TCP and UDP

Multiplexing

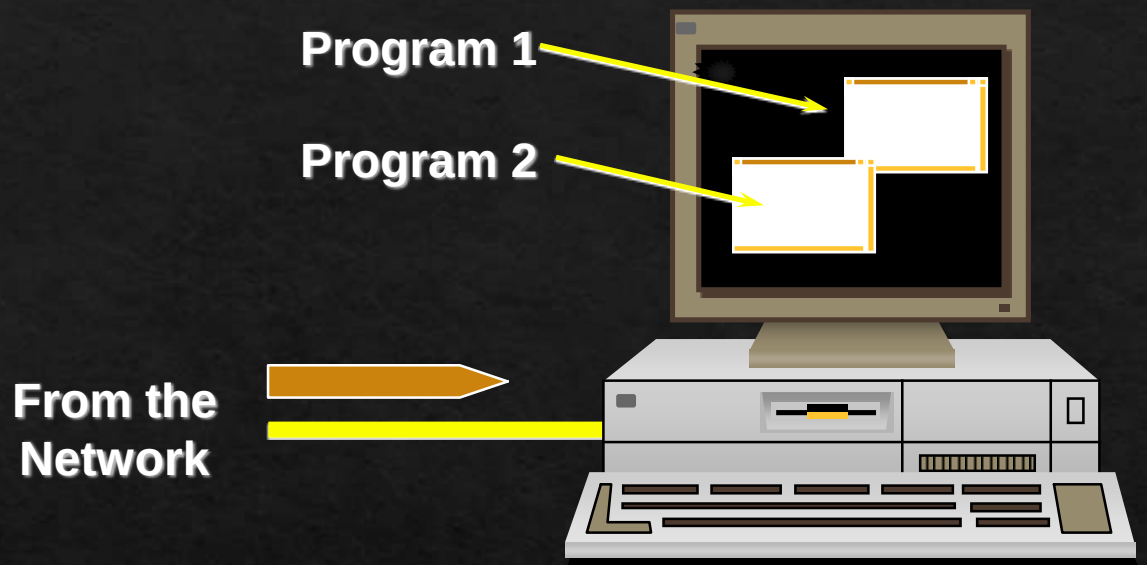
Problem: When a packet arrives at a host,

- Several network applications are running on a host.
- How to identify a specific application process.

Solution:

- Use Port numbers.
- Endpoint of a connection is defined as (host, port) pair.
- A connection is identified by

  (lhost, lport, rhost, rport).



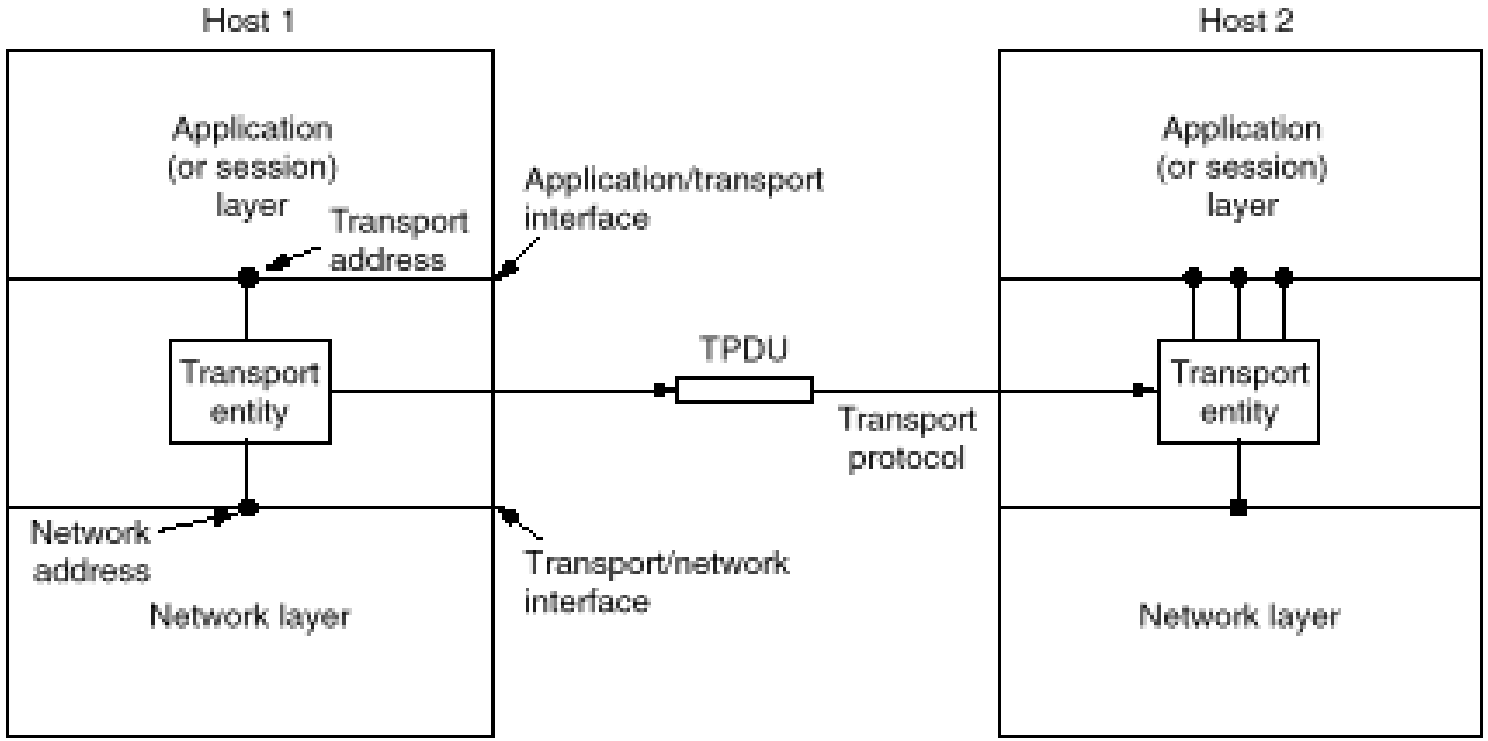
• Here we see a LAN frame heading towards a PC from the network. MAC and Network Layer addressing have got the frame this far, but now there's a problem.
• There are two possible communication programs running in the PC - Program 1 and Program 2.
• The MAC and IP addresses on the PC only identify the machine itself, not the program to which the packet should be sent.
* To differentiate between these programs, we use Transport Layer addressing.

Reserved TCP Port Numbers

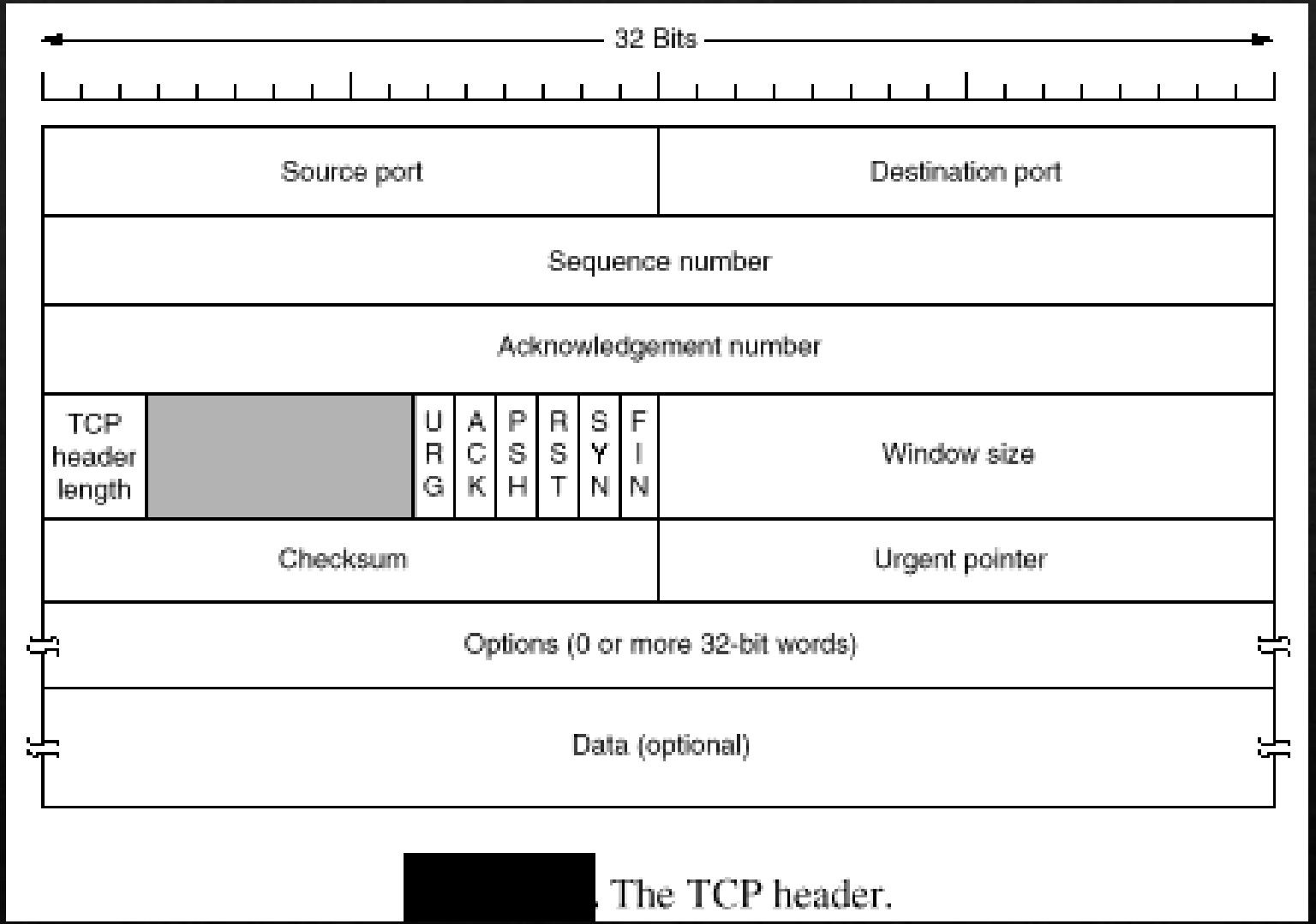
- For commonly used applications, port numbers have been reserved.For example,
  - 21 File Transfer Protocol
  - 23 Telnet
  - 25 Simple Mail Transfer Protocol
  - 53 Domain Name Service
  - 79 Finger
  - 119 Network News Transfer Protocol

What is TCP

- Transport layer protocol for data communication.
- Properties:
  - Reliable data transfer
  - Virtual circuit connection
  - Full duplex connection
  - Unstructured stream
  - Buffered transfer
  - Multiplexing
  - Piggybacking control information



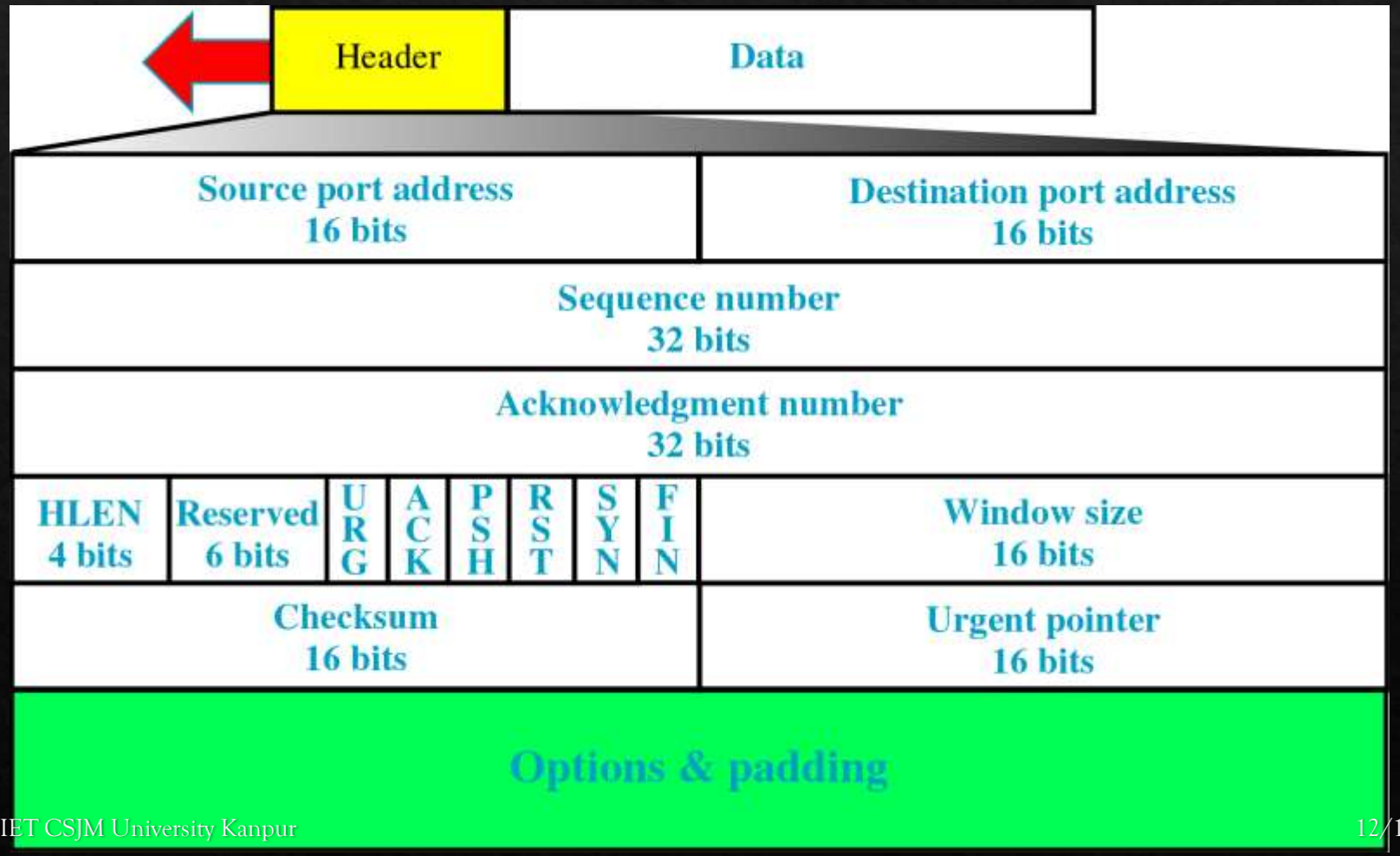
The network, transport, and application layers.



. The TCP header.

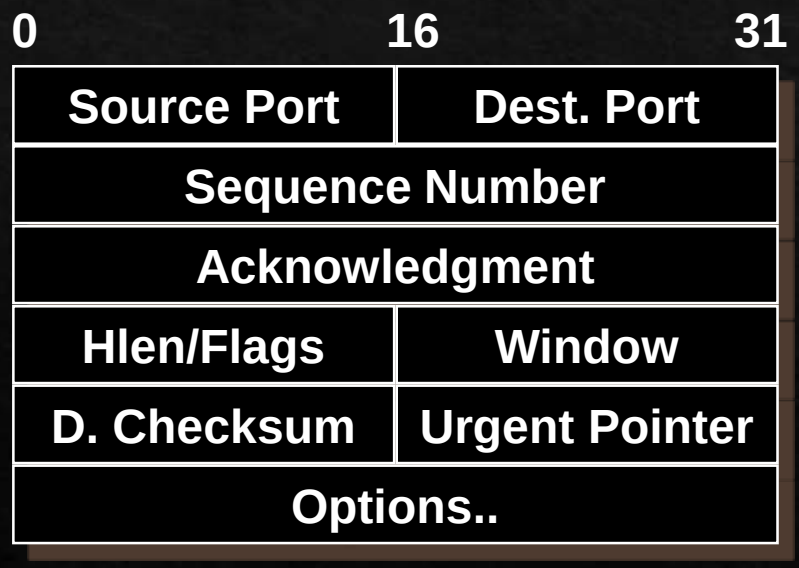
Figure 24-16

TCP Segment Format



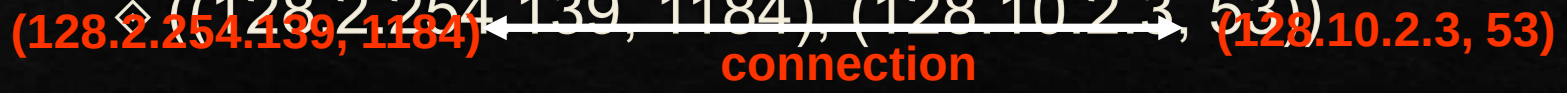
TCP functions

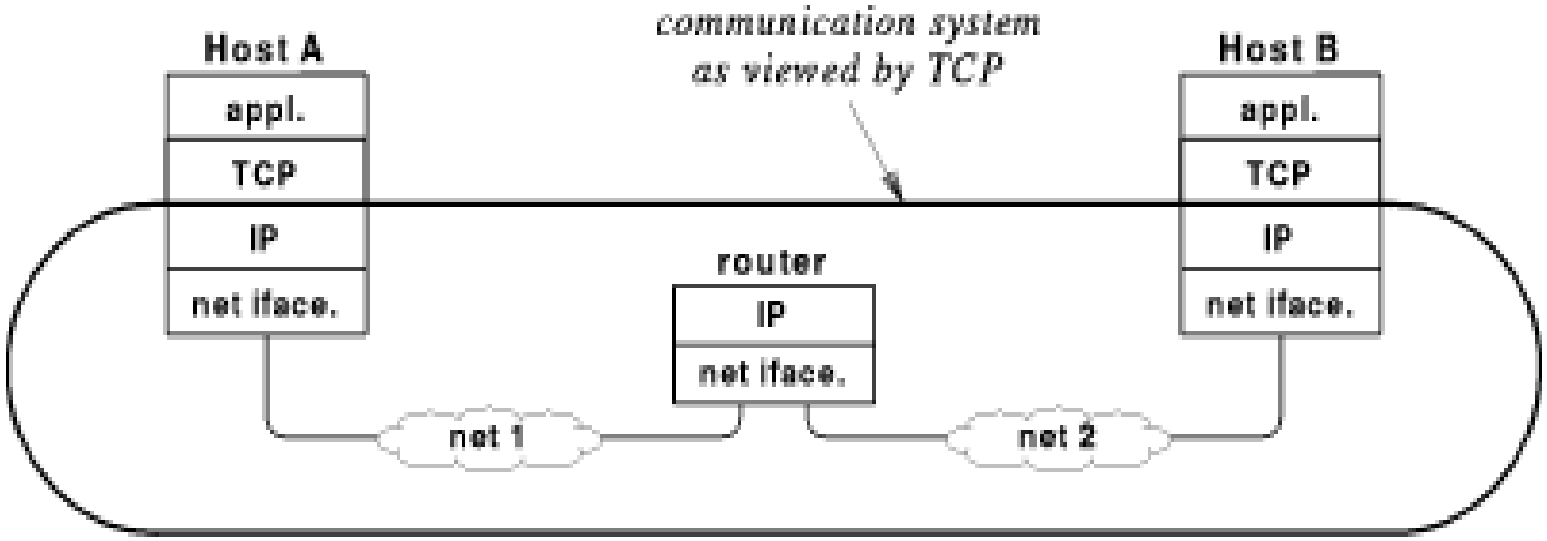
- Connections
- Sequence numbers
- Sliding window protocol
- Reliability and congestion control.

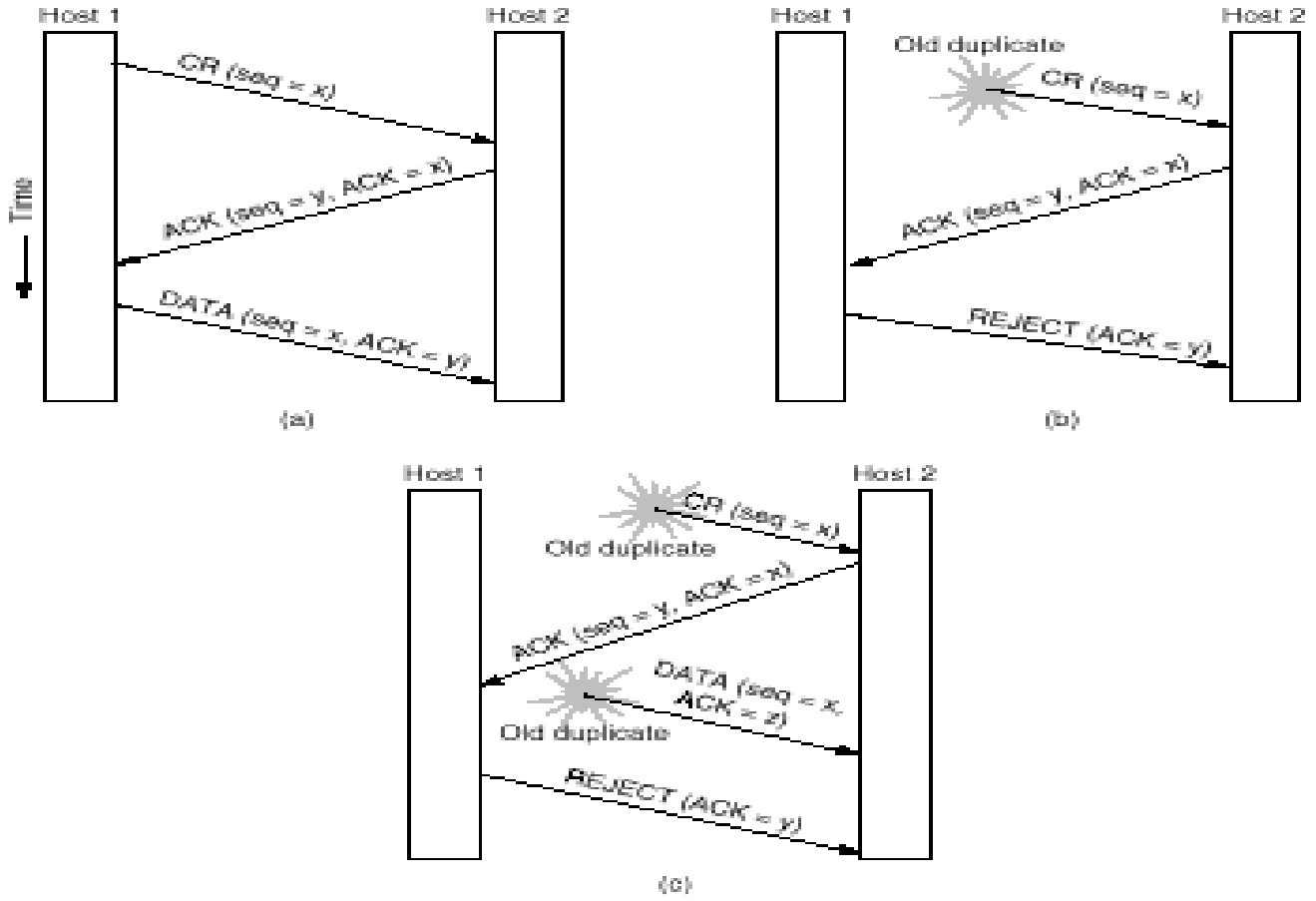


Connections

- Connection is a fundamental TCP communication abstraction.
  - data sent along a connection arrives in order
  - implies allocation of resources (buffers) on hosts
- The endpoint of a connection is a pair of integers:
  - (IP address, port)
- A connection is defined by a pair of endpoints:
  - ((128.2.254.139, 1184), (128.10.2.3, 53))



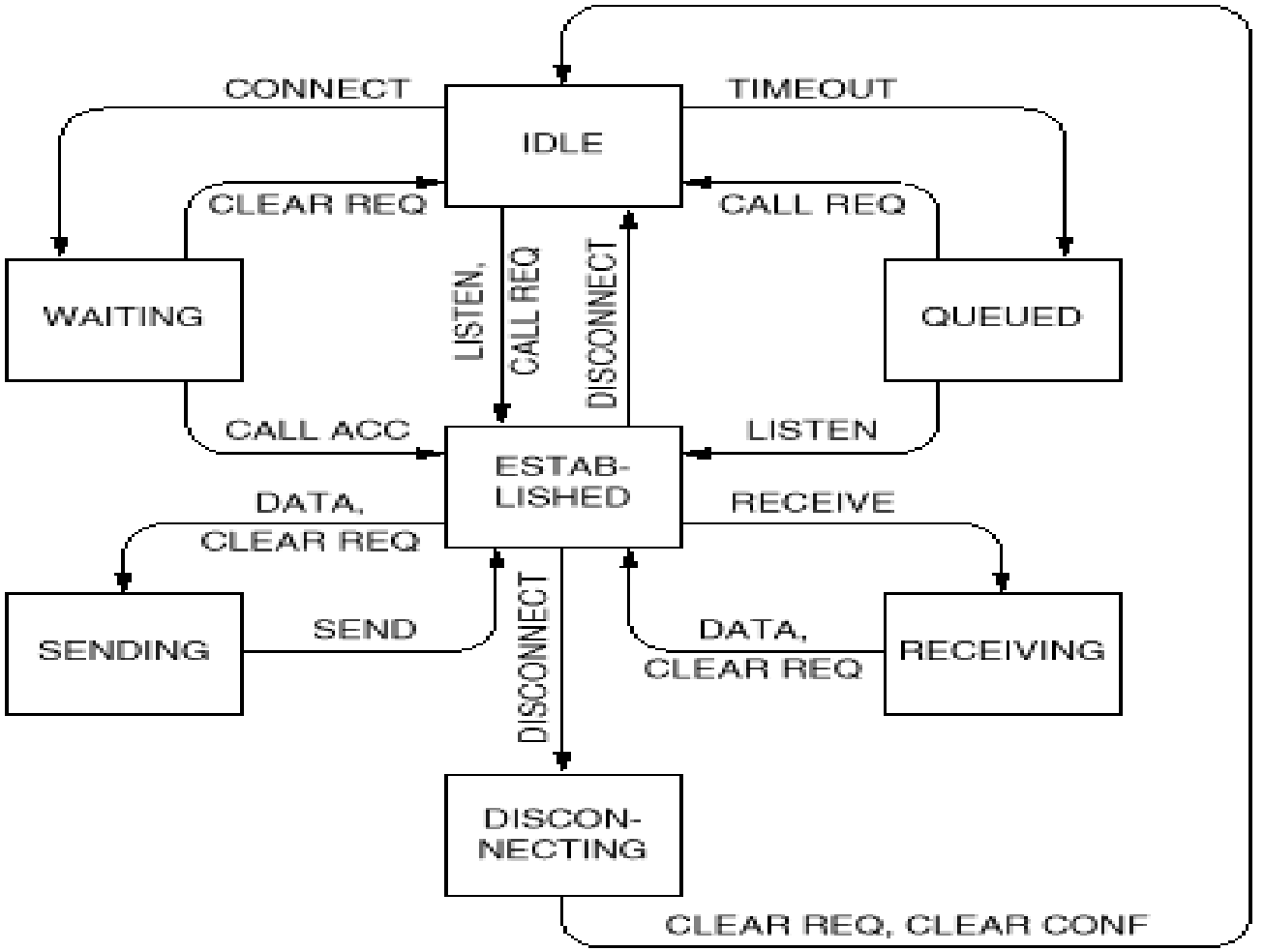

communication system
as viewed by TCP
Host A
appl.
TCP
IP
net iface.
router
IP
net iface.
Host B
appl.
TCP
IP
net iface.
net 1
net 2



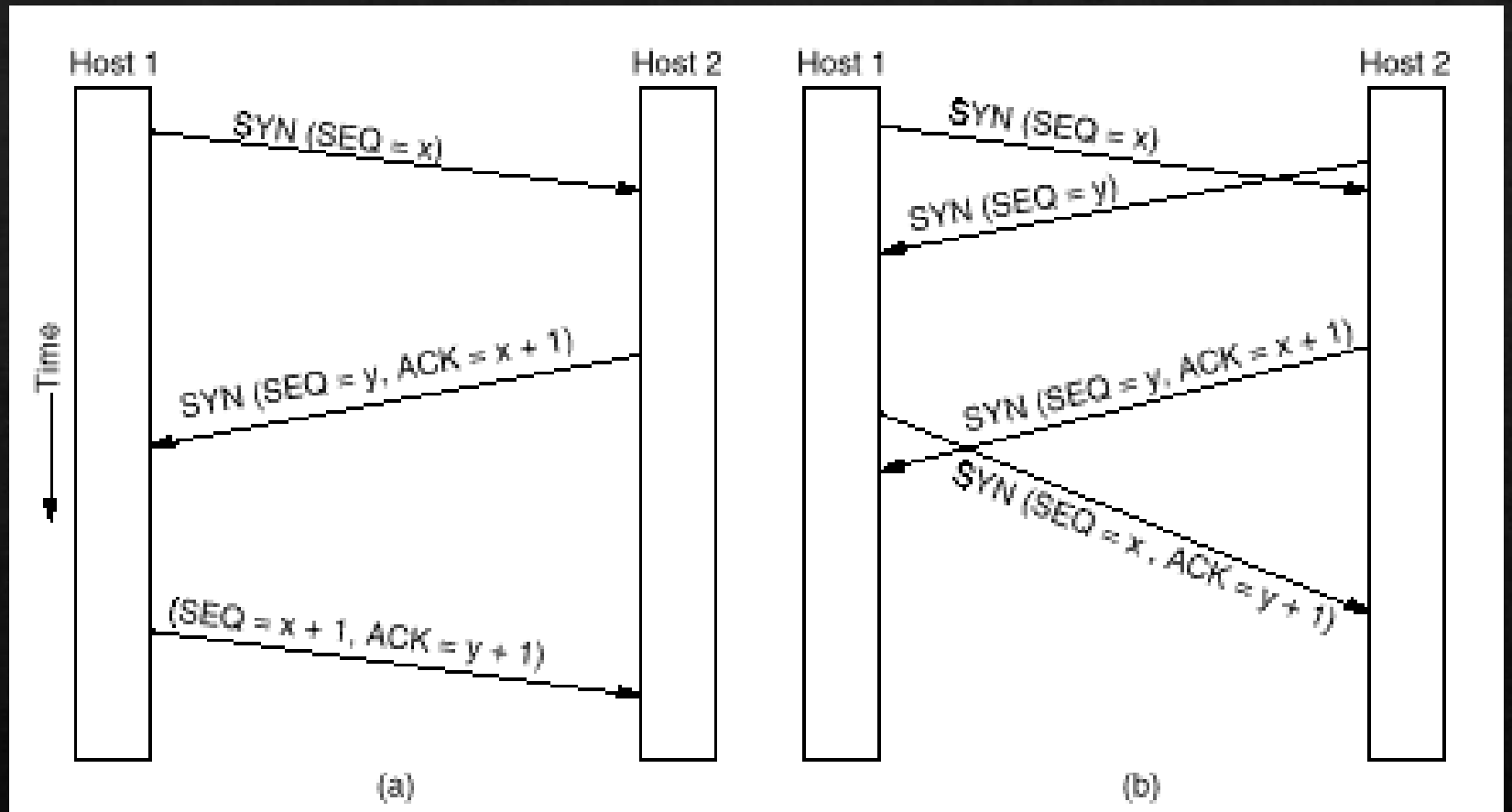
[redacted] Three protocol scenarios for establishing a connection using a three-way handshake. CR and ACC denote CONNECTION REQUEST and CONNECTION ACCEPTED, respectively. (a) Normal operation. (b) Old duplicate CONNECTION REQUEST appearing out of nowhere. (c) Duplicate CONNECTION REQUEST and duplicate ACK.

Primitive	Meaning
SOCKET	Create a new communication end point
BIND	Attach a local address to a socket
LISTEN	Announce willingness to accept connections; give queue size
ACCEPT	Block the caller until a connection attempt arrives
CONNECT	Actively attempt to establish a connection
SEND	Send some data over the connection
RECEIVE	Receive some data from the connection
CLOSE	Release the connection

The socket primitives for TCP.



[redacted] The example protocol in graphical form. Transitions that leave the connection state unchanged have been omitted for simplicity.



[redacted](a) TCP connection establishment in the normal case. (b) Call collision.

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for ACK
SYN SENT	The application has started to open a connection
ESTABLISHED	The normal data transfer state
FIN WAIT 1	The application has said it is finished
FIN WAIT 2	The other side has agreed to release
TIMED WAIT	Wait for all packets to die off
CLOSING	Both sides have tried to close simultaneously
CLOSE WAIT	The other side has initiated a release
LAST ACK	Wait for all packets to die off

[redacted] The states used in the TCP connection management finite state machine.

Sequence space

- Each stream split into a sequence of segments which are encapsulated in IP datagrams.
- Each byte in the byte stream is numbered.
  - 32 bit value
  - wraps around
  - initial values selected at runtime
- Each segment has a sequence number.
  - indicates the sequence number of its first byte
  - Detects lost, duplicate or out of order segments

TCP flow control mechanism: sliding window

- The purpose of *flow control* is to keep senders from flooding receivers with packets and filling up their memories.
- Often confused with *congestion control,* which tries to keep the senders from flooding the network with packets.

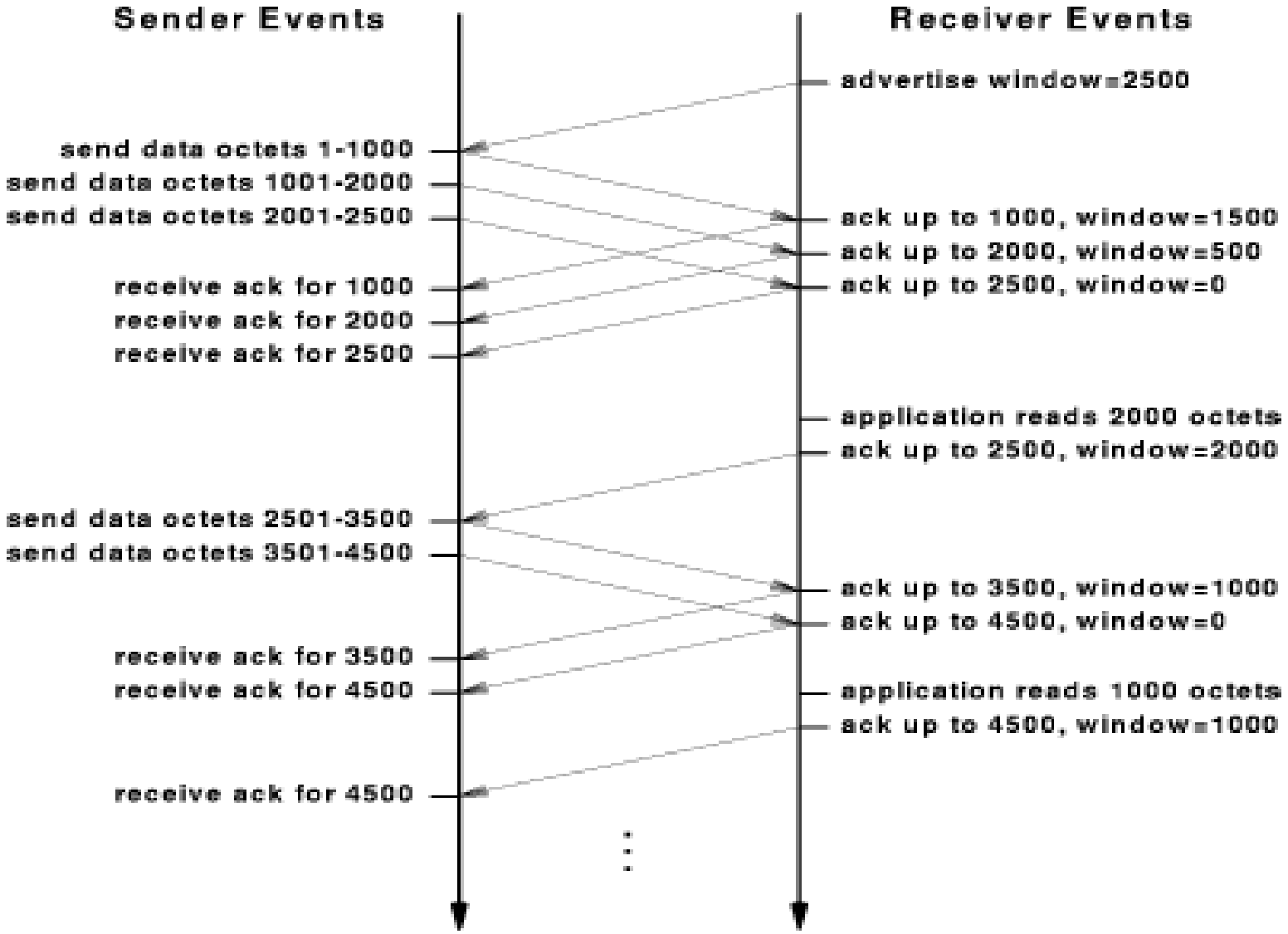
Sliding window protocol (sender)

- Sender maintains a “window” of unacknowledged bytes that it is allowed to send, and a pointer to the last byte it sent:

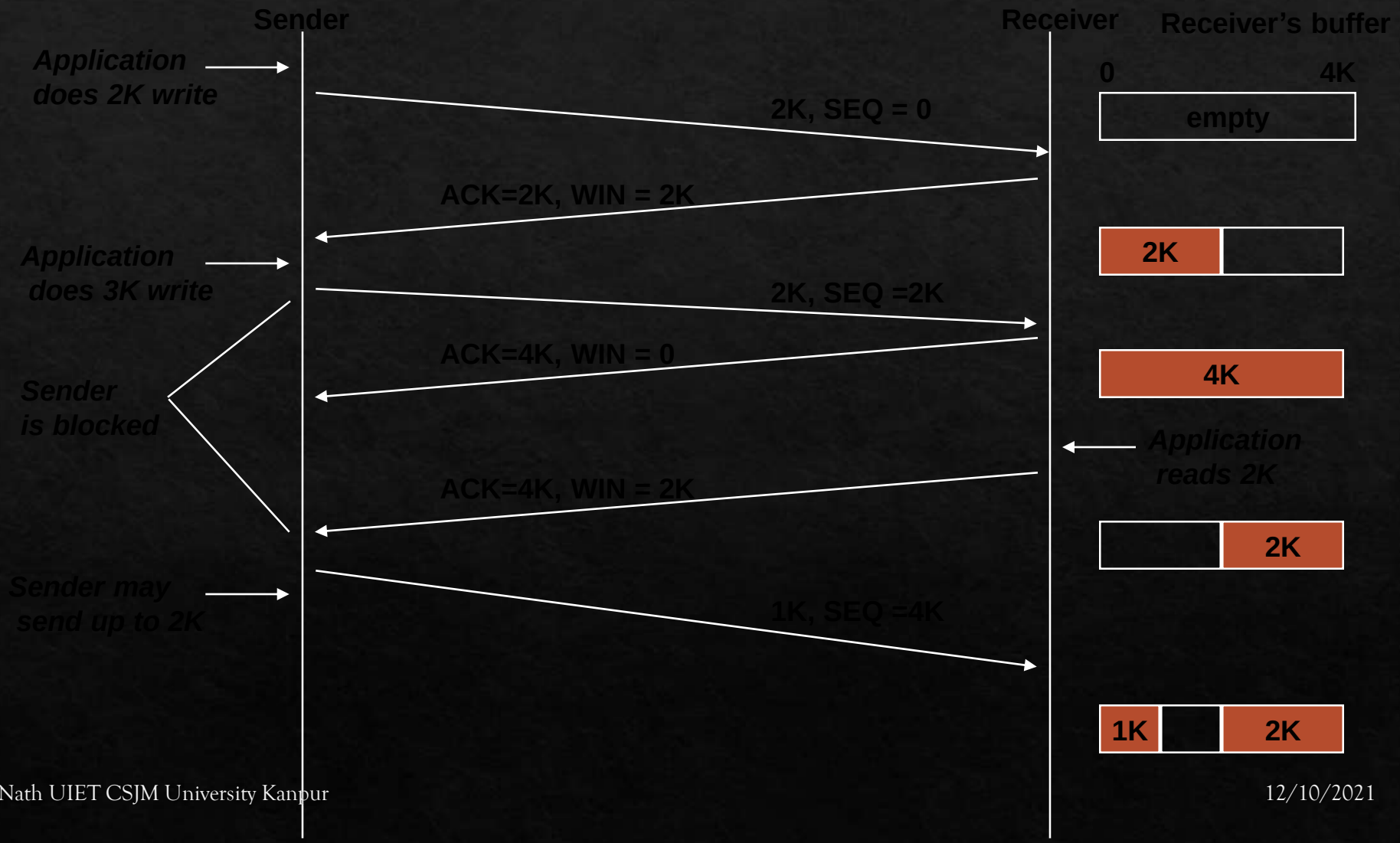
Bytes through 2 have been sent and acknowledged (and thus can be discarded)
Bytes 3 -- 6 have been sent but not acknowledged (and thus must be buffered)
Bytes 7 -- 9 have been not been sent but will be sent without delay.
Bytes 10 and higher cannot be sent until the right edge of window moves.

Sliding window protocol (receiver)

- Receiver acknowledges receipt of a segment with two pieces of information:
  - ACK: the sequence number of the next byte in the contiguous stream it has already received
  - WIN: amount of available buffer space.
- ACK indicates that data was received correctly.
  - sender can increment left edge of window
  - sender can delete data to the left of the window.
- WIN indicates that more buffer space was freed up.
  - sender can increment the right edge of its window
  - sender can transmit more data.


Sender Events
Receiver Events
advertise window=2500
send data octets 1-1000
send data octets 1001-2000
send data octets 2001-2500
ack up to 1000, window=1500
ack up to 2000, window=500
ack up to 2500, window=0
receive ack for 1000
receive ack for 2000
receive ack for 2500
application reads 2000 octets
ack up to 2500, window=2000
send data octets 2501-3500
send data octets 3501-4500
ack up to 3500, window=1000
ack up to 4500, window=0
receive ack for 3500
receive ack for 4500
application reads 1000 octets
ack up to 4500, window=1000
receive ack for 4500

Sliding window protocol (example)



Reliability and congestion control

- **Reliability:**
  - **sender**
    - saves segments inside its window
    - uses timeouts and sequence numbers in ACKS to detect lost segments.
    - retransmit segments it thinks are lost
  - **receiver**
    - uses sequence numbers to assemble segments in order
    - also to detect duplicate segments (how might this happen?)
- **Congestion control**
  - sender maintains separate separate congestion window
  - uses smaller of the two windows
  - uses "slow start" algorithm to adaptively set congestion window size

End-to-End data issues

- **Presentation formatting**
  - must account for different data formats on different machines
    - different byte orders
    - different word sizes
- **Compression**
  - data can be compressed/decompressed on the endpoints to save network bandwidth (beyond our scope)
- **Encryption**
  - sensitive data can be encrypted/unencrypted on the endpoints.
- **Authentication**
  - Receivers may want to verify that messages really do come from the sender.

Reliability

Networks can

- deliver packets out of order
- lose packets
- deliver duplicates
- corrupt packets due to noise

TCP provides protection against above errors.

Client-Server Model

- **Passive Open**

  Application indicating to OS that it wants to accept incoming connections.

- **Active Open**

  Application asking OS to try to establish connection with a peer entity.

- **Client:** Application that does Active Open.
- **Server:** Application that does Passive Open.

UDP: User datagram protocol

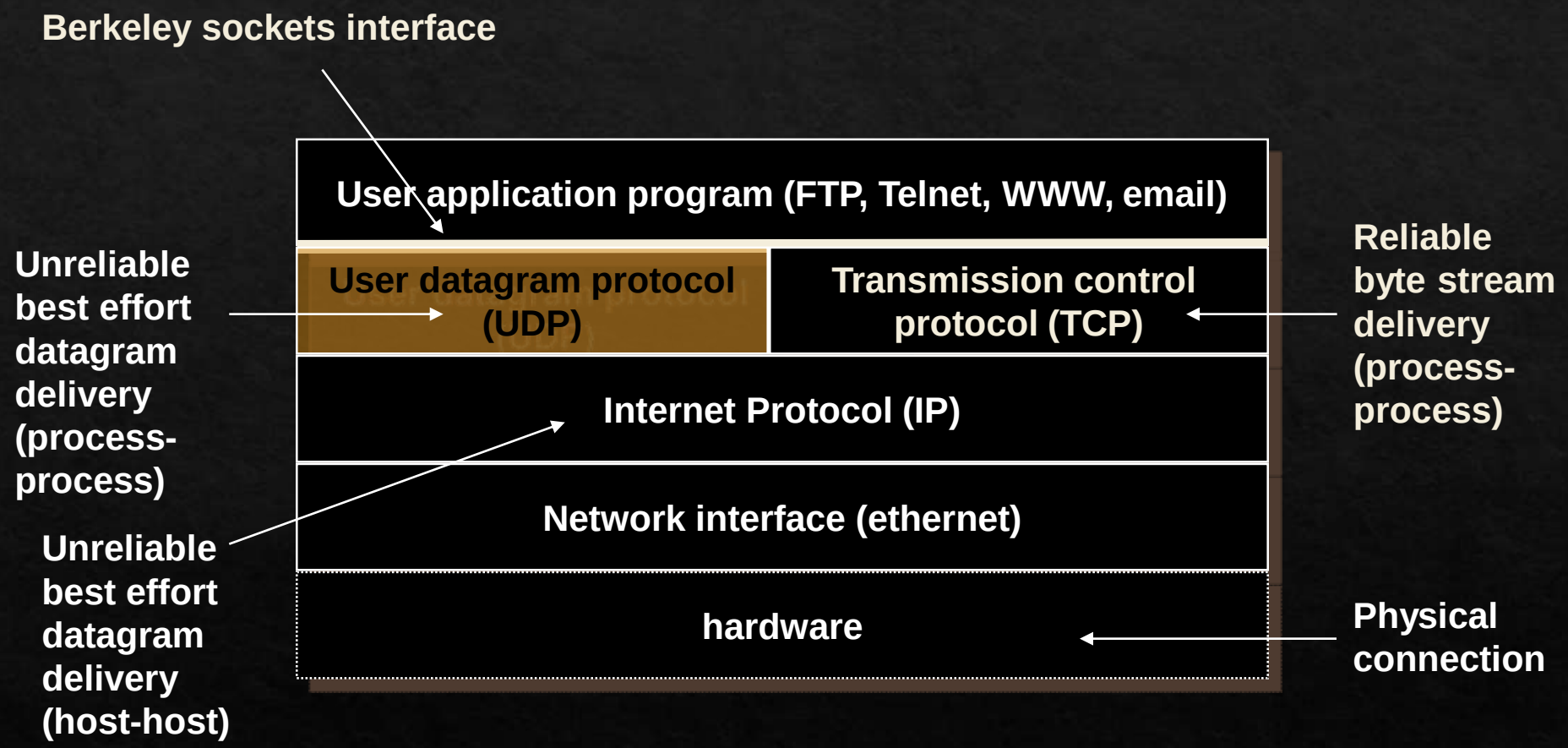
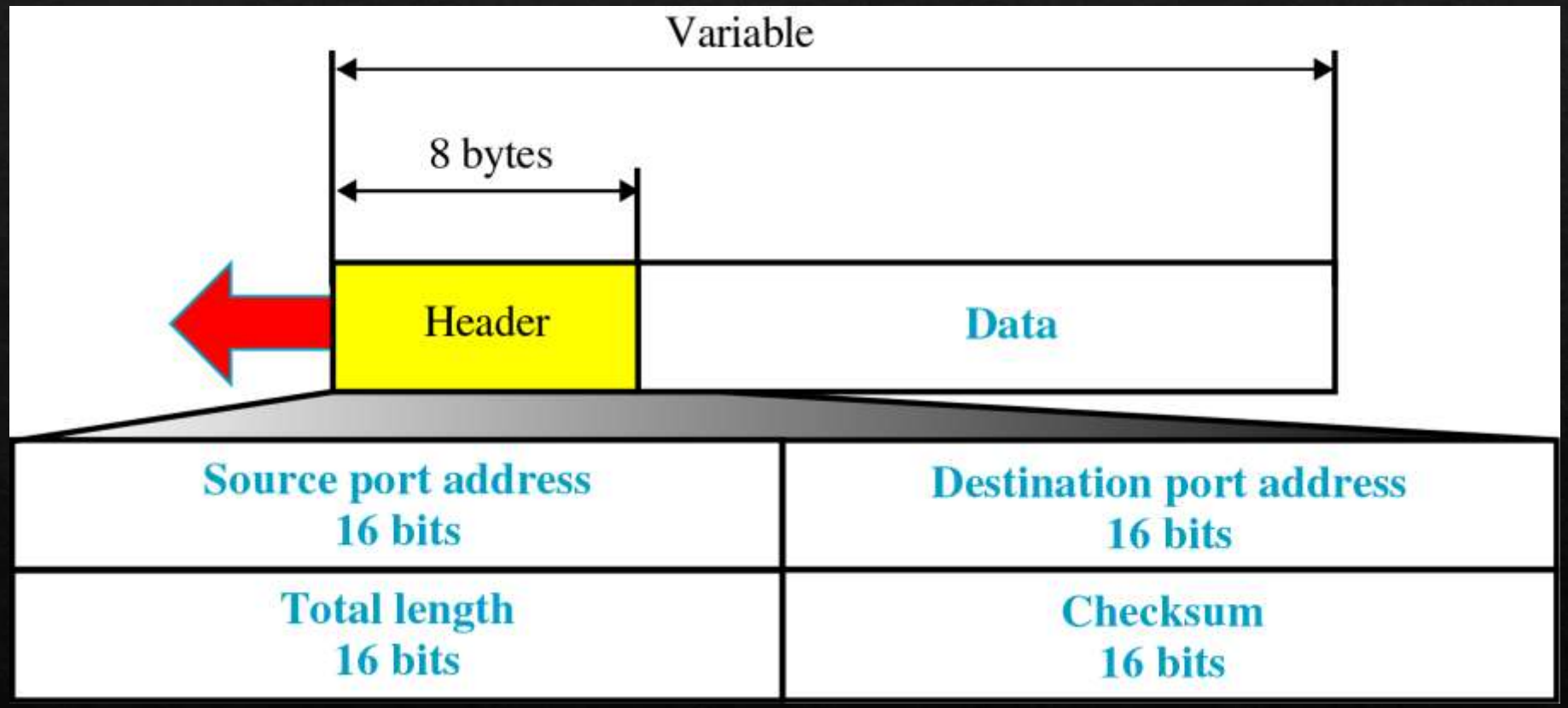


Figure 24-15

UDP Datagram Format

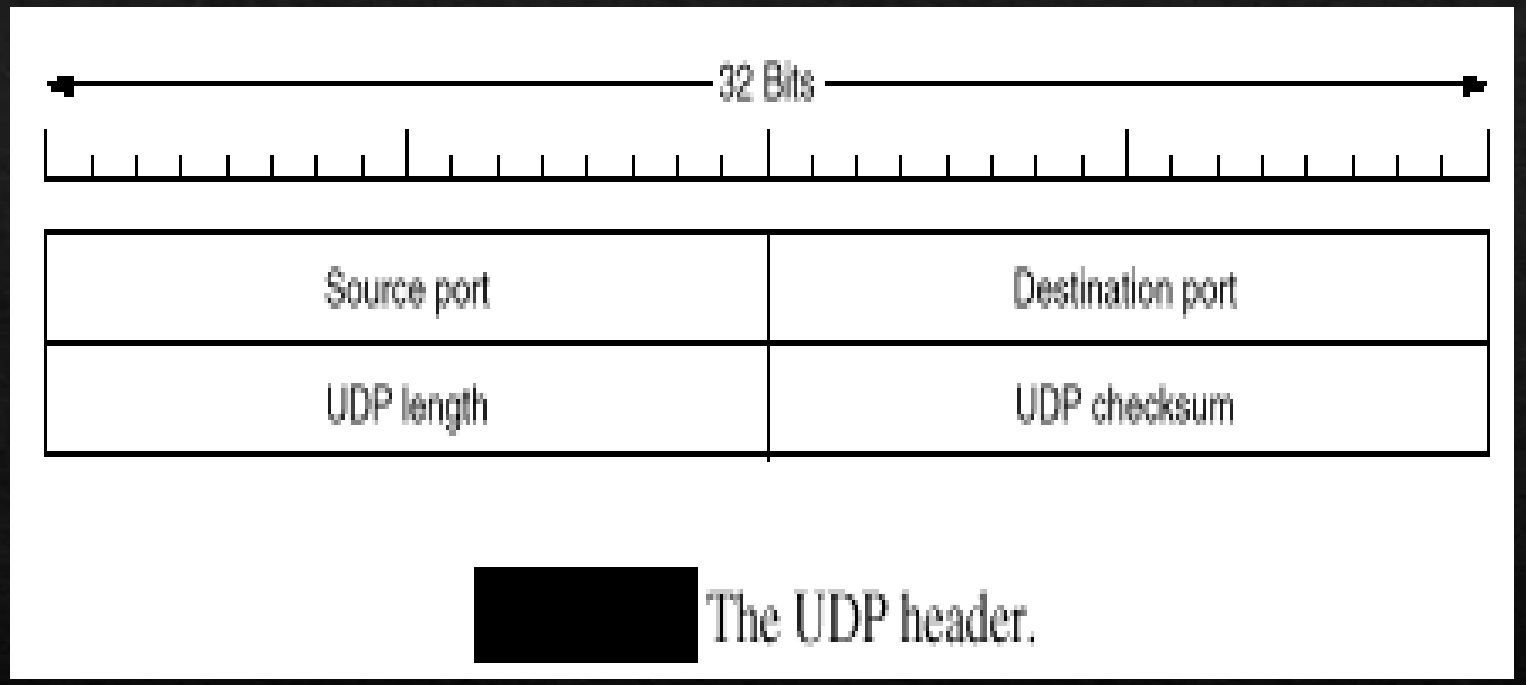


UDP: User datagram protocol

- **Extends IP to provide *process-to-process* (end-to-end) datagram delivery**
- **Mechanism for demultiplexing IP packets**
- **Based on port abstraction**
- **Process identified by <host, port> pair.**

User Datagram Protocol

- Provides unreliable, connectionless, end-to-end delivery service using IP.
- Adds ability to distinguish among multiple destinations within a given host.
- Optional checksum to ensure integrity of data.



The UDP header.

Port Numbers

- Similar to TCP ports.
- 16-bit identifier to select process attached to a communication.
- Port numbers 0-1023 are reserved for use by system.
- Well known ports (0-511) are assigned to important applications.For example,

  53 -- Domain name server

  123 -- Network time protocol

  161 -- SNMP server
- Other applications can bind a port on demand.

Title Lorem Ipsum

01

LOREM IPSUM DOLOR SIT AMET, CONSECTETUER ADIPISCING ELIT. MAECENAS

02

NUNC VIVERRA IMPERDIET ENIM. FUSCE EST. VIVAMUS A TELLUS.

03

PELLENTESQUE HABITANT MORBI TRISTIQUE SENECTUS ET NETUS ET MALESUADA FAMES.