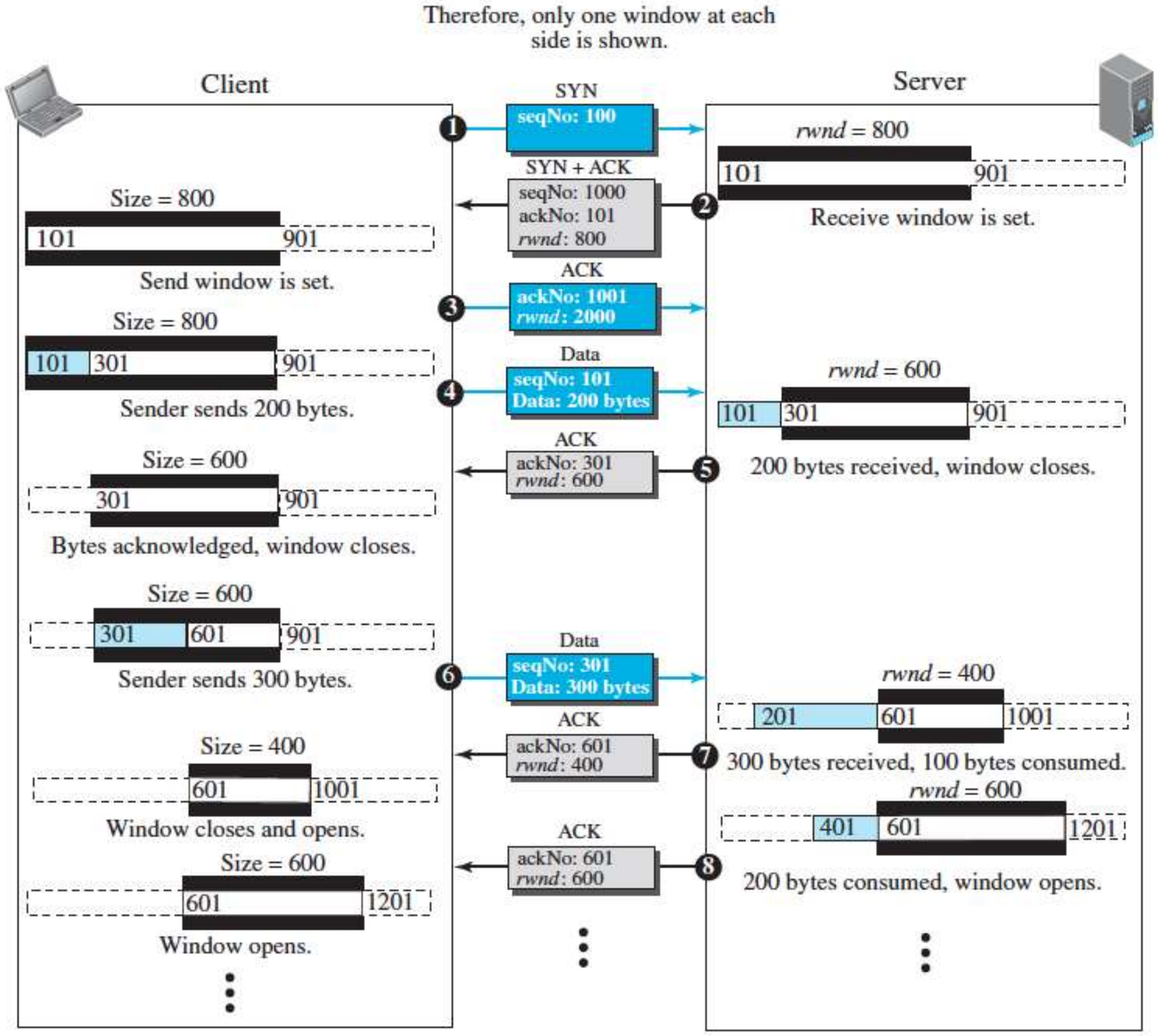
- Flow control balances the rate a sender creates data with the rate a receiver can use the data.



Flow Control

Opening and Closing Windows

- To achieve flow control, TCP forces the sender and the receiver to adjust their window sizes.
- The receive window closes (moves its left wall to the right) when more bytes arrive from the sender;
- It opens (moves its right wall to the right) when more bytes are pulled by the process.
- The opening, closing, and shrinking of the send window is controlled by the receiver.
- The send window closes (moves its left wall to the right) when a new acknowledgment allows it to do so.
- The send window opens (its right wall moves to the right) when the receive window size (rwnd) advertised by the receiver allows it to do so.


Therefore, only one window at each side is shown.
Client
Server
SYN
seqNo: 100
rwnd = 800
101
901
Receive window is set.
SYN + ACK
seqNo: 1000
ackNo: 101
rwnd: 800
Size = 800
101
901
Send window is set.
ACK
ackNo: 1001
rwnd: 2000
Size = 800
101
301
901
Sender sends 200 bytes.
Data
seqNo: 101
Data: 200 bytes
rwnd = 600
101
301
901
200 bytes received, window closes.
ACK
ackNo: 301
rwnd: 600
Size = 600
301
901
Bytes acknowledged, window closes.
Size = 600
301
601
901
Sender sends 300 bytes.
Data
seqNo: 301
Data: 300 bytes
rwnd = 400
201
601
1001
300 bytes received, 100 bytes consumed.
ACK
ackNo: 601
rwnd: 400
Size = 400
601
1001
Window closes and opens.
rwnd = 600
401
601
1201
200 bytes consumed, window opens.
ACK
ackNo: 601
rwnd: 600
Size = 600
601
1201
Window opens.

Flow Control

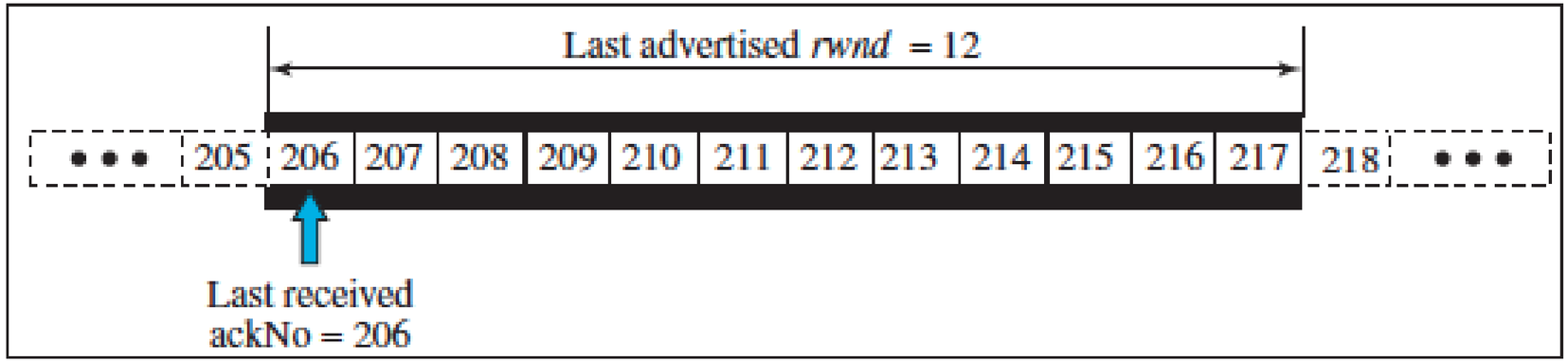
Shrinking of Windows

- The receive window cannot shrink.
- The send window, on the other hand, can shrink if the receiver defines a value for rwnd that results in shrinking the window.
- However, some implementations do not allow shrinking of the send window.
- The limitation does not allow the right wall of the send window to move to the left
- The receiver needs to keep the following relationship between the last and new acknowledgment and the last and new rwnd values to prevent shrinking of the send window.

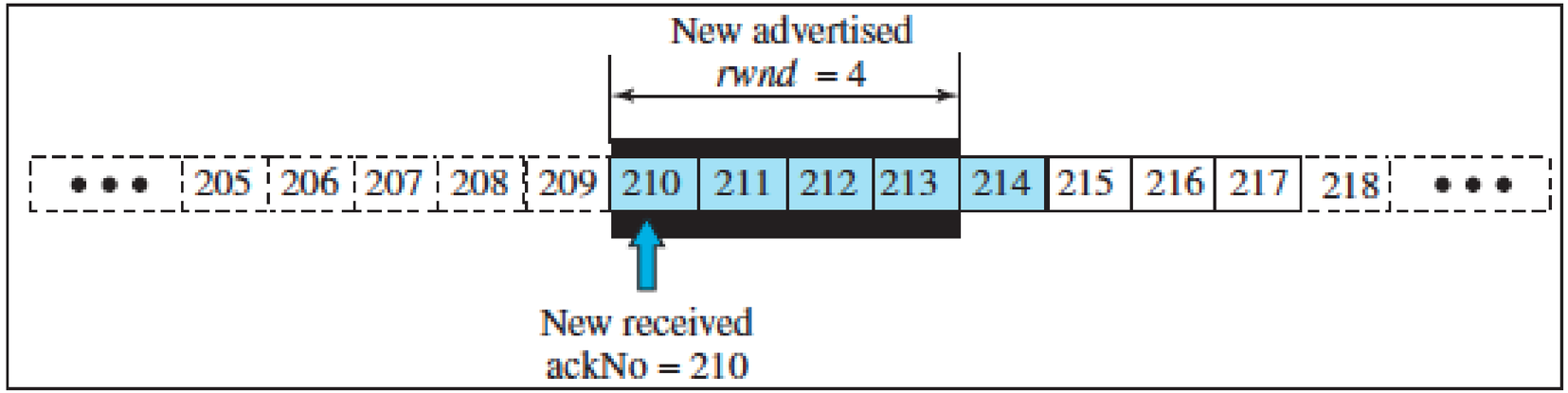
$$\text{new ackNo} + \text{new rwnd} \geq \text{last ackNo} + \text{last rwnd}$$

- The relationship shows that the right wall should not move to the left.

Flow Control



a. The window after the last advertisement



b. The window after the new advertisement; window has shrunk

Flow Control

Silly Window Syndrome

- A serious problem can arise in the sliding window operation when either the sending application program creates data slowly or the receiving application program consumes data slowly, or both.

- Any of these situations results in the sending of data in very small segments, which reduces the efficiency of the operation. It degrades the TCP performance and makes the data transmission extremely inefficient.

- It causes the sender window size to shrink to a silly value as low as 1 byte.

- The window size shrinks to such an extent where the data being transmitted is smaller than TCP Header.(20 bytes TCP Header + 20 bytes IP header). If one byte transferred trade off 41/1.

The two major causes of this syndrome are as follows:

- Sender window transmitting one byte of data repeatedly.

- Receiver window accepting one byte of data repeatedly.

Flow Control

Syndrome Created by the Sender

- The sending TCP may create a silly window syndrome if it is serving an application program that creates data slowly, for example, 1 byte at a time.
- The application program writes 1 byte at a time into the buffer of the sending TCP.
- The result is a lot of 41-byte segments that are traveling through an internet, which is very inefficient.
- To prevent this The sending TCP must be forced to wait and collect data to send in a larger block

Nagle's algorithm

1. The sending TCP sends the first piece of data it receives from the sending application program even if it is only 1 byte.
2. After sending the first segment, the sending TCP accumulates data in the output buffer and waits until either the receiving TCP sends an acknowledgment or until enough data have accumulated to fill a maximum-size segment. At this time, the sending TCP can send the segment.
3. Step 2 is repeated for the rest of the transmission. Segment 3 is sent immediately if an acknowledgment is received for segment 2, or if enough data have accumulated to fill a maximum-size segment.

Flow Control

Syndrome Created by the Receiver

- The receiving TCP may create a silly window syndrome if it is serving an application program that consumes data slowly, for example, 1 byte at a time.
- This makes receiving process slow and inefficient.
- There are two solution to this problem.

Clark's solution

- Send an acknowledgment as soon as the data arrive, but to announce a window size of zero until either there is enough space to accommodate a segment of maximum size or until at least half of the receive buffer is empty.

Delayed Acknowledgement

- The second solution is to delay sending the acknowledgment. This means that when a segment arrives, it is not acknowledged immediately. The receiver waits until there is a decent amount of space in its incoming buffer before acknowledging the arrived segments.
- The delayed acknowledgment prevents the sending TCP from sliding its window.
- Delayed acknowledgment also has another advantage: it reduces traffic.
- However, there also is a disadvantage in that the delayed acknowledgment may result in the sender unnecessarily retransmitting the unacknowledged segments.
- Therefore acknowledgment should not be delayed by more than 500 ms

THANK YOU