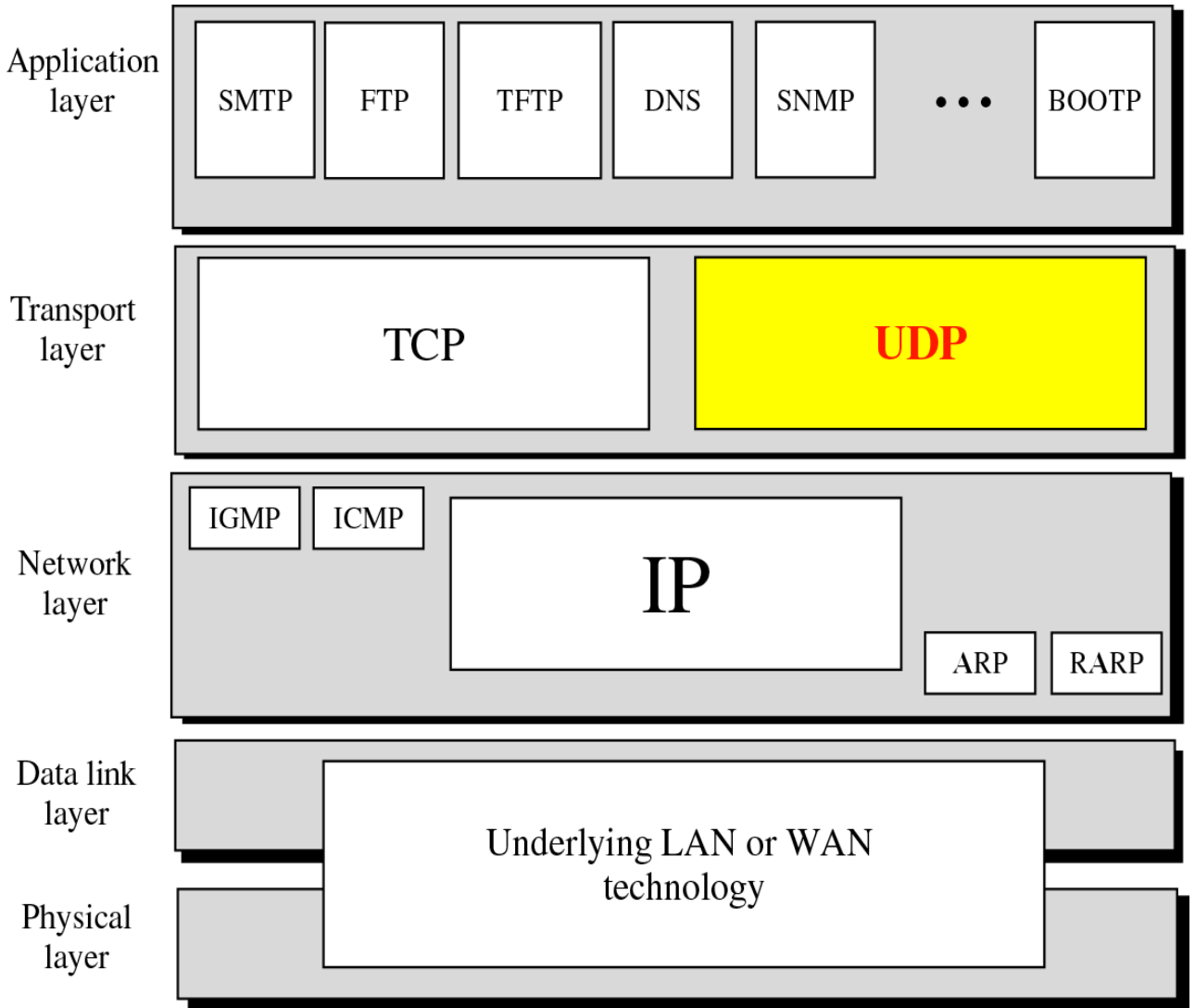
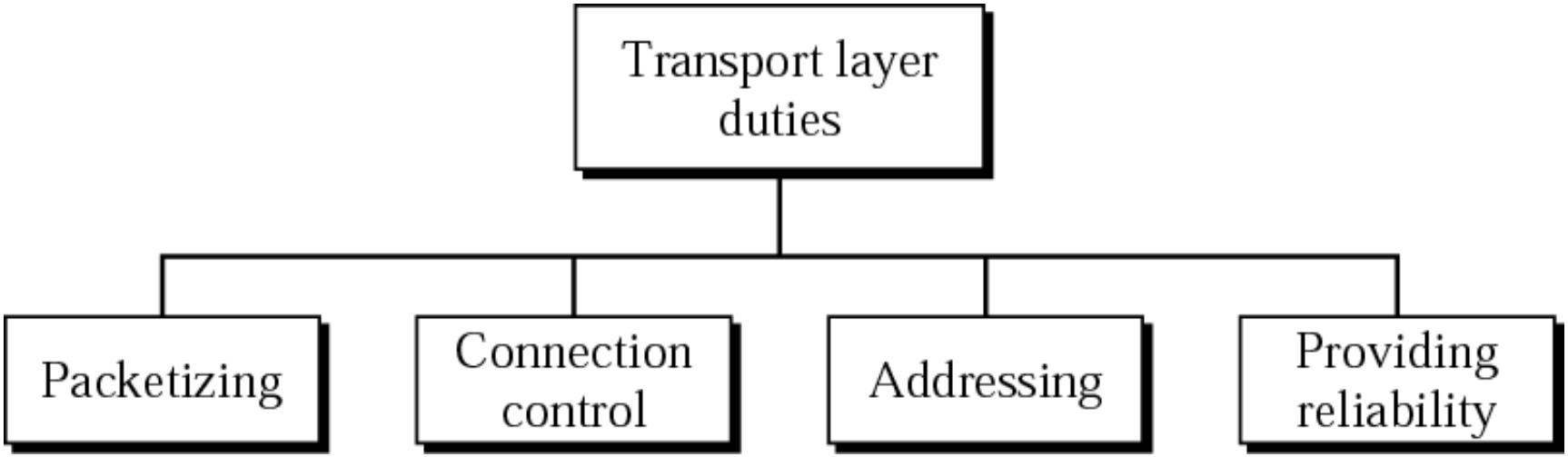
Module - 6
Transport Layer

Position of TCP & UDP in the TCP/IP protocol suite



Transport Layer Duties



Packetizing

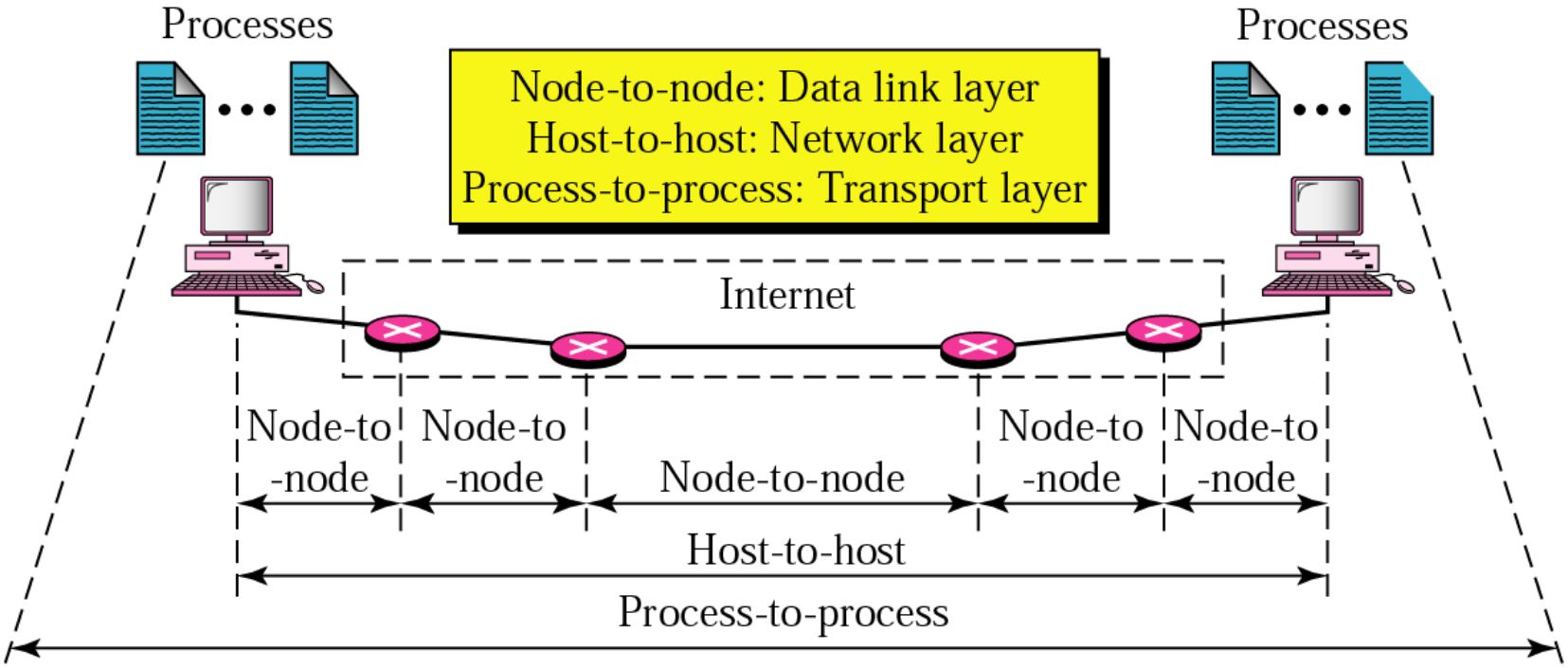
- Dividing large messages
- Adding a header

Connection Control

- Connection-oriented delivery
  - first establishing a connection (a virtual path) between the sender and receiver
- Connectionless delivery
  - treating each packet independently without any connection between the sender and receiver

Process-to-Process Delivery

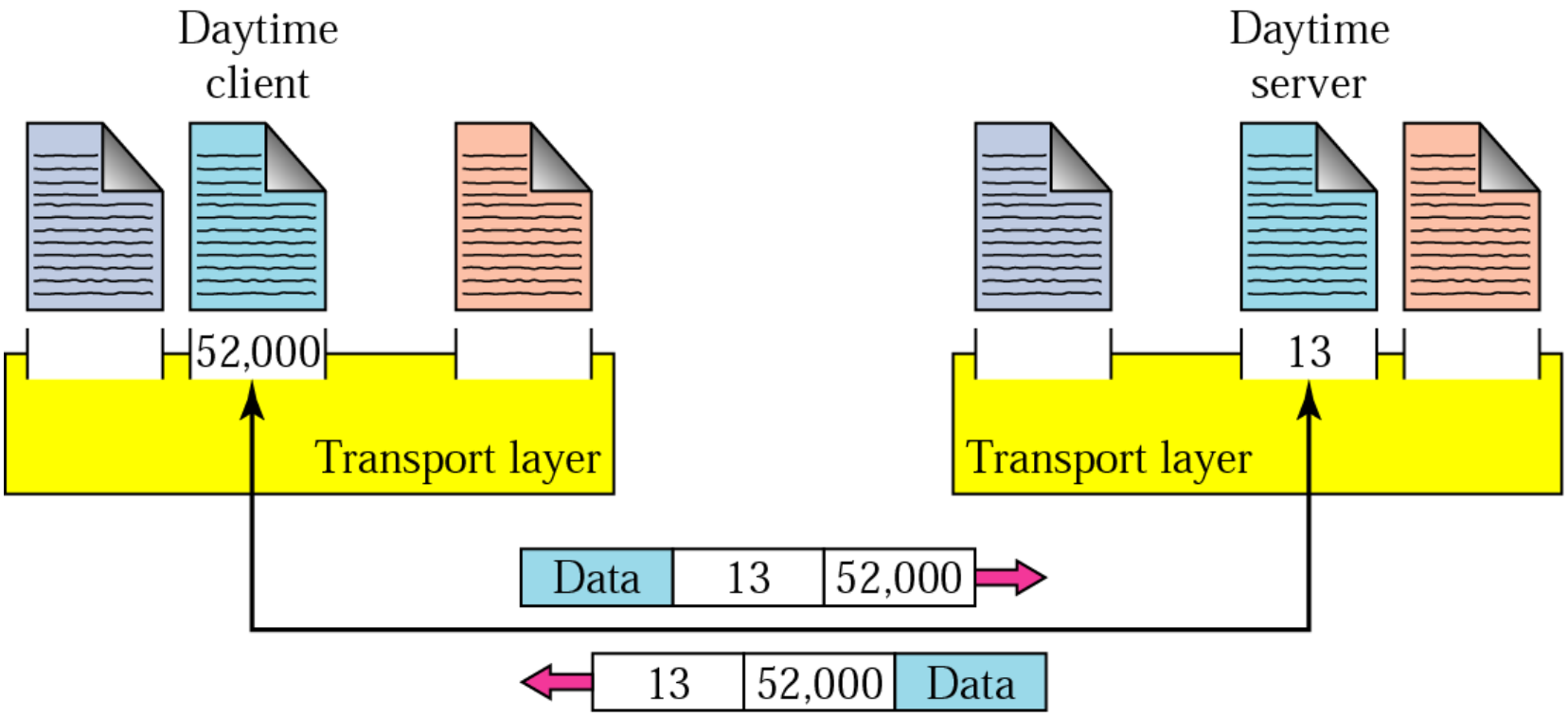
- Transport layer is responsible for process-to-process delivery, the delivery of a packet, part of a message, from one process to another.
- Two processes communicate in a client-



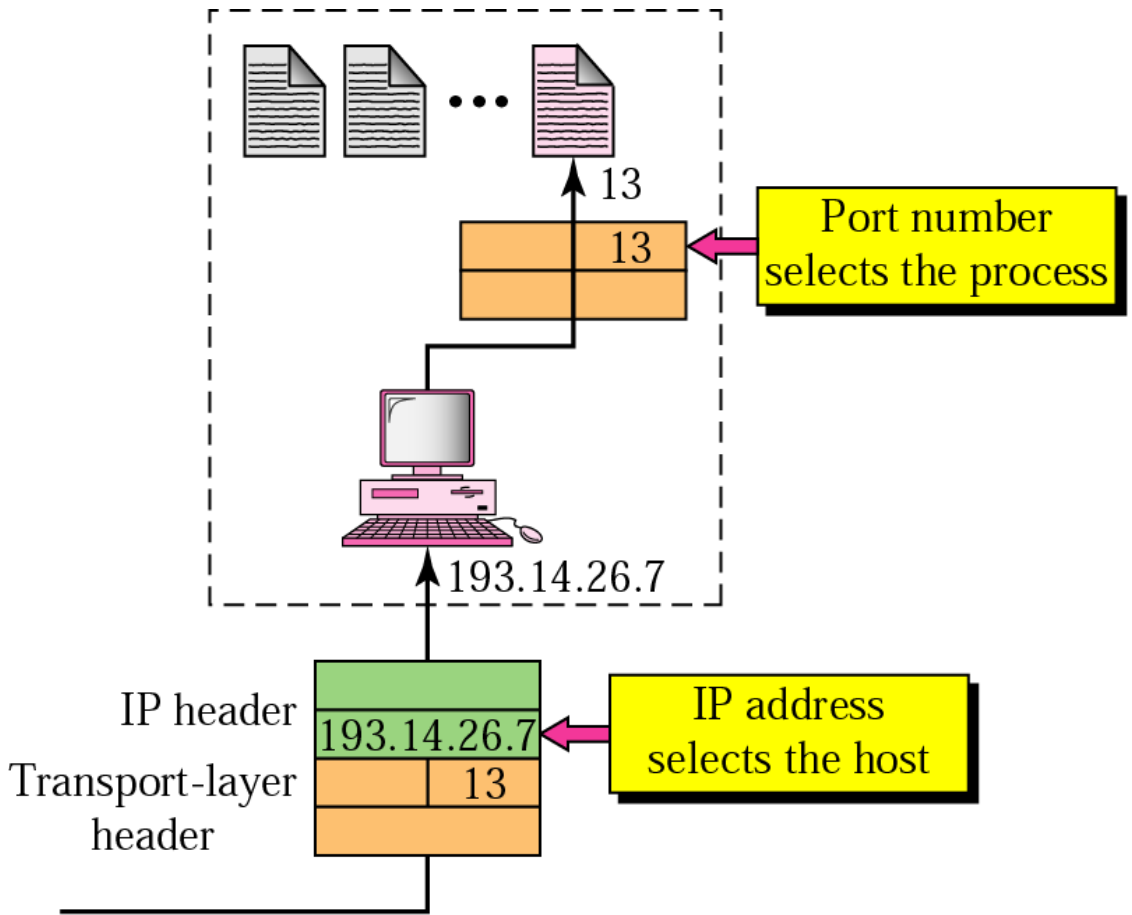
Client-Server Paradigm

- A process on the local host, called a client, needs services from a process usually on the remote host, called a server
- For communication between processes, we must define the following :
  1. Local host
  2. Local process
  3. Remote host
  4. Remote process

Addressing

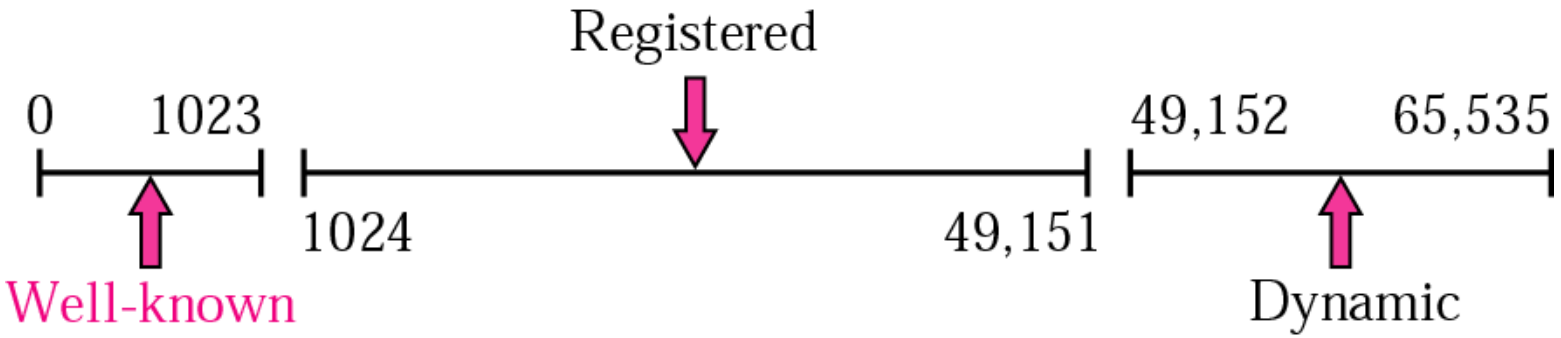

Daytime client
Daytime server
52,000
13
Transport layer
Transport layer
Data
13
52,000
13
52,000
Data

Addressing (cont'd)

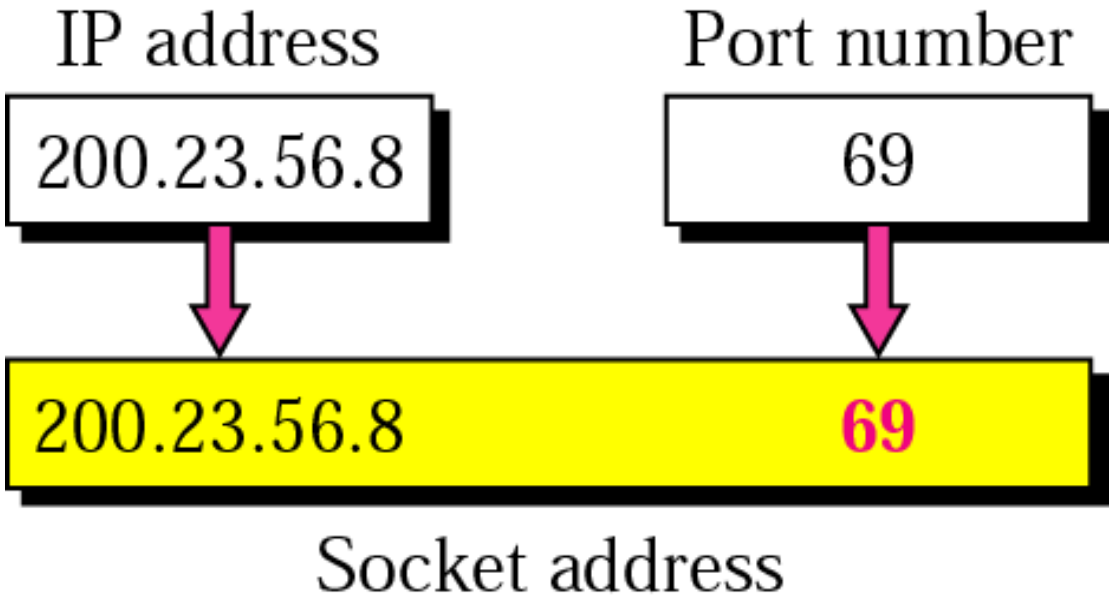


IANA Ranges

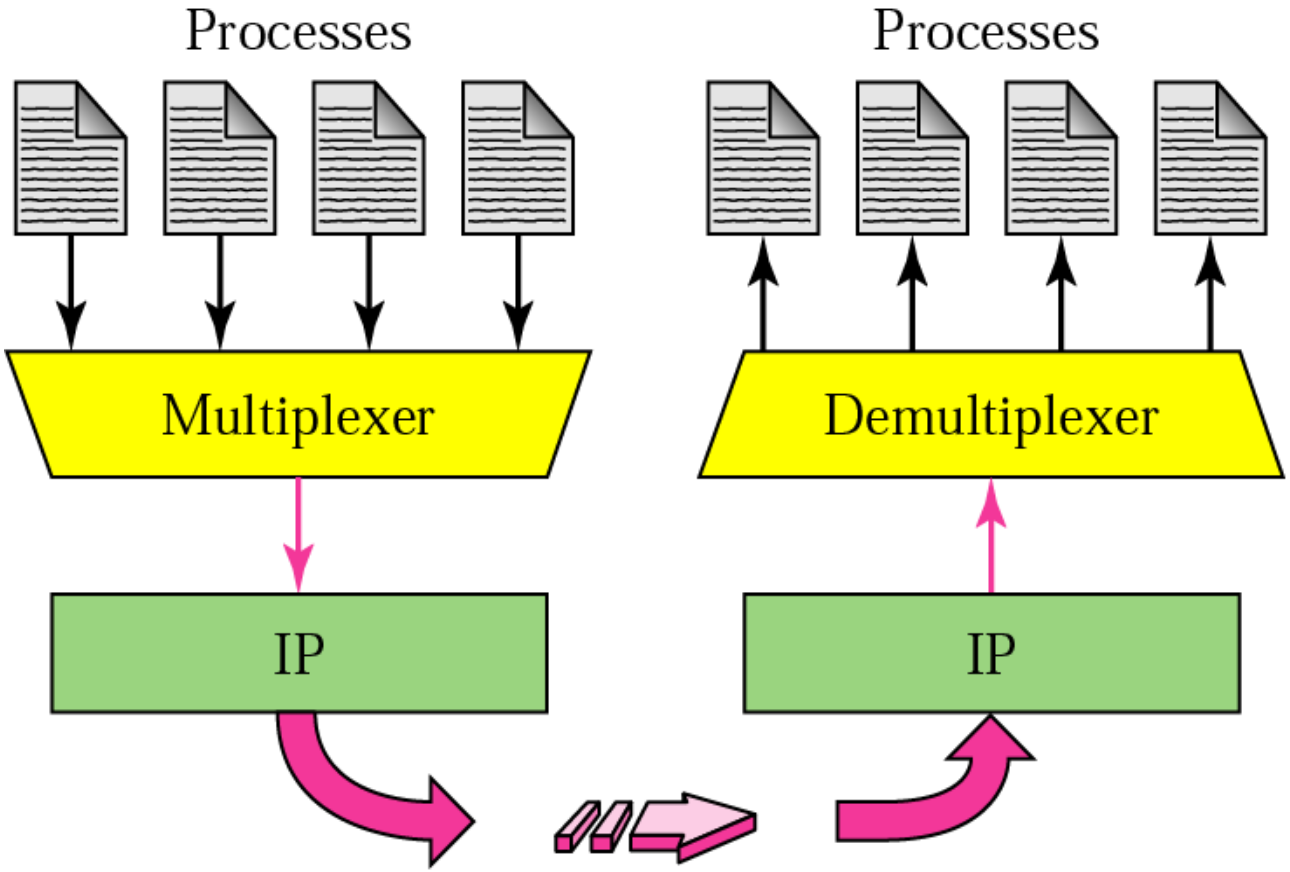
- IANA (Internet Assigned Number Authority) had divided the port numbers into three ranges :



Socket address



Multiplexing and Demultiplexing



22.3 Transmission Control Protocol (TCP)

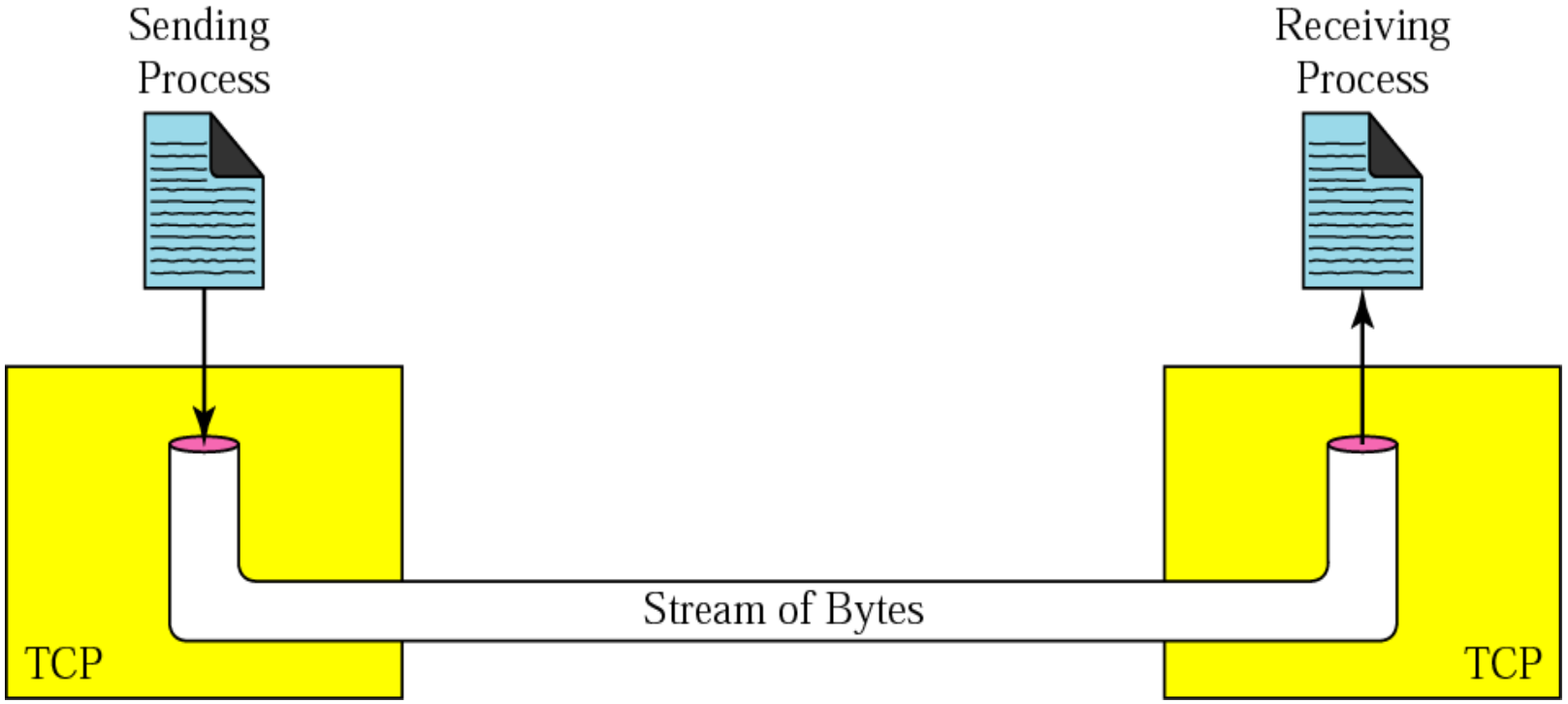
- is called a stream connection-oriented and reliable transport protocol.
- creates a process-to-process communication
  - using port numbers
- provides a flow-and-error control mechanism at the transport level
  - using sliding window protocol to achieve error control.
  - using the acknowledgment packet, time-out, and retransmission to achieve error control.
- providing a connection mechanism for the application program

Well-known Port Numbers

Port	Protocol	Description
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
20	FTP, Data	File Transfer Protocol (data connection)
21	FTP, Control	File Transfer Protocol (control connection)
23	TELNET	Terminal Network
25	SMTP	Simple Mail Transfer Protocol
53	DNS	Domain Name Server
67	BOOTP	Bootstrap Protocol
79	Finger	Finger
80	HTTP	Hypertext Transfer Protocol
111	RPC	Remote Procedure Call

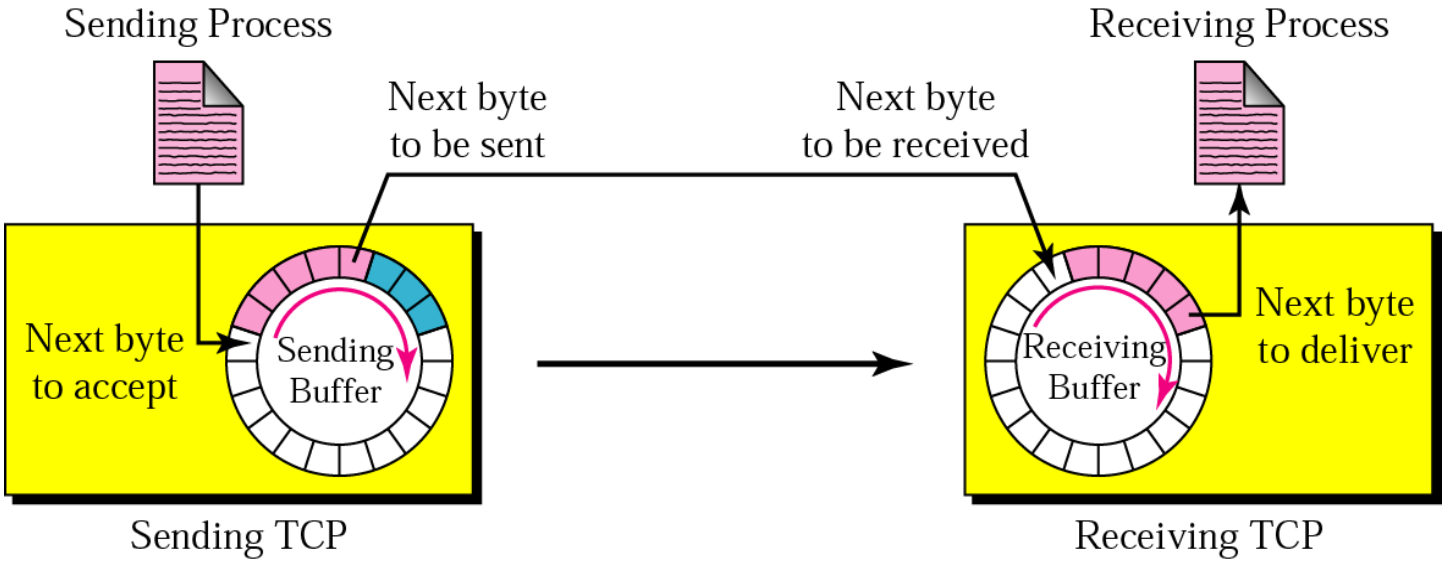
TCP Services

- Stream Delivery Service
  - TCP is a stream-oriented protocol
  - TCP creates an environment in which the two processes seem to be connected by an imaginary "tube" that carries their data across



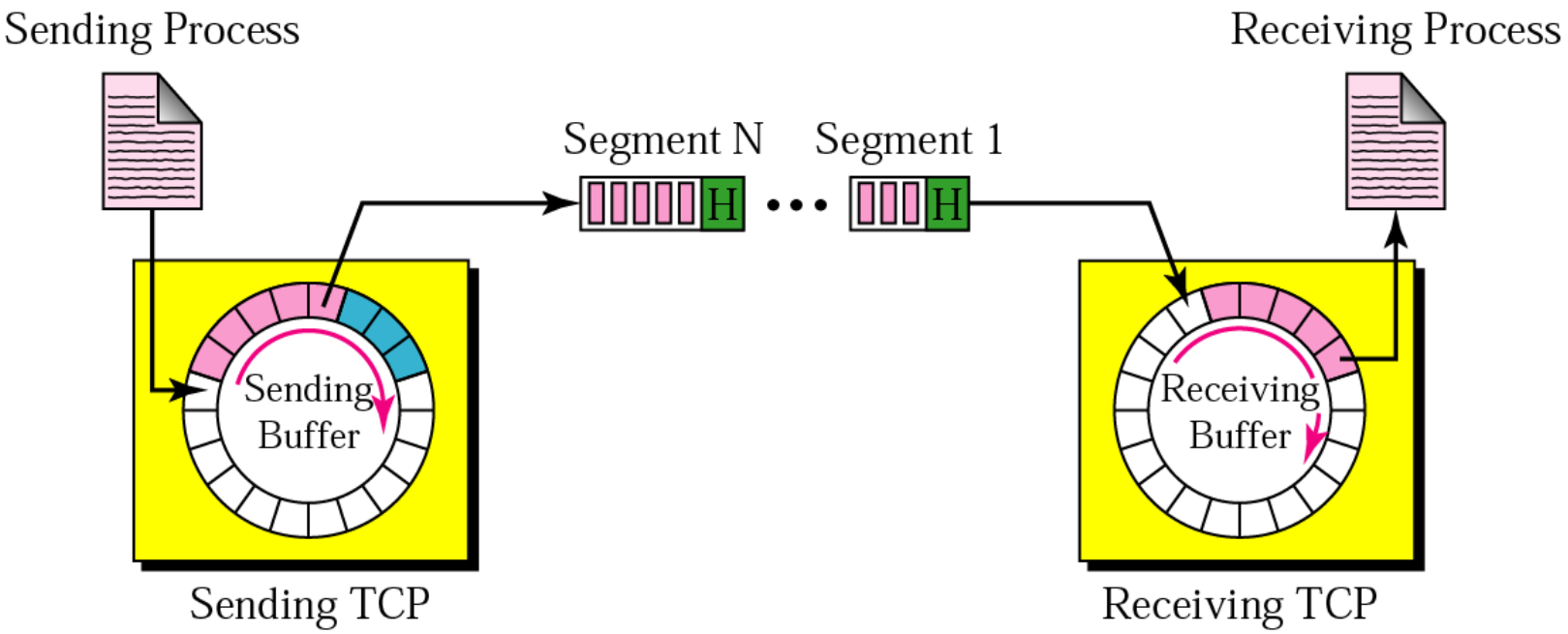
TCP Services (cont'd)

- Sending and Receiving Buffers
  - Because the sending and receiving processes may not produce and consume data at the same speed, TCP needs buffers for storage.
  - One way to implement is to use a circular array



TCP Services (cont'd)

- TCP Segments



TCP Services (cont'd)

- Stream Data Service (stream transport layer service)
  - The sending TCP

    1) accepts a stream of characters from sending application program

    2) creates packets called *segments*, of appropriate size extracted from the stream

    3) sends segments across the network
  - The receiving TCP

    1) receives segments, extracts data from segments

    2) orders segments if they have arrived out of order

    3) delivers segments as a stream of characters to the receiving application program

TCP Services (cont'd)

- Full-Duplex Service
  - TCP offers full-duplex service
    - After two application programs are connected to each other, they can both send and receive data.
  - Piggybacking
    - When a packet is going from A to B, it can also carry an acknowledgment of the packets received from B

TCP Services (cont'd)

- Connection-Oriented Services
  1. A's TCP informs B's TCP and gets approval from B's TCP
  2. A's TCP and B's TCP exchange data in both directions
  3. After both processes have no data left to send and the buffers are empty, two TCPs destroy their buffers
- Reliable Service
  - TCP uses the acknowledgment mechanism to check the safe and sound arrival of data

Numbering Bytes

- Byte numbers
  - All data bytes being transferred in each connection are numbered by TCP.
  - The numbering starts with a randomly generated number.
  - Number range for first byte : 0 ~ 2^{32} -1
    - If random number is 1,057 and total number 6,000bytes, the bytes are numbered from 1,057 to 7,056
  - Byte numbering is used for flow and error control.

Numbering Bytes (cont'd)

- Sequence number
  - After the bytes have been numbered, TCP assigns a sequence number to each segment that is being sent.
  - Sequence number for each segment is number of the first byte carried in that segment.

Numbering Bytes (cont'd)

- Example 1
  - Imagine a TCP connection is transferring a file of 6,000 bytes. The first byte is numbered 10010. What are the sequence numbers for each segment if data are sent in five segments with the first four segments carrying 1,000bytes and the last segment carrying 2,000bytes?

Numbering Bytes (cont'd)

Solution

The following shows the sequence number for each segment:

Segment 1 ==> sequence number: 10,010 (range: 10,010 to 11,009)

Segment 2 ==> sequence number: 11,010 (range: 11,010 to 12,009)

Segment 3 ==> sequence number: 12,010 (range: 12,010 to 13,009)

Segment 4 ==> sequence number: 13,010 (range: 13,010 to 14,009)

Segment 5 ==> sequence number: 14,010 (range: 14,010 to 16,009)

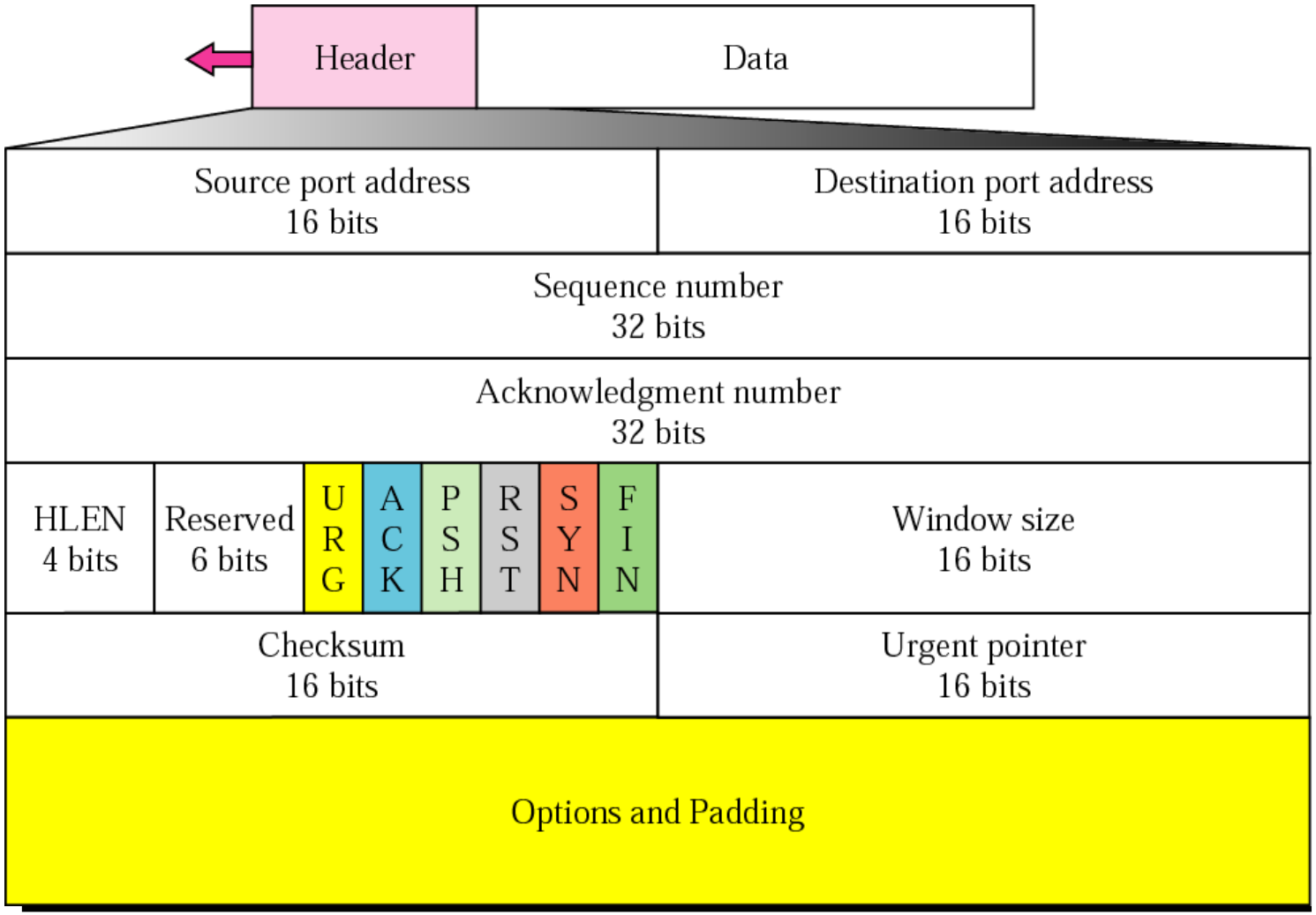
The value of the sequence number field in a segment defines the number of the first data byte contained in that segment.

Numbering Bytes (cont'd)

- Acknowledgment Number
  - The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receives. The acknowledgment number is cumulative.

Segment

- A packet in TCP is called segment



Segment (cont'd)

- Source port address
  - defining the port number of application program in the host that is sending the segment
- Destination port address
  - defining the port number of application program in the host that is receiving the segment
- Sequence number
  - defining the number assigned to the first byte of data contained in this segment
  - during the connection establishment, each party uses a random number generator to create an *initial sequence number (ISN)*

Segment (cont'd)

- Acknowledgment number
  - If the source of the segment has successfully received byte number x from the other party, it defines x+1 as the acknowledgment number
- Header length
  - Indicating the number of 4-byte words in the TCP header
    - the value between 5 and 15 (20 and 60 bytes)
- Reserved
  - For future use

Segment (cont'd)

- Control
  - Enabling flow control, connection establishment and termination, and mode of data transfer in TCP

URG: Urgent pointer is valid	RST: Reset the connection
ACK: Acknowledgment is valid	SYN: Synchronize sequence numbers
PSH: Request for push	FIN: Terminate the connection

Segment : Description of flags in the control field

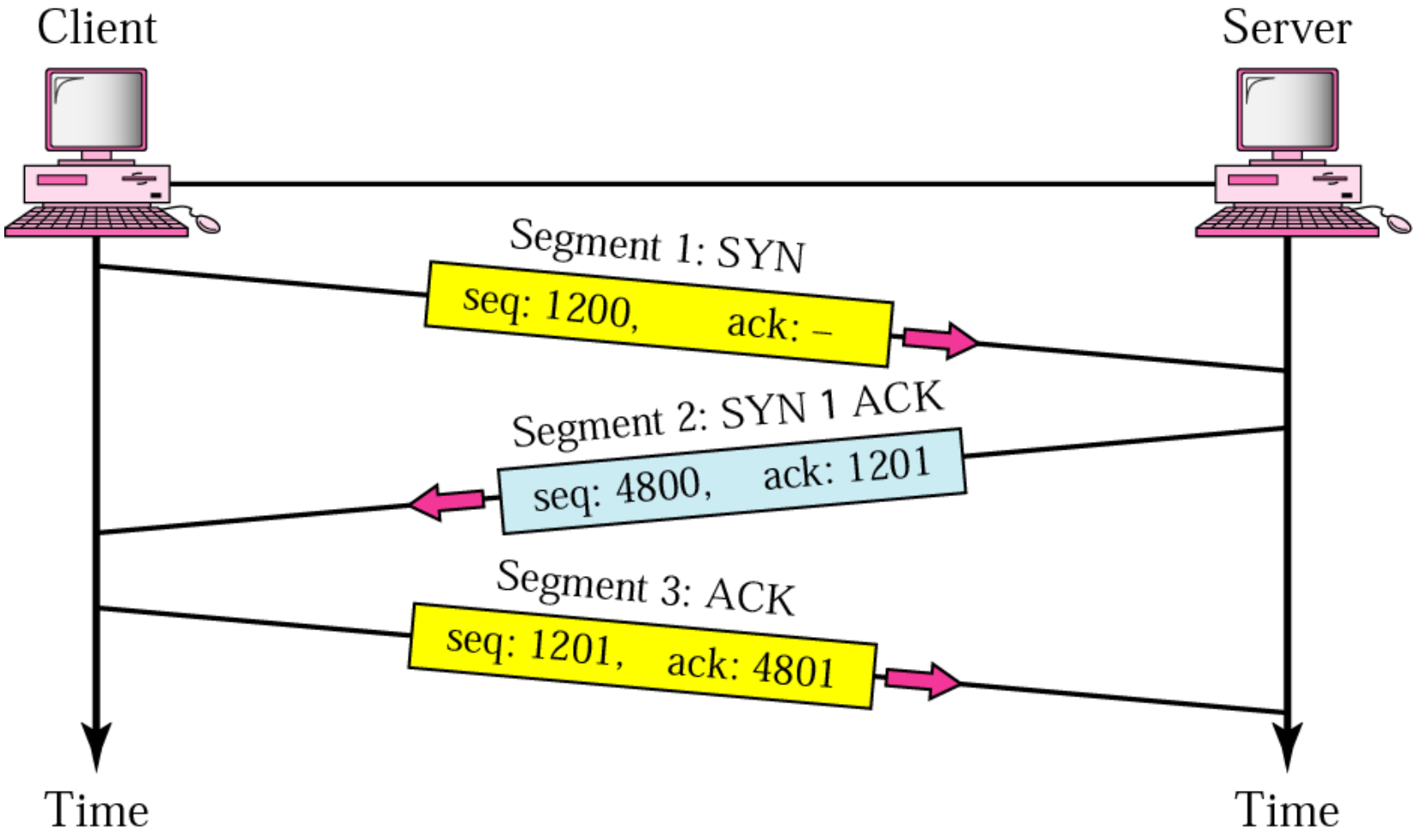
Flag	Description
URG	The value of the urgent pointer field is valid.
ACK	The value of the acknowledgment field is valid.
PSH	Push the data.
RST	The connection must be reset.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

Segment (cont'd)

- Window size
  - defining the size of the window, in bytes, that the other party must maintain.
  - maximum size of window : 65,535 bytes
- Checksum : 16-bit field contains the checksum; following the same procedure as the one described for UDP
- Urgent pointer
  - used when the segment contains urgent data
  - defining the number that must be added to the sequence number to obtain the number of the last urgent byte in the data section of the segment
- Options : 40 bytes

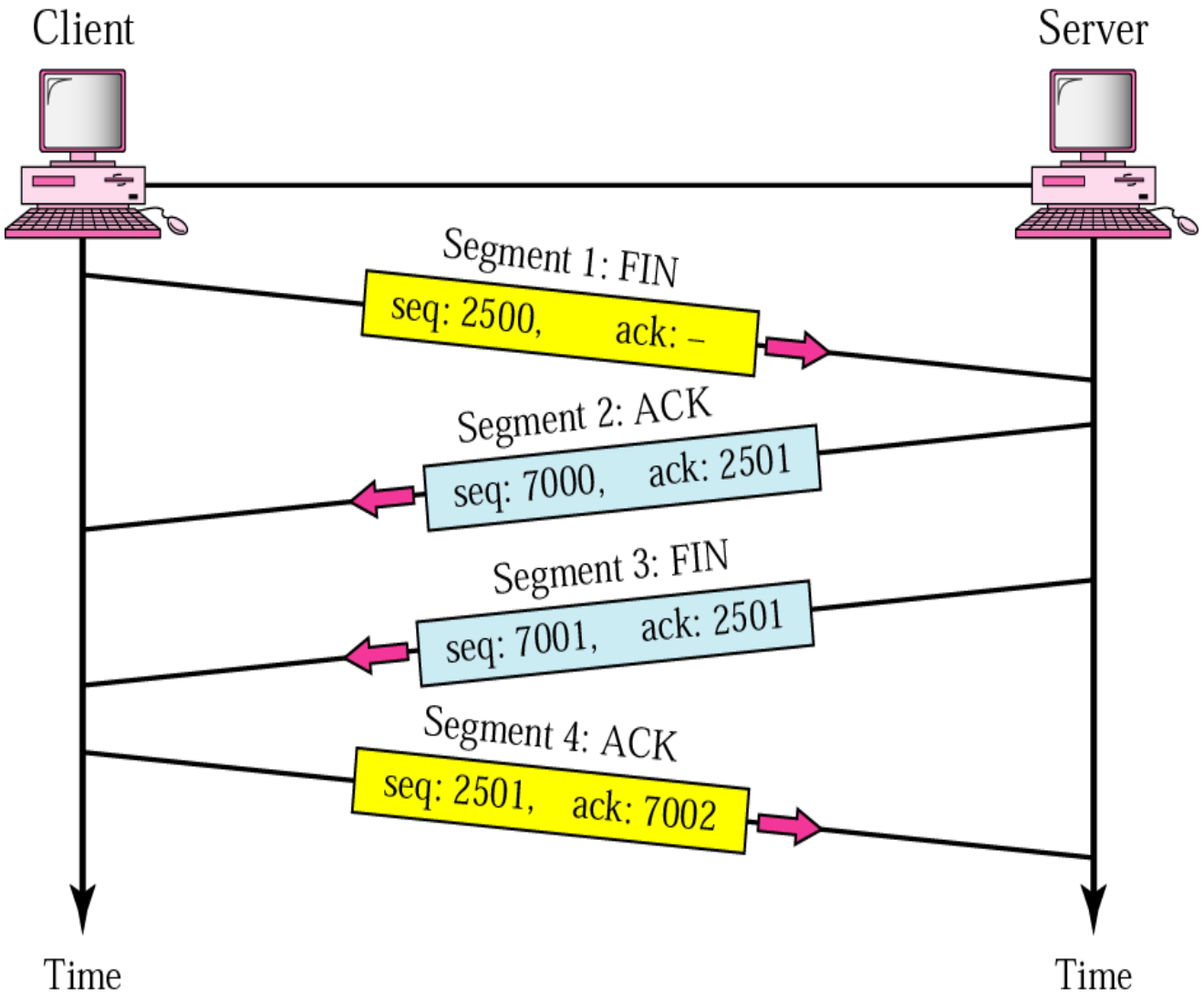
Connection

- Three-step (Three-way Handshaking) connection establishment



Connection Termination

- Four-step connection termination



Connection Resetting

- Resetting here means that the current connection is destroyed. This happens in one of three cases:
  1. The TCP on one side has requested a connection to a nonexistent port. The TCP on other side may send a segment with its TST bit set to annul the request
  2. One TCP may want to abort the connection due to an abnormal situation. It can send an RST segment to close the connection
  3. The TCP on one side may discover that the TCP on the other side has been idle for a long time. It may send an TST segment to destroy the connection

22.2 User Datagram Protocol (UDP)

- UDP is called a connectionless, unreliable transport protocol
  - provides process-to-process communication instead of host-to-host communication
  - no flow and error control
  - uses port numbers to multiplex data from the application layer

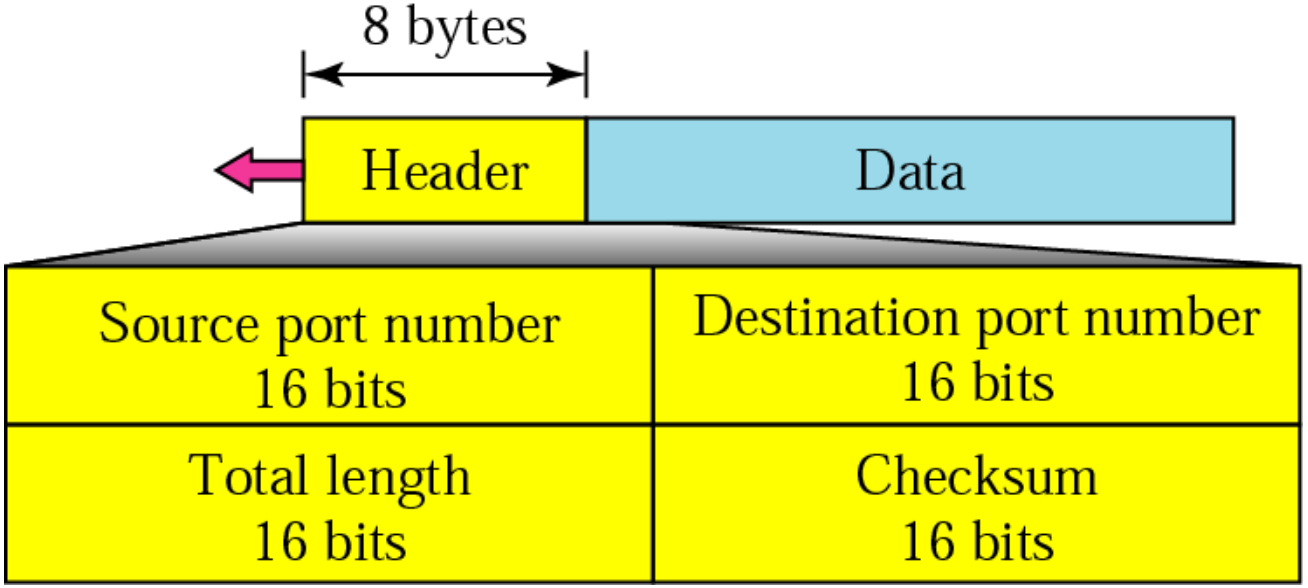
UDP (cont'd)

- If UDP is so powerless, why would a process want to use it ?
  - 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
    - if it sends a small message, taking much less interaction between the sender and receiver than it does using TCP

UDP : Well-known ports used by UDP

Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
67	Bootps	Server port to download bootstrap information
68	Bootpc	Client port to download bootstrap information
69	TFTP	Trivial File Transfer Protocol
111	RPC	Remote Procedure Call
123	NTP	Network Time Protocol
161	SNMP	Simple Network Management Protocol
162	SNMP	Simple Network Management Protocol (trap)

User Datagram Format



- The calculation of checksum and its inclusion in the user datagram are optional.

Applications of UDP

- TFTP including flow and error control
- Suitable transport protocol for multicasting
- Used in Routing Information Protocol (RIP)
- Used in conjunction with the Real Time Transport (RTP)
- Not used for sending bulk data, such as FTP
- For multimedia applications.