Networks I

Transmission Control Protocol

Prof. MSc. Ivan A. Escobar Broitman

Instituto Tecnológico y de Estudios Superiores de
Monterrey Campus Estado de México

TC1007

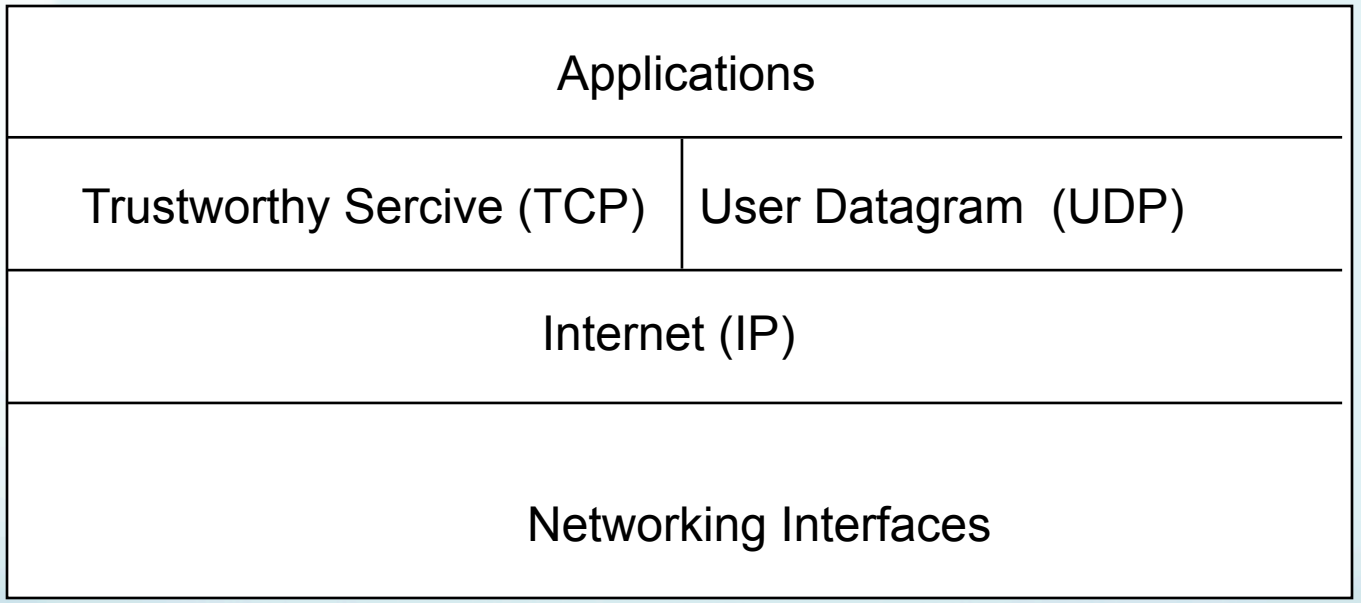
TCP

- Transmission Control Protocol
  - Designed specifically to provide end to end byte streams over unreliable networks.
  - Connection Oriented Protocol.
- Internet Protocol:
  - Connectionless packet delivery service, best effort system.
  - Not reliable by its own.

TCP: Introduction

- General purpose transport protocol.
- TCP can run over any routed protocol not just necessarily IP.
- TCP is identified in the IP datagram by checking the protocol field, which has a number of 6.

TCP and the OSI model



TCP: Properties

- Caracter flow based.
- Virtual Circuits.
- Manages data using segments.
  - Character flow is generated by blocks (application).
  - Block handed to TCP, waits to fill in a buffer (segment).
  - Push Mechanism.
- Full Duplex connections.
- Provides a reliable flow of data.
- Piggybacking.

TCP: Services

- Defines data format.
- Distinguishes between different destinations under the same host.
  - Port Demultiplexing.
- Error Control.
- Flow of data.
  - Point to point.
  - Congestion.

TCP: Ports and Connections

- TCP allows multiple applications in the same host to communicate with the outside world in a concurrent manner.
- Each applications is assigned a port number.
- An extreme point can be defined by the pair host, port number.
- A TCP connection is defined by two extreme points.

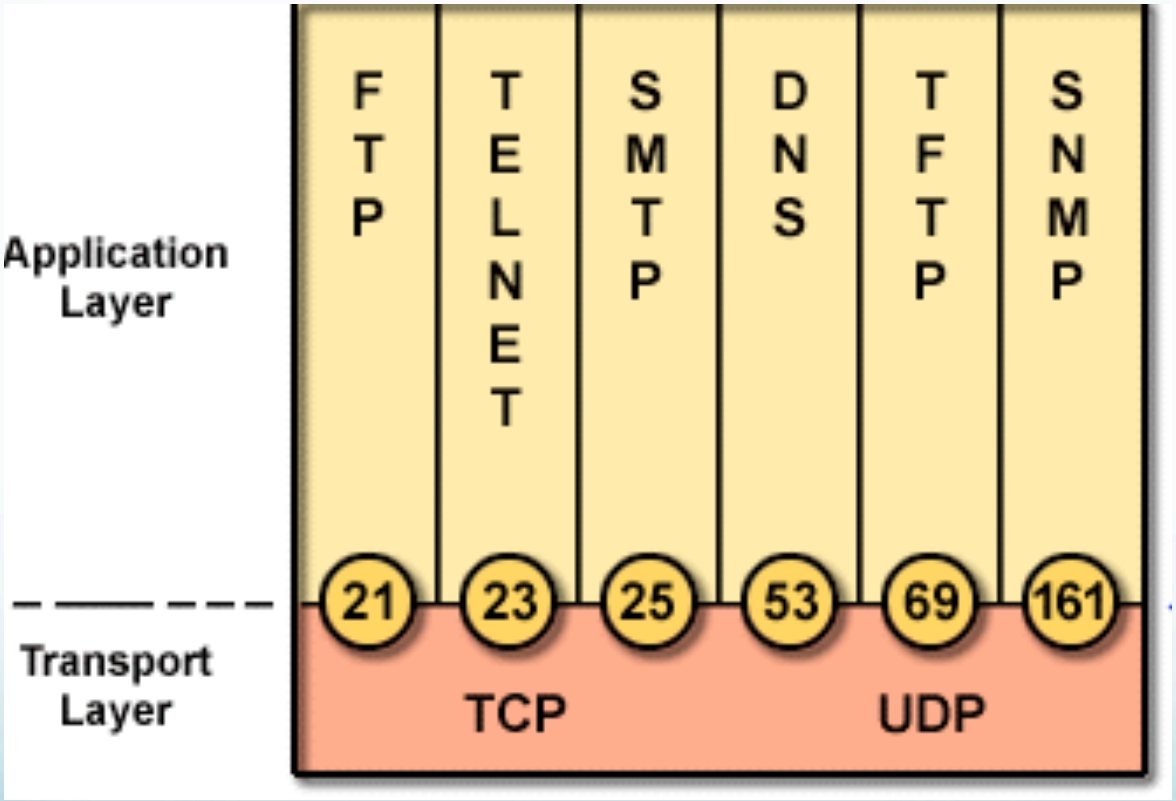
TCP: Service Model

- Extreme or end points are created by the sender and the receiver. They are called “sockets”.
- Socket number:
  - Host’s IP address.
  - Port: 16 bit number.

TCP: Socket Ports

- Well known ports.
  - Port numbers below 1024.
  - An application willing to establish a FTP session from a host to a destination must establish a connection to port 21.
  - Similarly a telnet session must use port 23.
  - RFC 1700.

Ports


Application Layer
FTP
TELNET
SMTP
DNS
TFTP
SNMP
21
23
25
53
69
161
Transport Layer
TCP
UDP

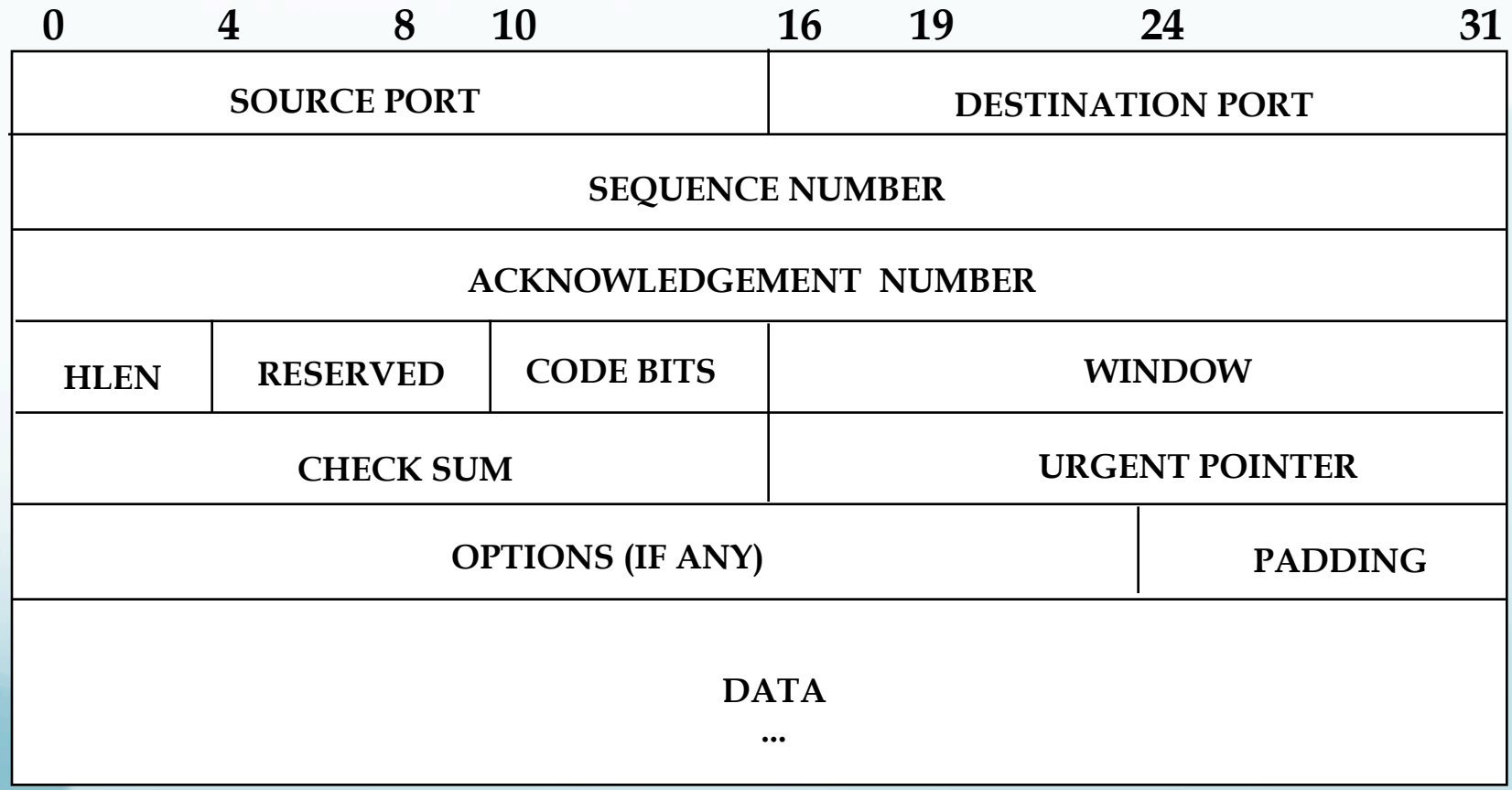
The TCP Protocol

- The sending and receiving TCP entities exchange data in the form of segments.
- Segment:
  - 20 byte header (+ options).
  - Zero or more data bytes.
- The TCP software decides how big the segments should be.
- Size restricted by:
  - IP payload 65,536 bytes.
  - Network MTU's

The TCP Protocol

- If a segment is too large to transit in a certain type of network, it can be broken up into multiple segments by a router (fragmentation).
- Each new segment will get its own IP header.
  - Fragmentation by a router increments overhead.
  - Each additional segment adds 20 bytes.
- The basic protocol used by TCP entities is the sliding windows protocol.

The TCP Segment Header



The TCP Segment Header

- Source/Destination ports (16 bits each):
  - Identify local end points of a connection.
  - One port plus a host's IP address for a unique 48 TSAP.
  - This connection is identified by the source and destination socket numbers.
- Sequence Number/ ACK number(32 bits each):
  - SEQ: used by destinations to match up packets.
  - ACK: specifies the next byte expected.
- HLEN(4 bits)
  - Header length, variable due to options. Minimum 20 bytes.
- Unused (6 bits).

The TCP Segment Header

- Code Bits(6 bits) flags:

  Set when value = 1

  - URG:
    - Urgent pointer in use to indicate byte offset of sequence number.
  - ACK:
    - When set used to indicate that the ACK is valid.
    - When cero, segment doesn't contain an ACK, ignored.

- Code Bits(6 bits) flags:

  Set when value = 1

  - PSH:
    - Indicates Pushed data.
    - Receiver must deliver data directly to the application without buffering.
  - RST:
    - Resets a connection with problems.
    - Rejects an invalid segment.

The TCP Segment Header

- Code bits(6 bits) flags:

  Set when value = 1

  - SYN:
    - Used to establish connections.
    - Connection request has SYN=1 ACK=0 to indicate that piggybacking ack field is not used.
    - Connection accepted SYN=1 ACK=1.

- Code bits(6 bits) flags:

  Set when value = 1

  - FIN:
    - Used to release the connection.
    - It specifies that the sender has no more data to transmit.

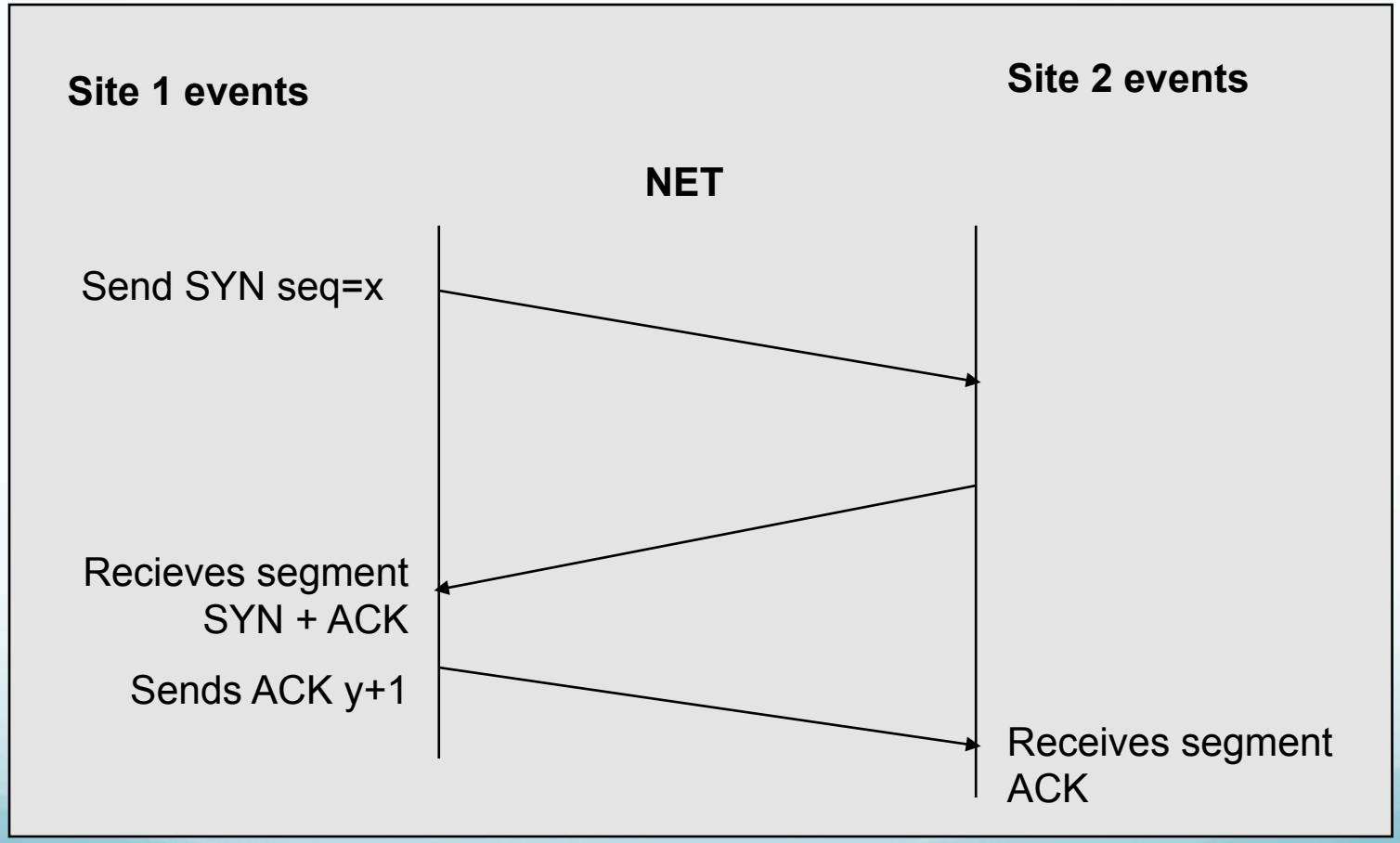
The TCP Segment Header

- Window(16 bits):
  - Indicates the size of the sliding window to use.
  - Size = 0 is valid, indicates a stop in flow control.
- Cheksum(16 bits):
  - Provided for extreme reliability.
  - Source adds all 16bit words performs 1 complement. Receiver compares its value.
- Urgent Pointer(16 bits):
  - Used in conjunction with the URG flag to indicate the receiver to process the data urgently to the application that requested it.
  - Example, a stuck telnet session. On the local host, the user presses the escape key which is transmitted urgently to the remote host.

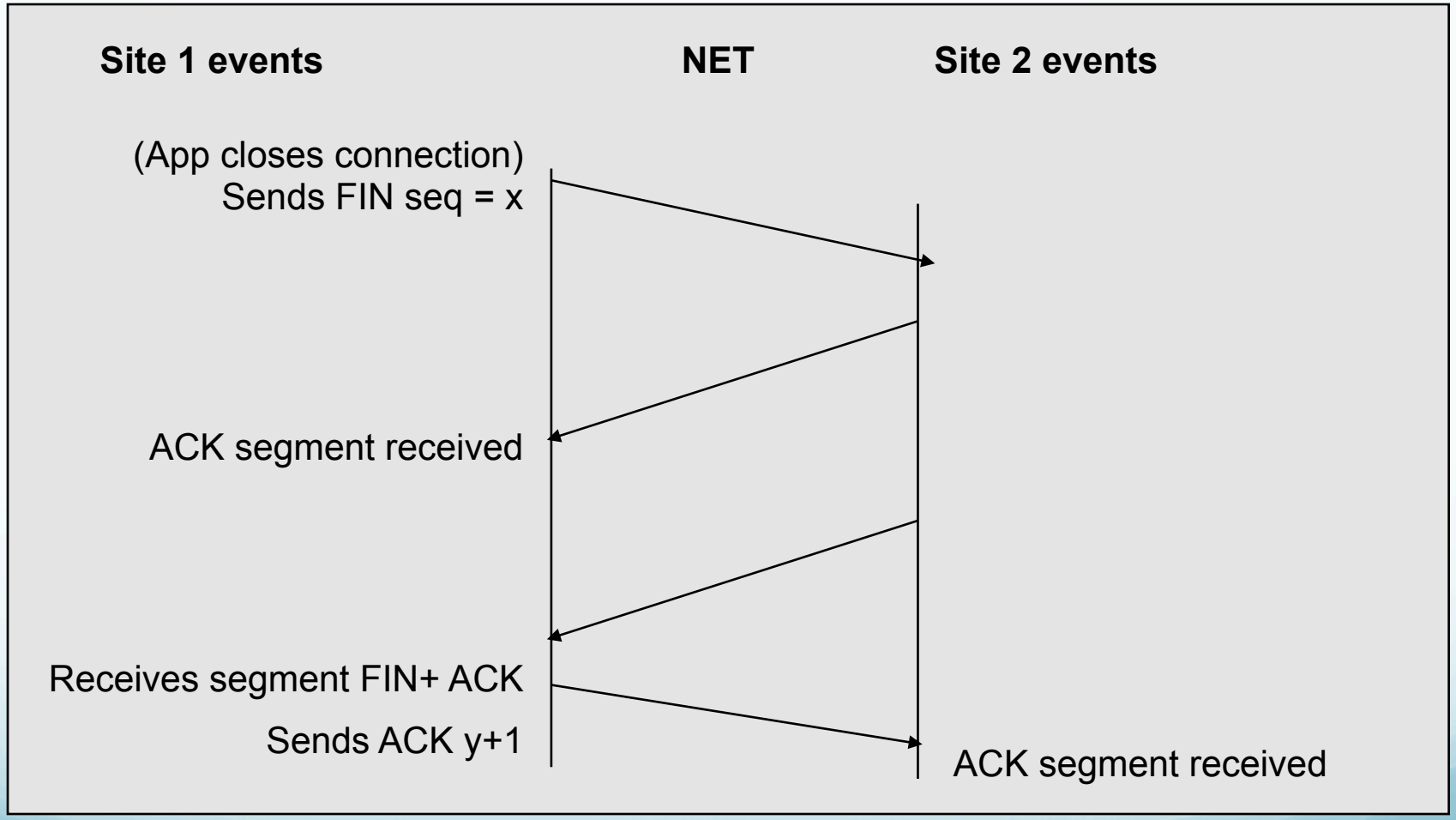
The TCP Segment Header

- Options(0 or more 32 bit words):
  - A way to add extra facilities.
  - Most important option:
    - Allow each host to specify the maximum TCP payload it is willing to accept.
    - Using large segments is more efficient than using small ones due to the 20 byte header.
    - During connection Setup each side can announce its maximum length.
    - Default 536 byte payload.

Establishing a TCP Connection



Establishing a TCP Connection



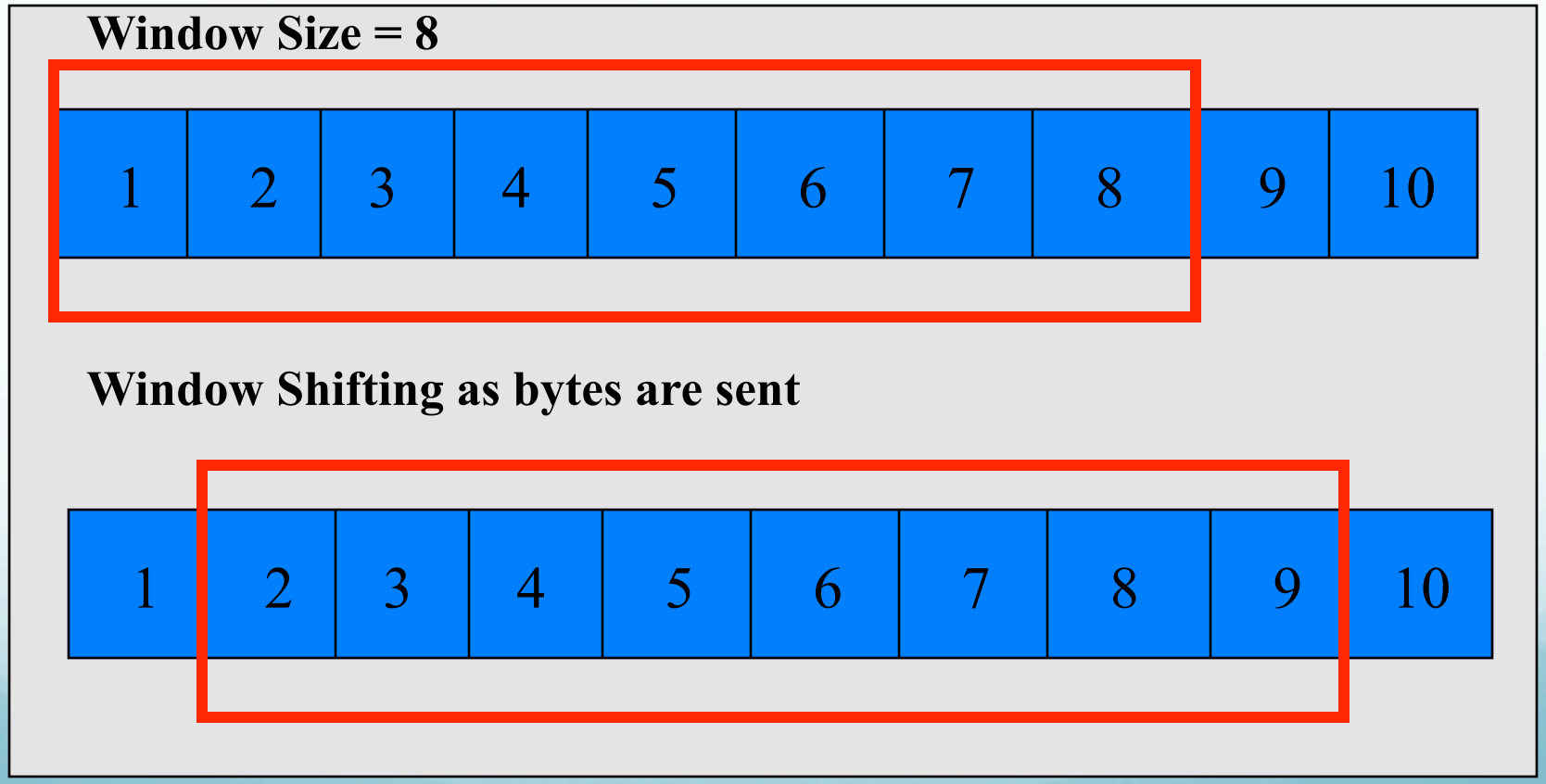
Sliding Window Protocol

- Most efficient method of Flow Control.
- Better use of the channel.
- The emitter can transmit a certain number of packets without the need for acknowledgments.
- The number of packets it can transmit is limited by the size of the window.

Sliding Window Protocol

- Window size one, works like the positive acknowledgment.
- Its a full duplex protocol.
- As the window size increases we can actually use more the capacity of the channel.
- We can reach a state of equilibrium when the emitter can transmit at the same rate as the network can process the packets.
- When we need to slow down the transmission we send a window of size cero to indicate that a hosts buffer is full.

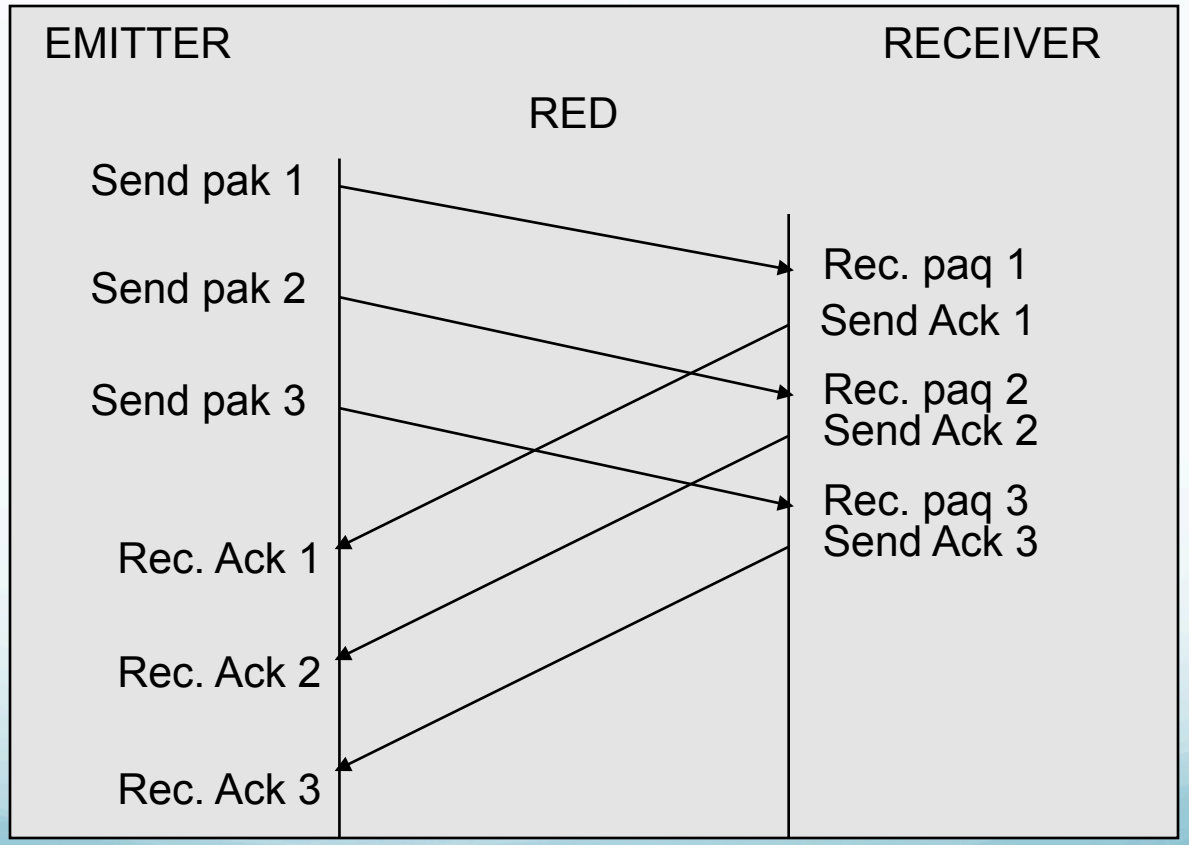
Sliding Window Protocol



Sliding Window Protocol

- The window moves or slides right as the acknowledgments of the packets start to arrive.
- The window can only move when the acknowledgments of the previous packets have arrived, even though it might get ack's of later packages.
- The packets that are in the window are the ones that are being transmitted.
- If we loose a single packet we can retransmit only that one.

Sliding Window Example size = 3



Sliding Window Protocol

- TCP treats the data flow as a sequence of bytes, which are divided into segments for transmission.
- The sliding window protocol lets us:
  - Send multiple packets without waiting for each individual acknowledgment.
  - Lets the receiver control the flow of data by changing the window size to fit its needs.
  - Flow control → Point to Point.

Sliding Window Protocol

- Important note: The window works at a byte level, not packets nor segments.
- The bytes of data are numbered in a sequential manner.
- The emitter uses 3 different pointers to delimit the window size.
- The receiver manages data in a similar fashion.

Emitter/Receiver Window Definition