CSE2029: Data Communication & Computer Networks

Lecture-9: Transport Layer

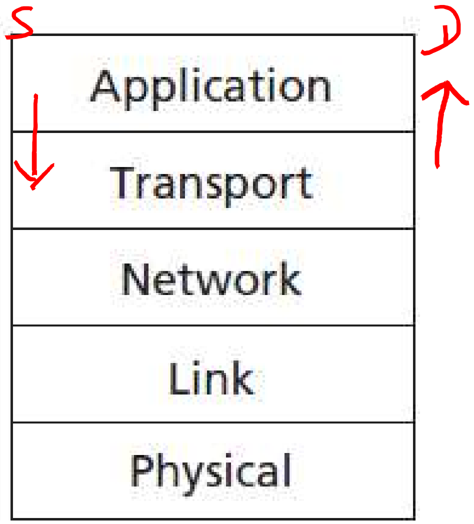
Faculty: Dr. Sandeep Kumar

Outline

- ***Transport Layer*** ✓
    - ***Introduction*** ✓
    - ***Functions*** ✓
    - ***Services*** ✓
- ***TCP***
    - ***Connection Establishment***
    - ***Connection Release***
    - ***Flow Control***

Transport Layer

- The Internet's transport layer transports application-layer messages between application endpoints.
- There **are two transport protocols, TCP and UDP**, either of which can transport application-layer messages.
- **TCP** provides a connection-oriented service to its applications. This service includes guaranteed delivery of application-layer messages to the destination and flow control.
- The **UDP** protocol provides a connectionless service to its applications. This is a no-frills service that provides no reliability, no flow control, and no congestion control.
- The packet of information at the transport layer as a **segment**.



Five-layer Internet protocol stack

Transport Layer

program
or
application
↑ or

- The Transport layer is responsible for **process-to-process or end-end** delivery of the entire message.

- The transport layer ensures that the whole message arrives intact and overseeing both:

Error control and flow control at the process-to-process level.

Transport Layer
Functions

- Service point addressing(Process-Process delivery)of messages
- Segmentation and reassembly
- Connection control

multiplexing

- Flow control(QoS) – MUX & Demux
- Error control – error checking and recovery
- Congestion control

Transport Layer

Services

- Transport Layer Provides :
  - **Efficient**
  - **Reliable and**
  - **Cost-effective services**
- Another TWO Kinds of Services are :
  - **Connection oriented - TCP** ✓
  - **Connectionless - UDP** ✓

Transport Layer Simple Service Primitives

- Simple primitives:
  - Connect ✓
  - Send ✓
  - Receive ✓
  - Disconnect ✓

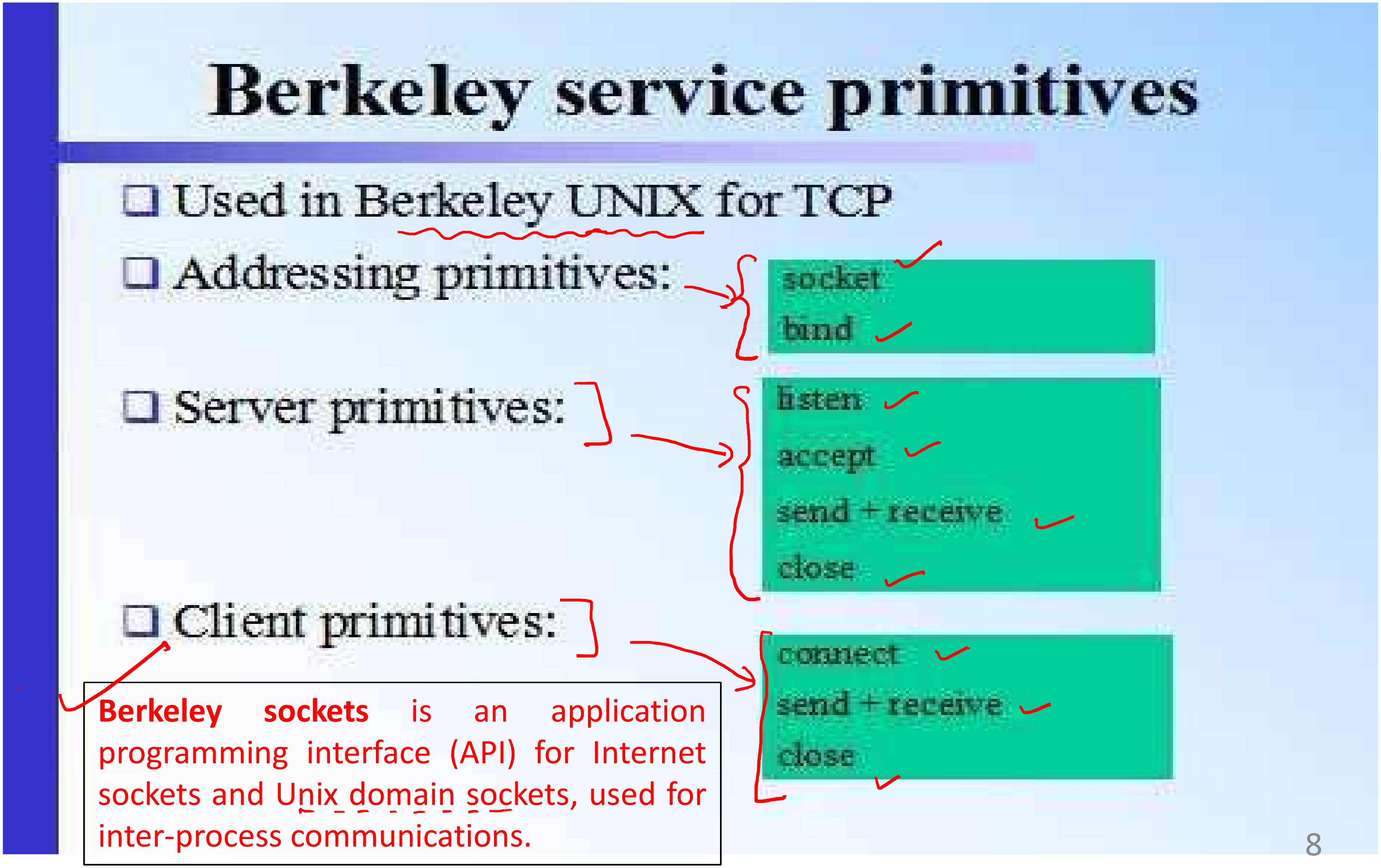
A **primitive** simply means Operations. A Service is specified by set of primitives that are available and given to user or other various entities to access the service.

- How to handle incoming connection request in server process?

  ➔Wait for connection request from client!

  - listen ✓

Transport Layer Berkeley Service Primitives



Berkeley service primitives

- ❑ Used in Berkeley UNIX for TCP
- ❑ Addressing primitives:
  - socket
  - bind
- ❑ Server primitives:
  - listen
  - accept
  - send + receive
  - close
- ❑ Client primitives:
  - connect
  - send + receive
  - close

Berkeley sockets is an application programming interface (API) for Internet sockets and Unix domain sockets, used for inter-process communications.

TCP Connection Establishment

- Once a connection is established, both client and server may exachnge data using several system calls.
- A connection is typically used for client-server interaction.
- A server advertizes a particular service at a well-known address and clients establish connections to that socket to avail of the offered service.
- Thus the connection estblishment procedure is asymmetric.

–Problems to solve (by TCP)

- Selection of the initial sequence number for a new connection.
- Wrap around of sequence numbers for an active connection.
- To Handle host crashes.

TCP: connection release

- Asymmetric
  - Connection broken when one party hangs up
  - It may result in data loss

- ✓ Symmetric
  - Both parties should agree to release connection
  - How to reach agreement?
  - Solution: three-way-handshake

to be discussed later on....

TCP: Flow Control

- It is a set of procedures to tell the sender how much data it can transmit before it must wait for an acknowledgement from the receiver.

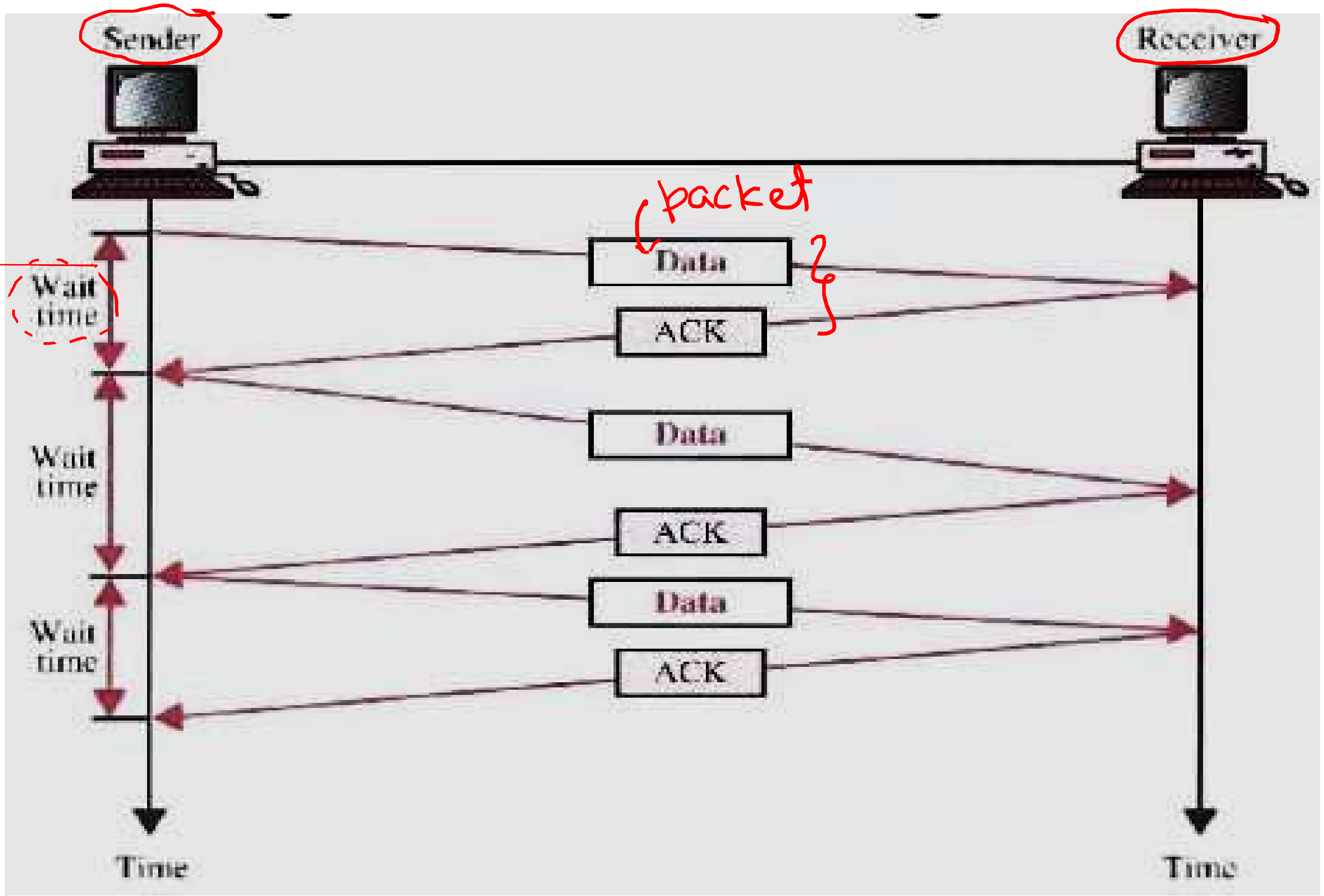
- Two categories of flow control:
  - Stop-and-wait

    Send one packet at a time.
  - Sliding window

    Send several packets at a time.

Stop-and-wait

- Sender sends one packet and waits for an acknowledgement before sending the next packet.



Stop-and-wait

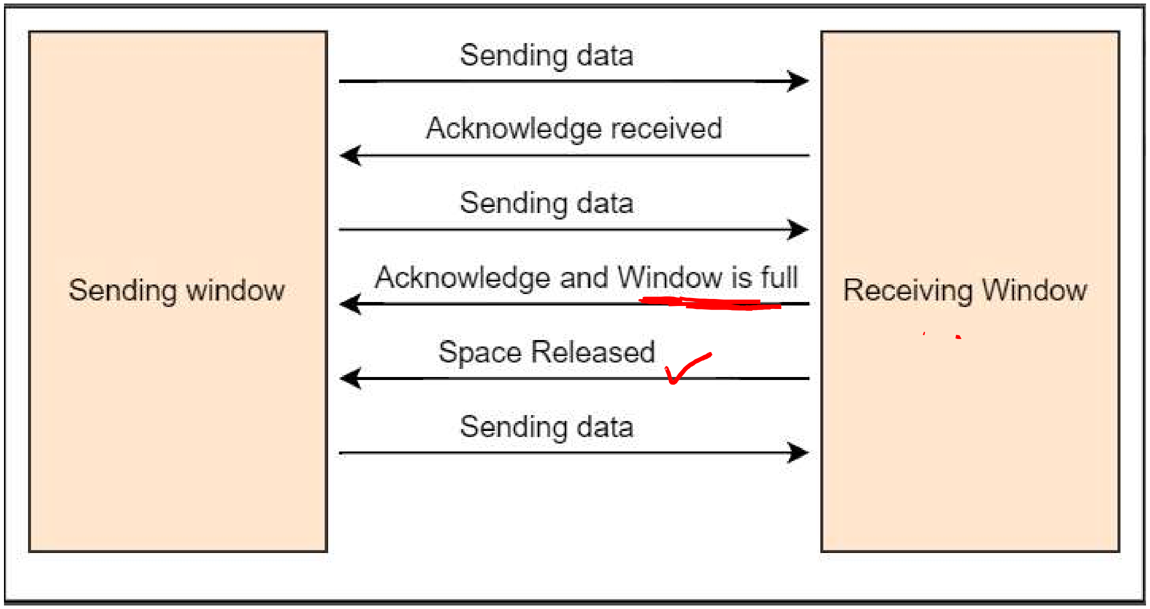
- Advantages:
  - Simplicity.
  - Each packet is checked and acknowledged before the next packet is sent.
- Disadvantages:
  - Slow.
    - Can add significantly to the total transmission time if the distance between devices is long.
  - Inefficiency
    - Each packet is alone on the line.

Sliding Window

- The TCP sliding window determines the number of **data bytes, *x***, that one system can send to another.
- Two factors determine the value of ***x***: (1) The size of the send buffer on the sending system, (2) The size and available space in the receive buffer on the receiving system.
- The sending system cannot send more bytes than space that is available in the receive buffer on the receiving system.
- On the receiving system, TCP stores received data in a receive buffer. TCP acknowledges receipt of data → (packets) to the sender, and keep advertising **new receive windows** to the sending system. The receive window represents the number of bytes that are available in the receive buffer.

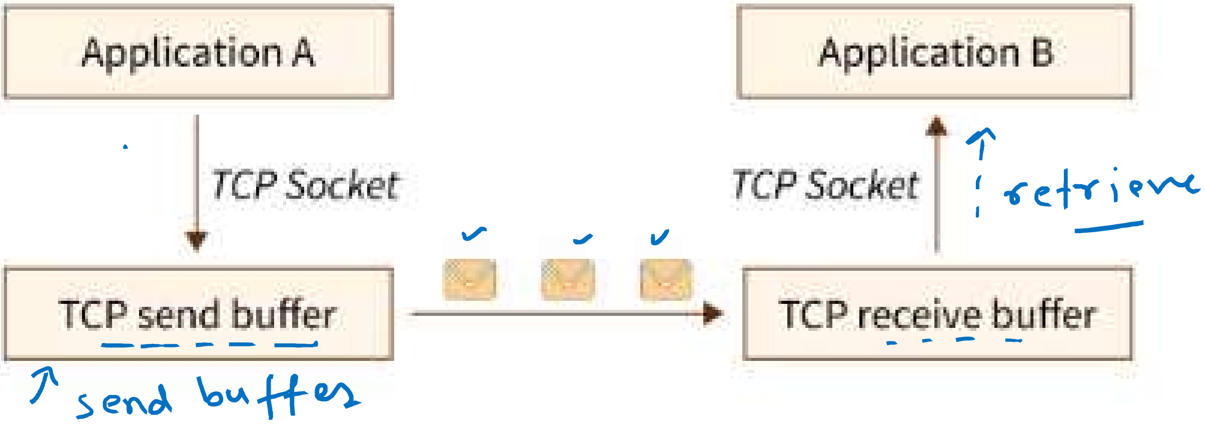
Sliding Window

- If the receive buffer is full, the receiving system advertises a receive window size of zero, and the sending system must wait to send more data.
- After the receiving application retrieves data from the receive buffer, the receiving system can then advertise a receive window size that is equal to the amount of data that was read (retrieved). Then, TCP on the sending system can resume sending data.



Sliding Window

1



Application A

TCP Socket

TCP send buffer

send buffer

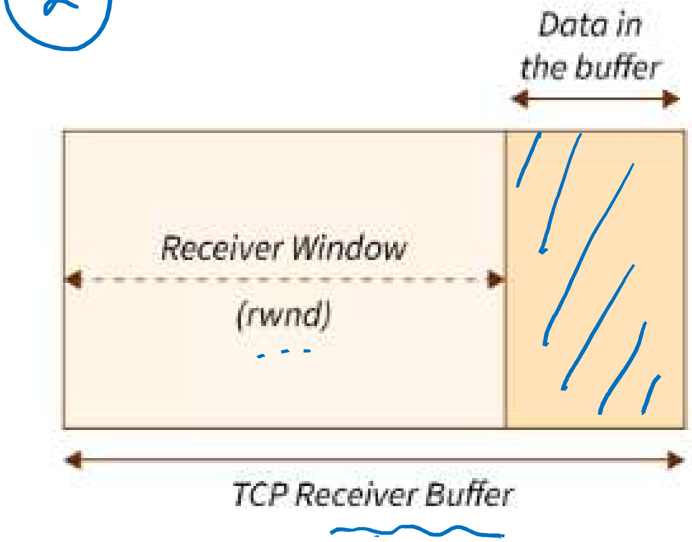
Application B

TCP Socket

retrieve

TCP receive buffer

2



Data in the buffer

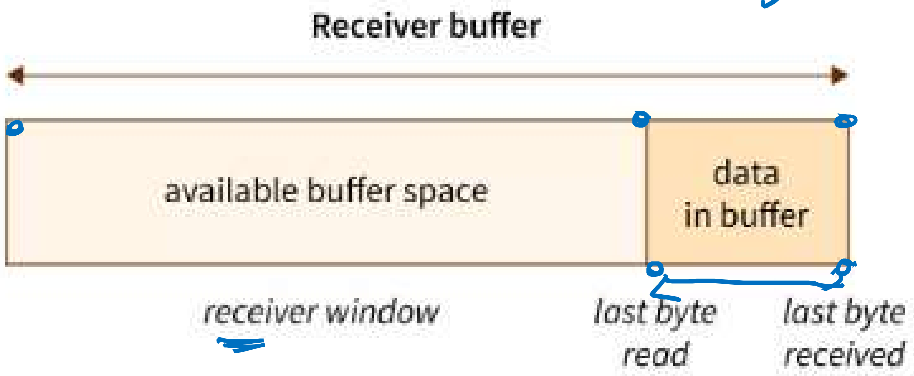
Receiver Window (rwnd)

TCP Receiver Buffer

3

Receiver Window (rwnd)

rwnd = receive buffer - (last byte received - last byte read)



Receiver buffer

available buffer space

data in buffer

receiver window

last byte read

last byte received

To be continued in next lecture.
Thank you.