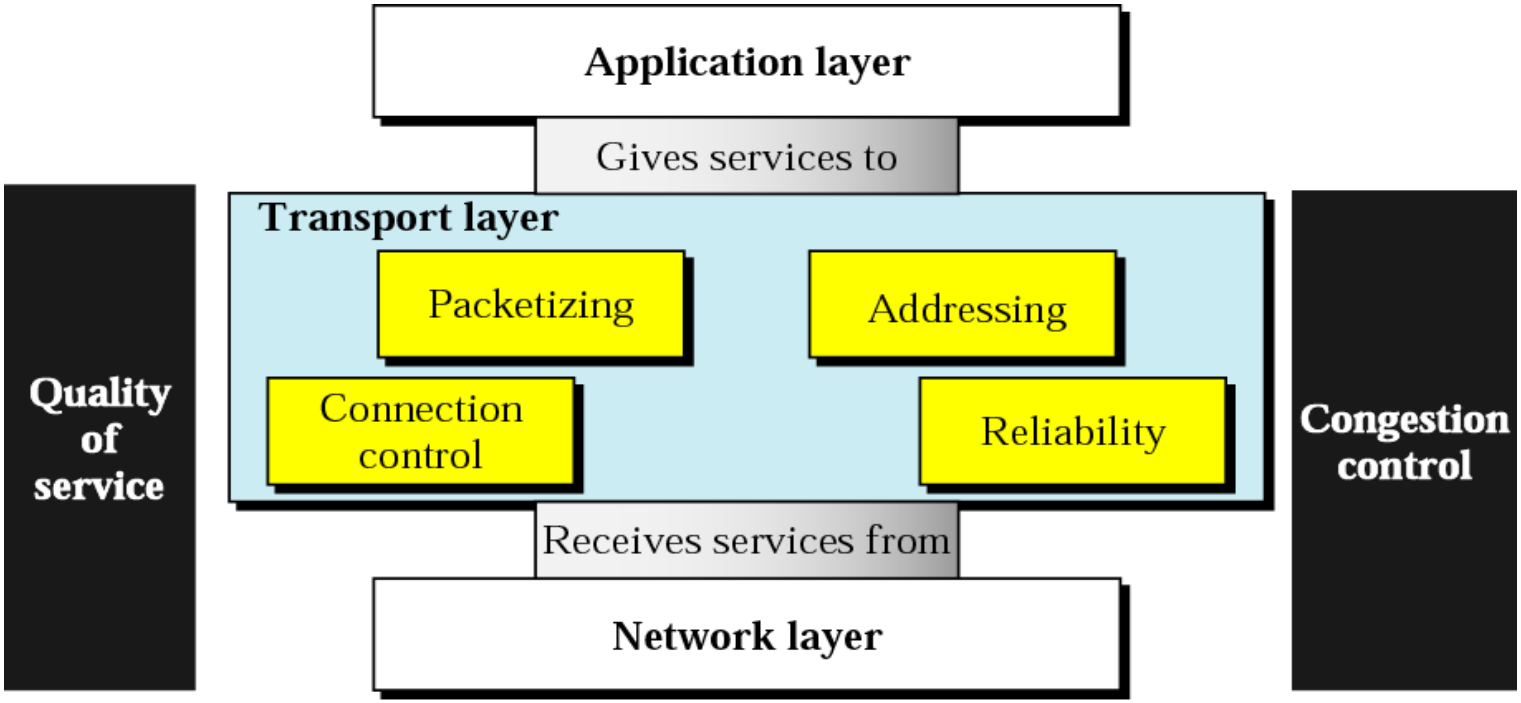
Transport Layer

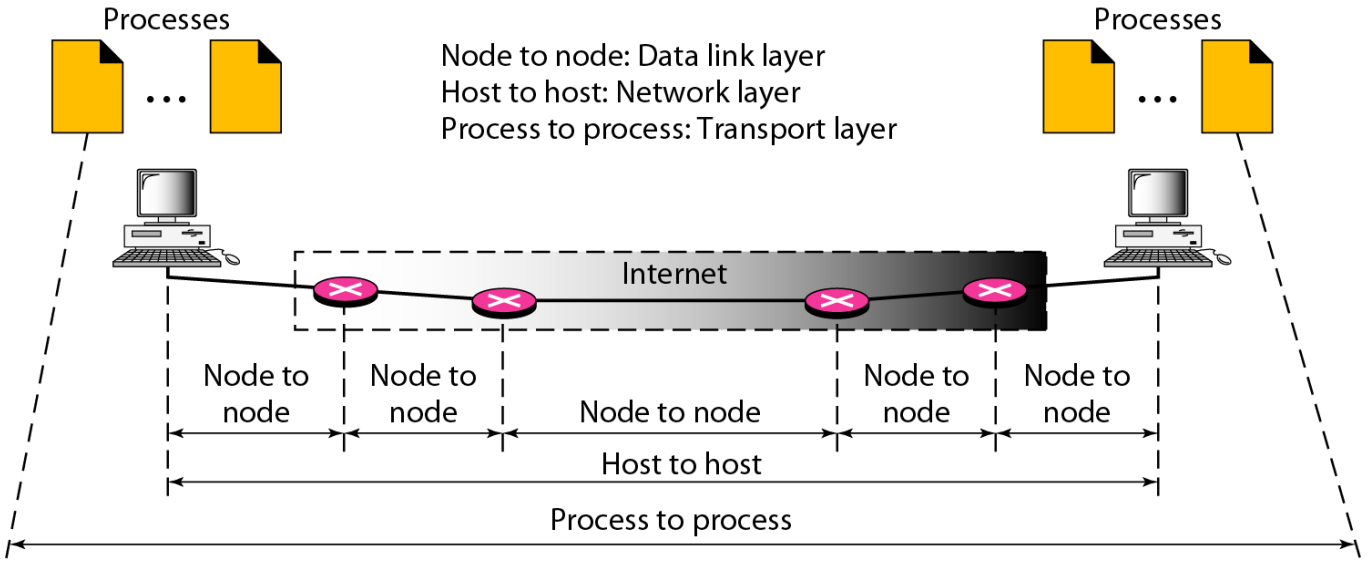
Position of Transport Layer

- Responsible for the delivery of a message from one process to another



Types of data deliveries

- The transport layer is responsible for process-to-process delivery.



Client-Server Paradigm

- Most common process-to-process communication is the client-server paradigm
- Operating systems support multiuser and multiprogramming environments.
- Local host, local process, remote host, remote process must be defined

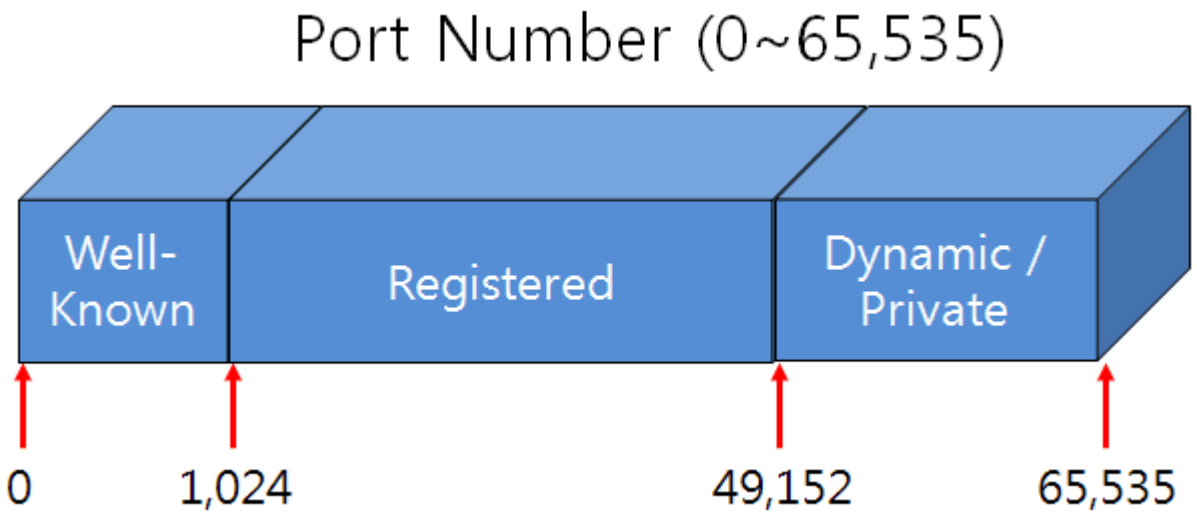
Addressing

- Address required for delivery
- Transport layer address: port number

Port Numbers

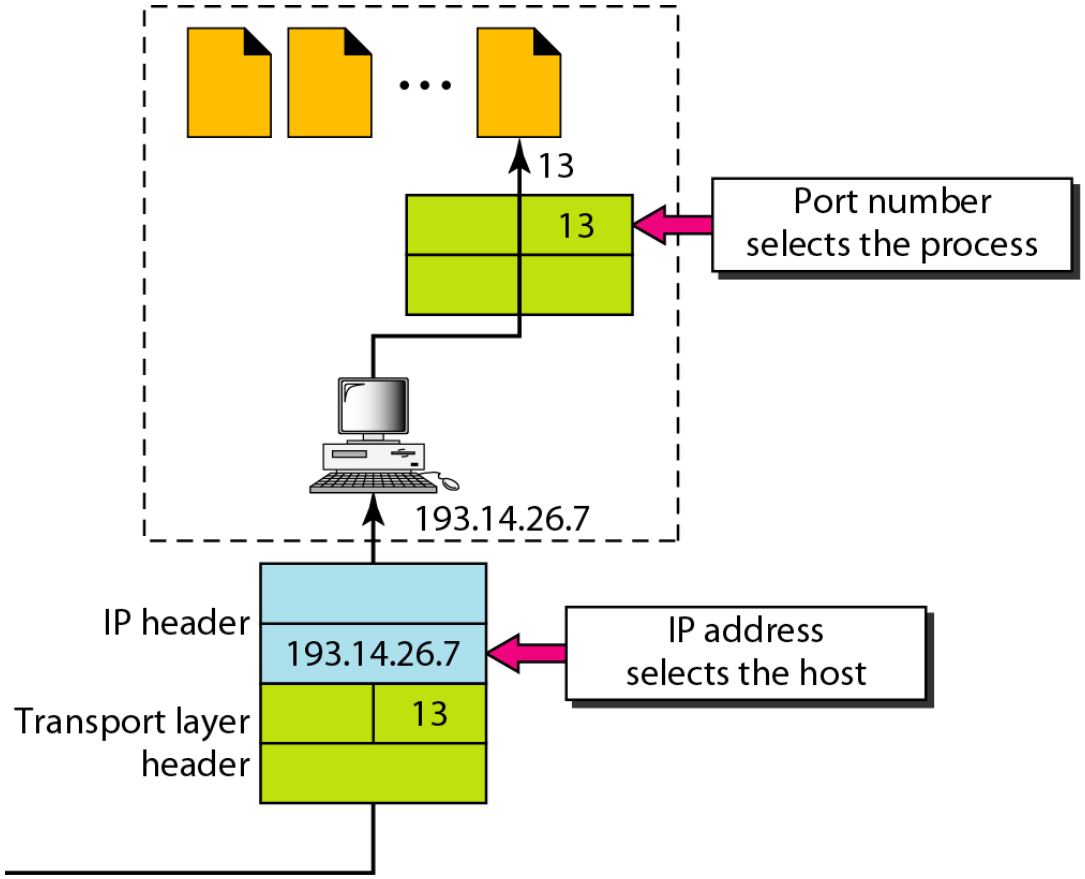
- The lANA (Internet Assigned Number Authority) has divided t he port numbers into three ranges: well known, registered, and dynamic (or private)
- Well-known ports.
  - The ports ranging from 0 to 1023 are assigned and controlled by lANA. These are the well-known ports.
- Registered ports.
  - The ports ranging from 1024 to 49,151 are not assigned or co ntrolled by lANA.
- Dynamic ports
  - The ports ranging from 49,152 to 65,535 are neither controlle d nor registered. They can be used by any process. These are the ephemeral ports

Port Numbers



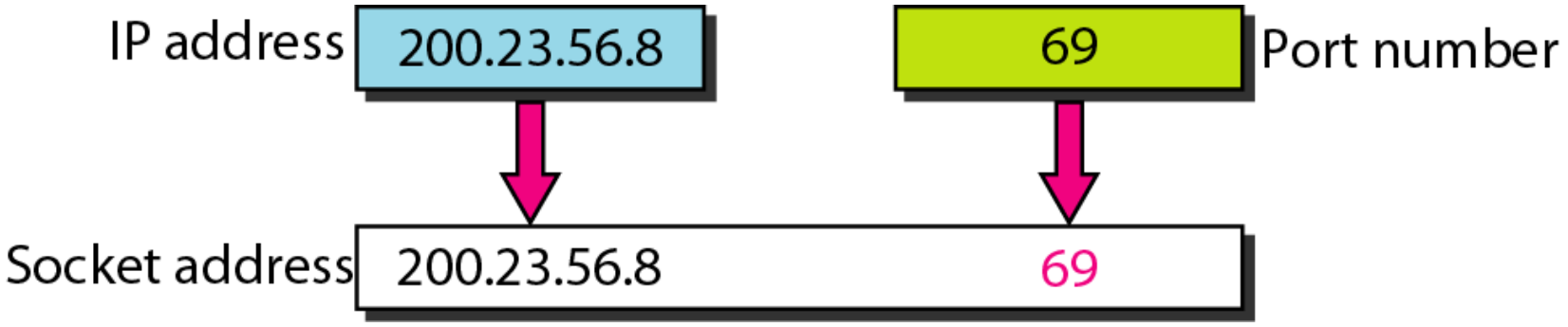
Application	Protocol	Port Number
File Transfer Protocol FTP Client	TCP	20
File Transfer Protocol FTP Server	TCP	21
Secure Shell SSH	TCP	22
Telnet	TCP	23
Simple Mail Transport Protocol SMTP	TCP	25
Domain Name System DNS	UDP / TCP	53
Dynamic Host Configuration Protocol DHCP	UDP	67,68
Trivial File Transfer Protocol TFTP	UDP	69
Hypertext Transfer Protocol HTTP	TCP	80
Post Office Protocol 3 POP3	TCP	110
Simple Network Management Protocol SNMP	UDP	161
Hypertext Tranfer Protocol Secure HTTPS	TCP	443

IP Addresses vs. Port Numbers



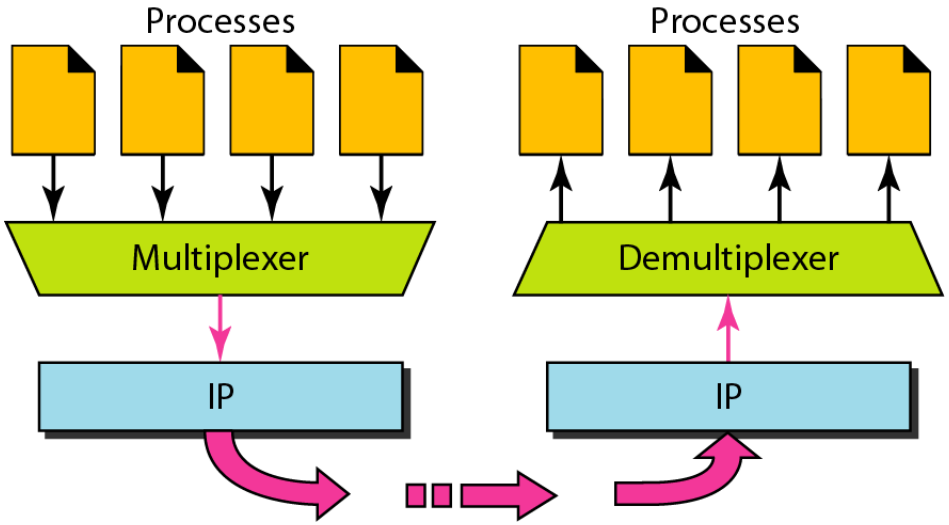
Socket address

- Process-to-process delivery needs two identifiers, IP address and the port number
- Socket address is the combination of an IP address and a port number
- A transport-layer protocol needs a pair of socket addresses; the client and server socket address



Multiplexing and demultiplexing

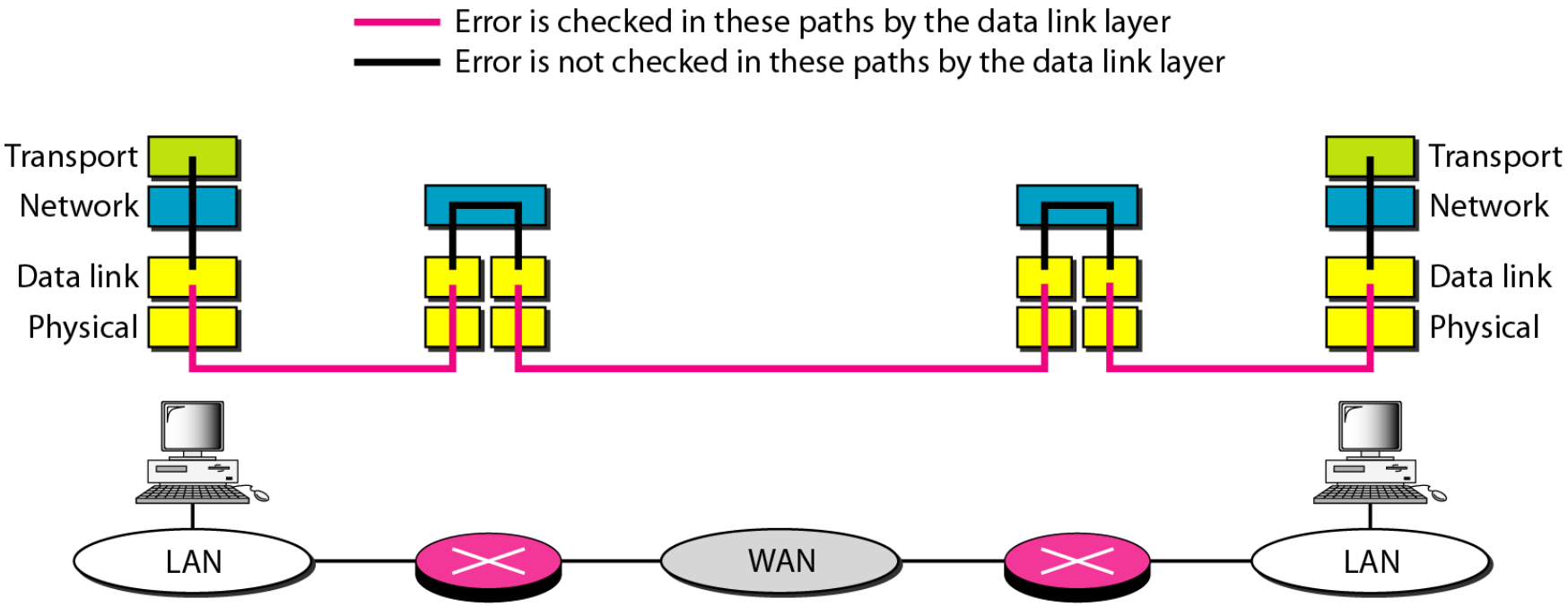
- Addressing mechanism allows multiplexing and demultiplexing by the transport layer



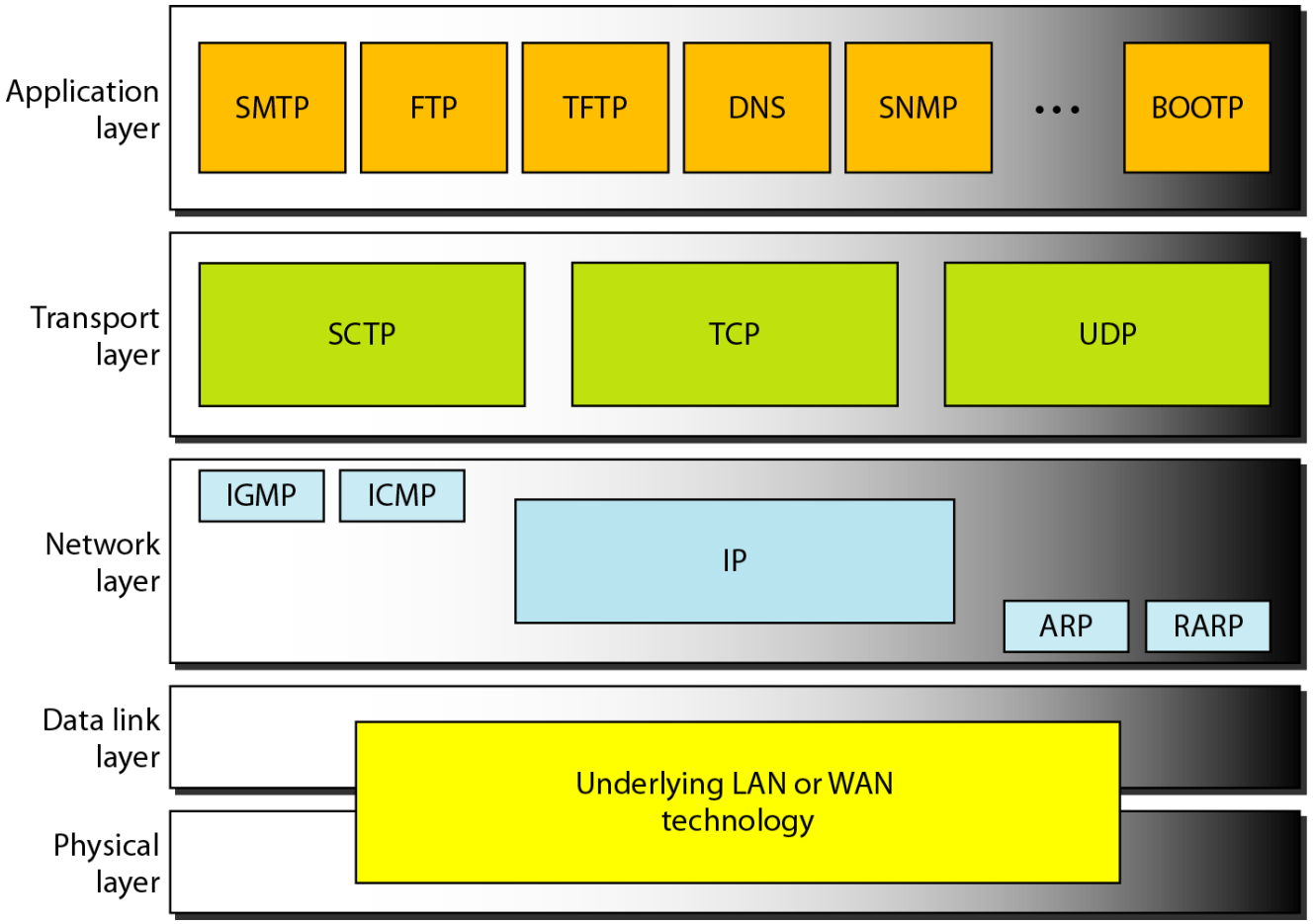
Connectionless vs. Connection-oriented

- Connection-oriented: connection established, data transferred, connection released
  - TCP and SCTP
- Connectionless: UDP

Reliable vs. Unreliable

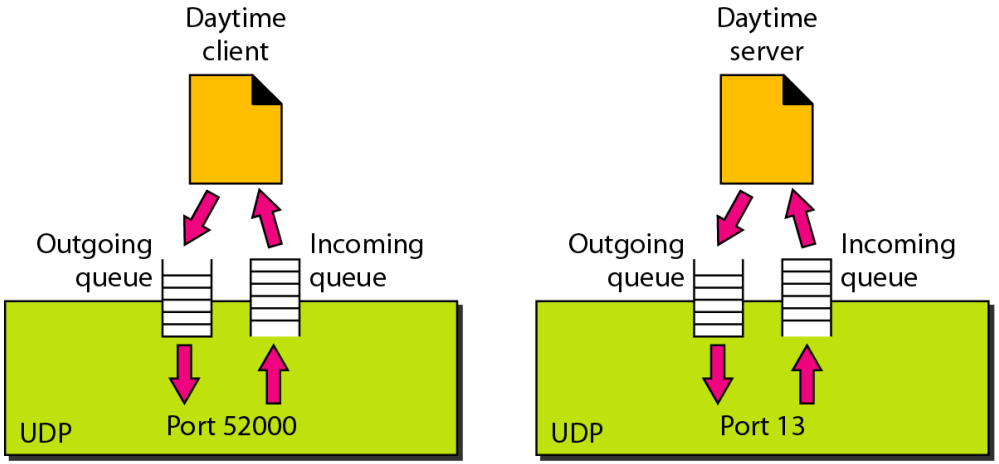


Position of UDP, TCP, and SCTP

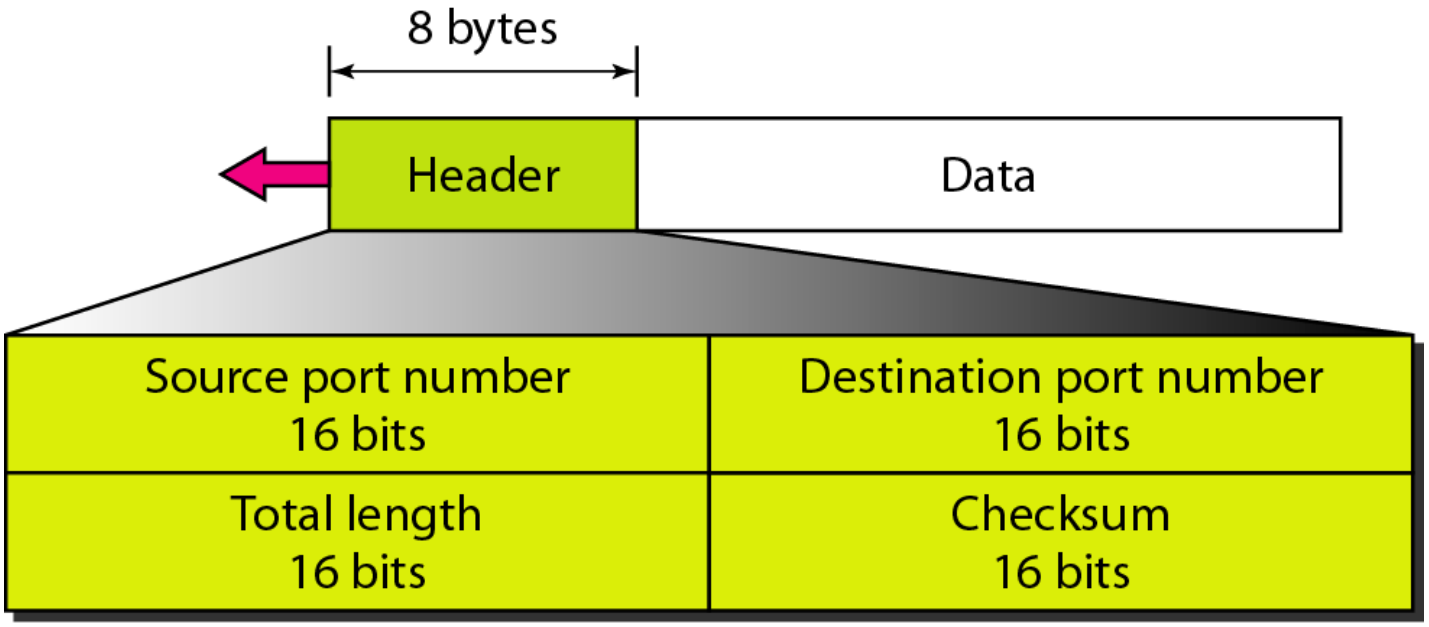


UDP Operation

- Connectionless service- no relation between datagram, not numbered
- No Flow and error control- no flow control so no window mechanics.
- No error control except checksum (silently discard packet)
- Encapsulation and Decapsulation



- The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication
- UDP is a convenient transport-layer protocol for applications that provide flow and error control. It is also used by multimedia applications.
- The calculation of checksum and its inclusion in the user datagram are optional



Well-known Ports for UDP

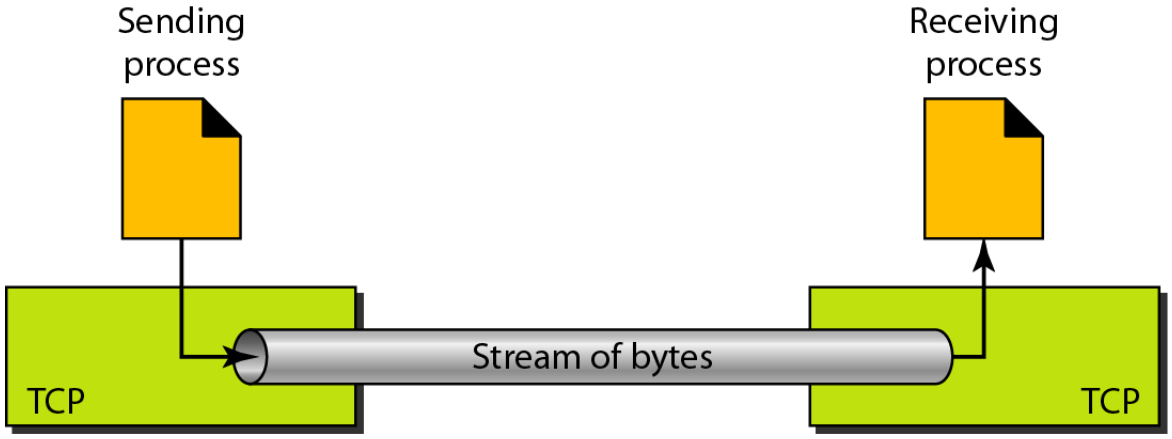
Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service
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)

Use of UDP

- Suitable for a process that requires simple request-response communication with little concern for flow and error control
- Suitable for a process with internal flow and error control mechanisms such as TFTP
- Suitable for multicasting
- Used for management processes such as SNMP
- Used for some route updating protocols such as RIP

TCP

- Transmission Control Protocol
- Connection-oriented, reliable transport protocol
- It adds connection-oriented and reliability features to the services of IP
- Like UDP, TCP uses port numbers as transport-layer addresses
- Unlike UDP, TCP is a stream-oriented protocol

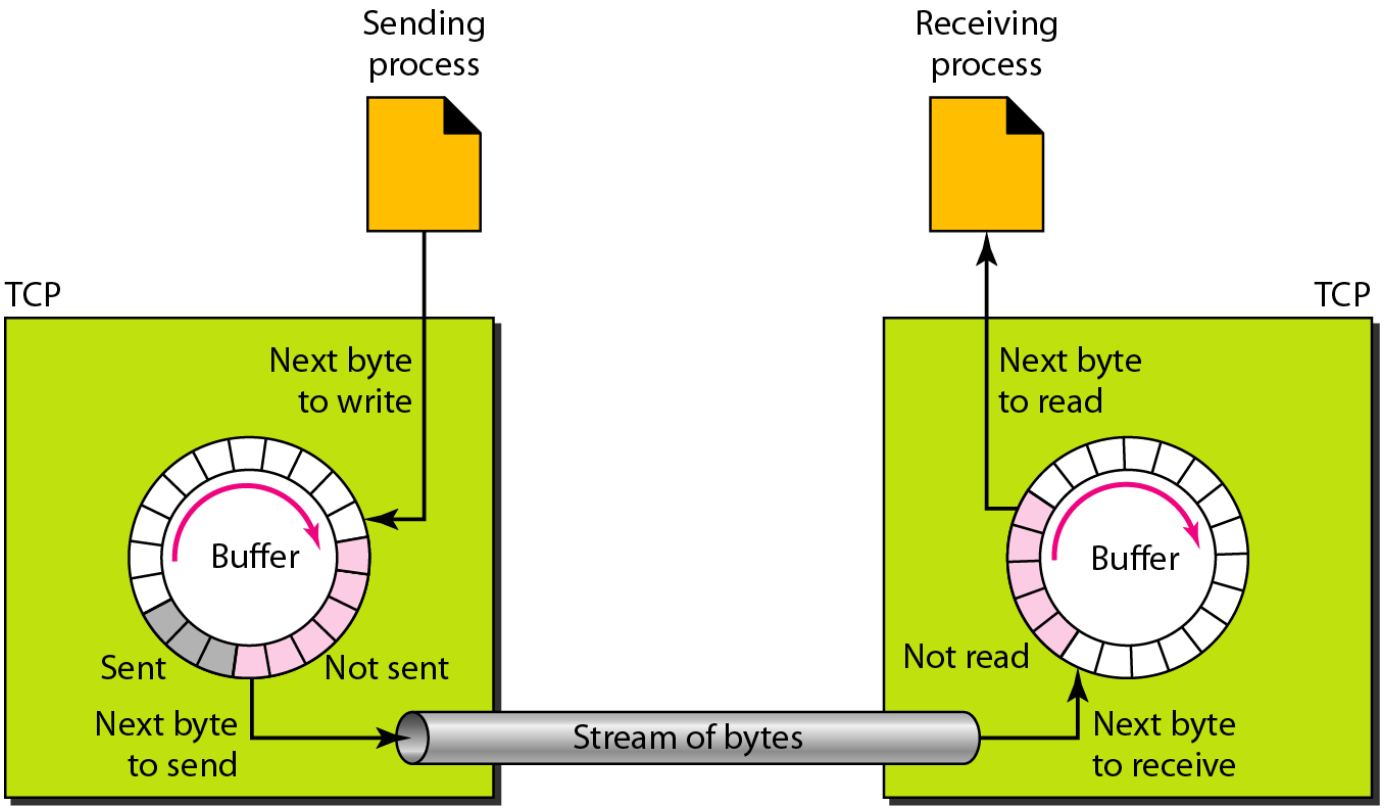


Well-known Ports for TCP

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

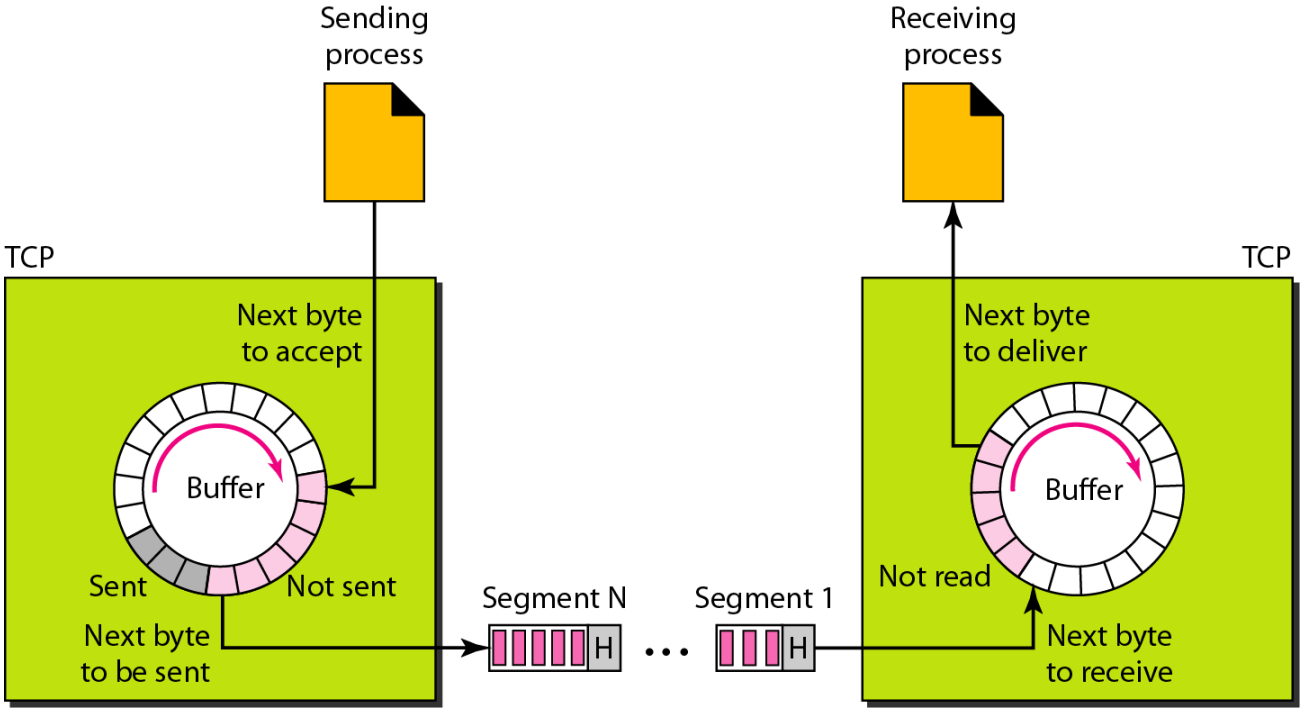
Sending and Receiving Buffers

- Buffering handles the disparity between the speed of the producing and consuming processes
- One example: to use a circular array of 1-byte locations



TCP Segments

- IP layer needs to send data in packets not as a stream of byte

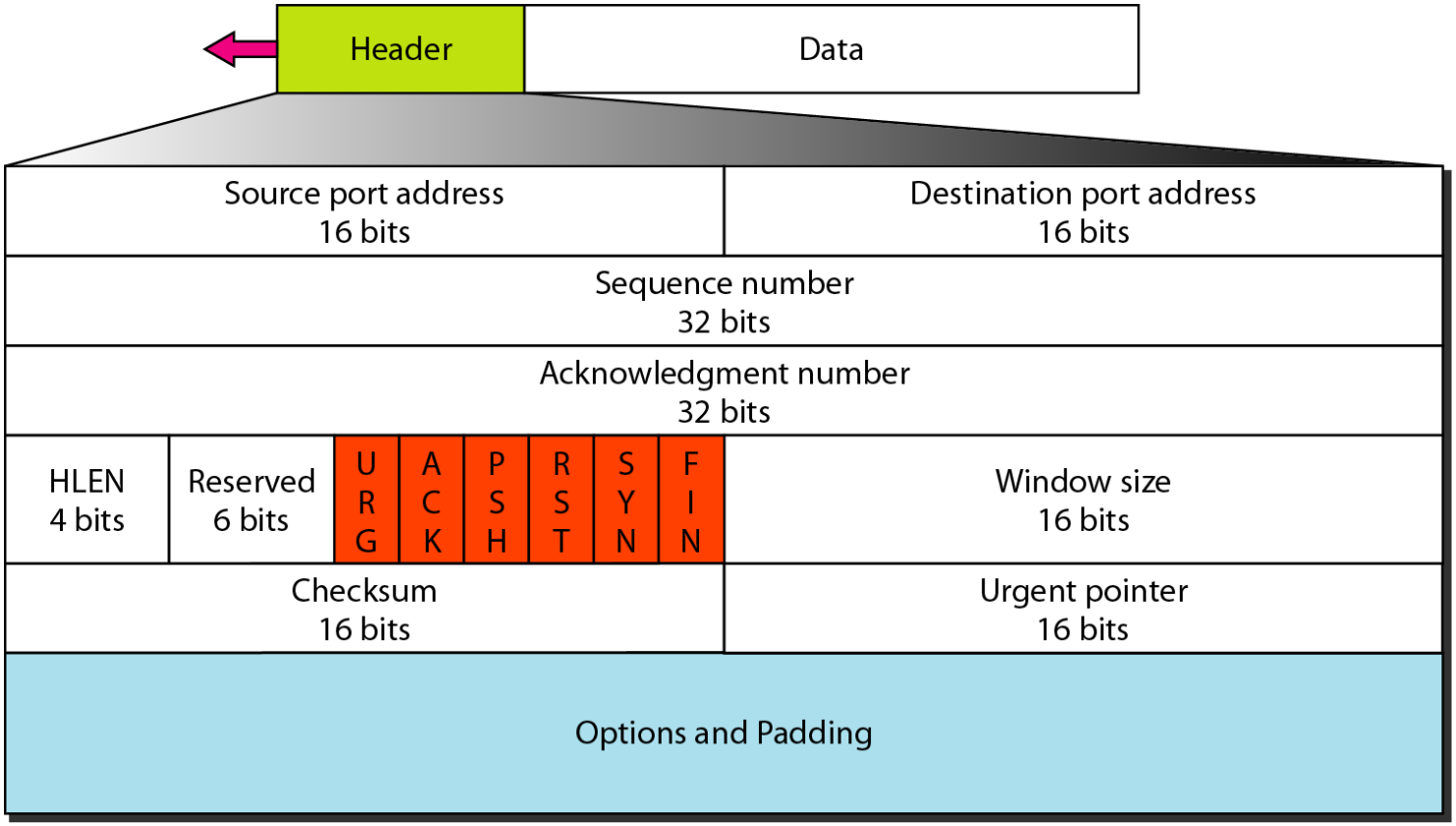


TCP Features

- Numbering system: sequence number and acknowledgment number
- Byte number: The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number
- The value in the sequence number field of a segment defines the number of the first data byte contained in that segment
- The value of the acknowledgment field in a segment defines the number of the next byte a party expects to receive. The acknowledgment number is cumulative

Segment 1	→	**Sequence Number: 10,001 (range: 10,001 to 11,000)**
Segment 2	→	**Sequence Number: 11,001 (range: 11,001 to 12,000)**
Segment 3	→	**Sequence Number: 12,001 (range: 12,001 to 13,000)**
Segment 4	→	**Sequence Number: 13,001 (range: 13,001 to 14,000)**
Segment 5	→	**Sequence Number: 14,001 (range: 14,001 to 15,000)**

TCP Segment Format



TCP Control Field

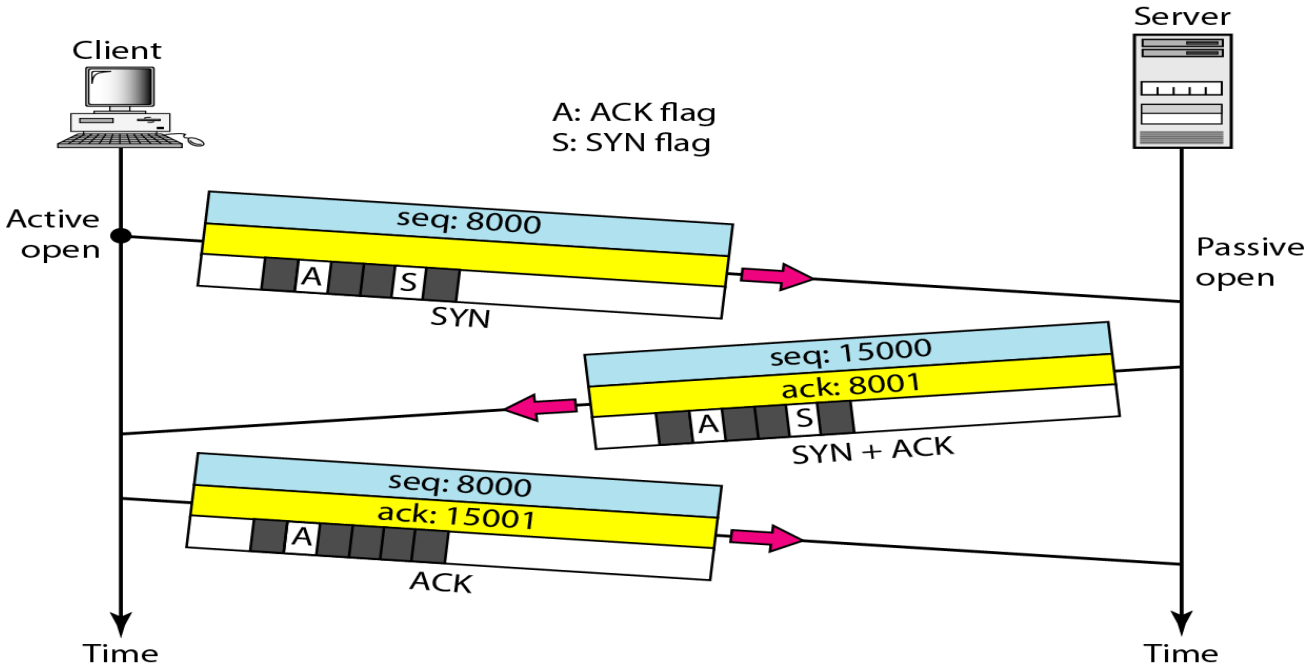
URG: Urgent pointer is valid
ACK: Acknowledgment is valid
PSH: Request for push

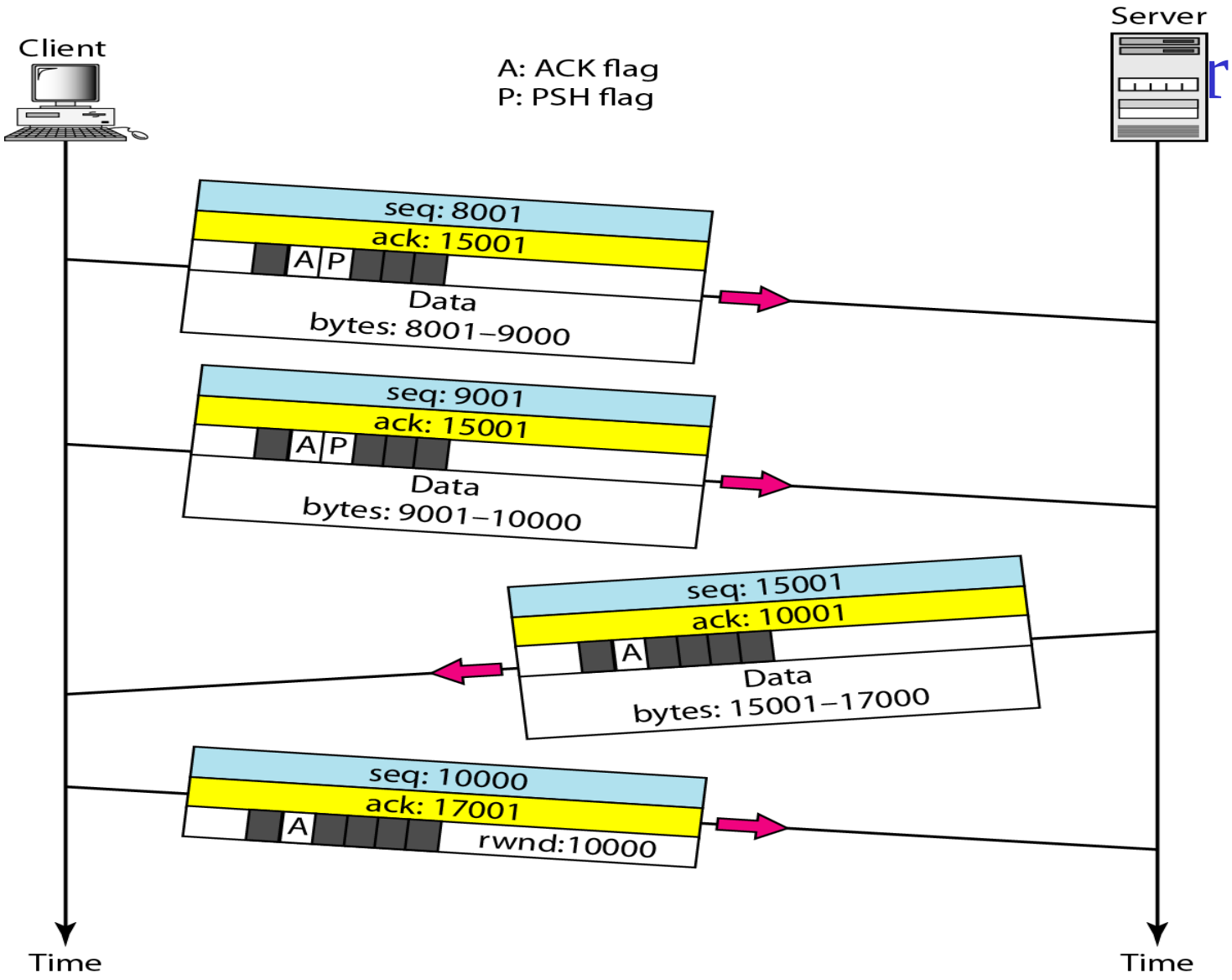
RST: Reset the connection
SYN: Synchronize sequence numbers
FIN: Terminate the connection

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	Reset the connection.
SYN	Synchronize sequence numbers during connection.
FIN	Terminate the connection.

A TCP Connection: Establishment

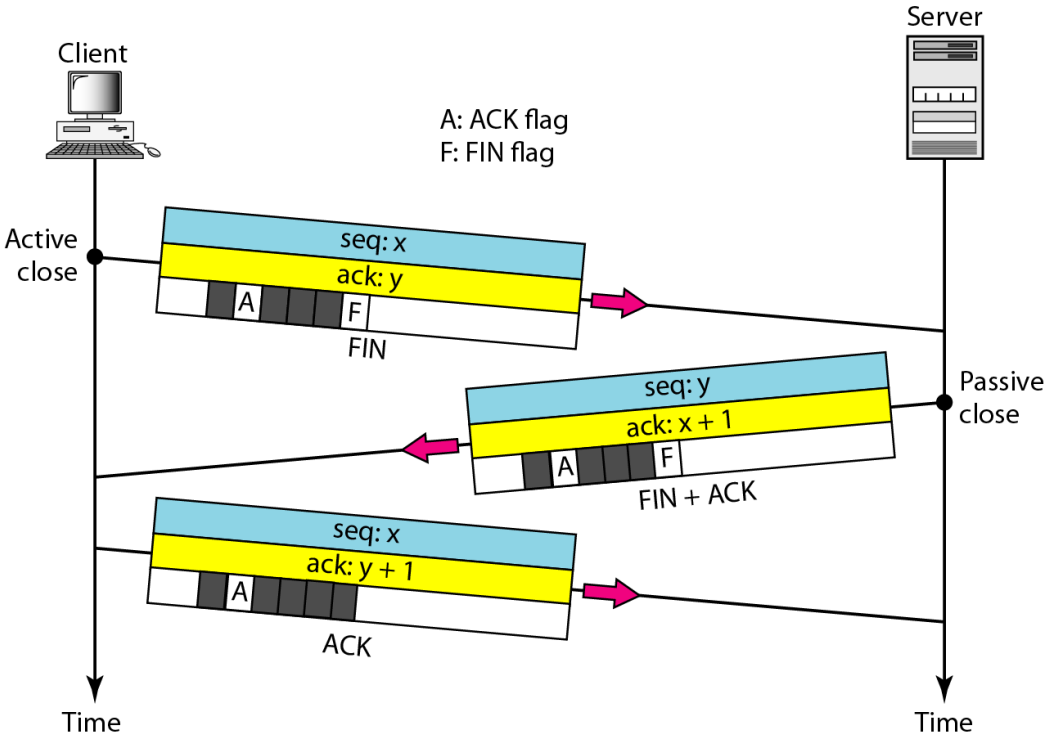
- Connection establishment: Three-way handshaking
- A SYN segment cannot carry data, but it consumes one sequence number
- A SYN + ACK segment cannot carry data, but does consume one sequence number
- An ACK segment, if carrying no data, consumes no sequence number
- Simultaneous open and ***SYN flooding attack (denial-of service attack, cookie)***



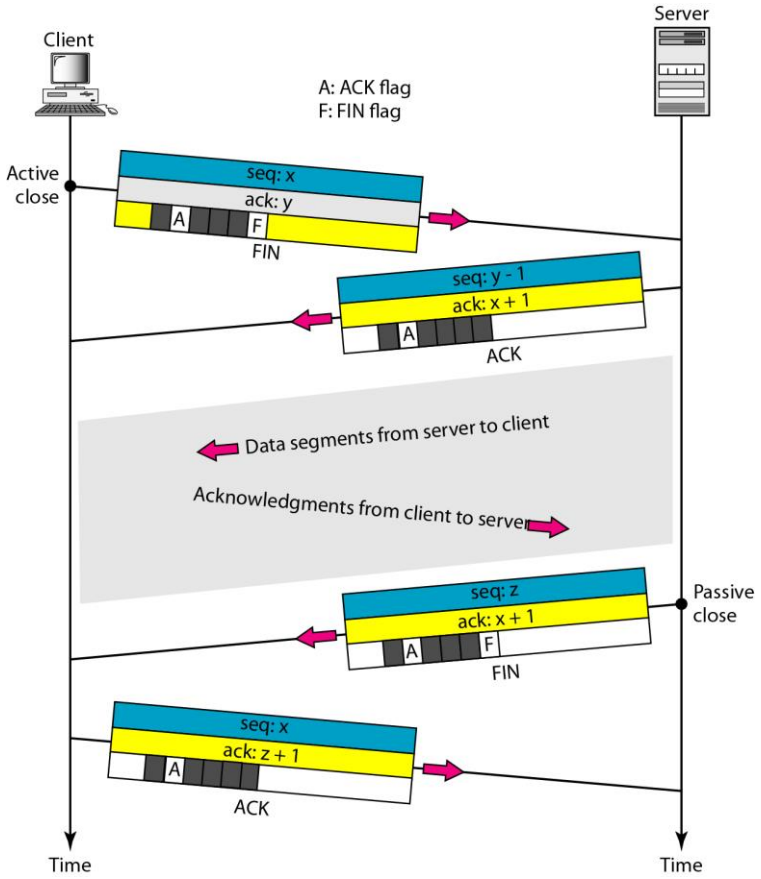

Client
Server
A: ACK flag
P: PSH flag
seq: 8001
ack: 15001
A P
Data
bytes: 8001–9000
seq: 9001
ack: 15001
A P
Data
bytes: 9001–10000
seq: 15001
ack: 10001
A
Data
bytes: 15001–17000
seq: 10000
ack: 17001
A
rwnd:10000
Time
Time

A TCP Connection: Connection Termination

- Three-way handshaking
- The FIN segment consumes one sequence number if it does not carry data
- The FIN + ACK segment consumes one sequence number if it does not carry data

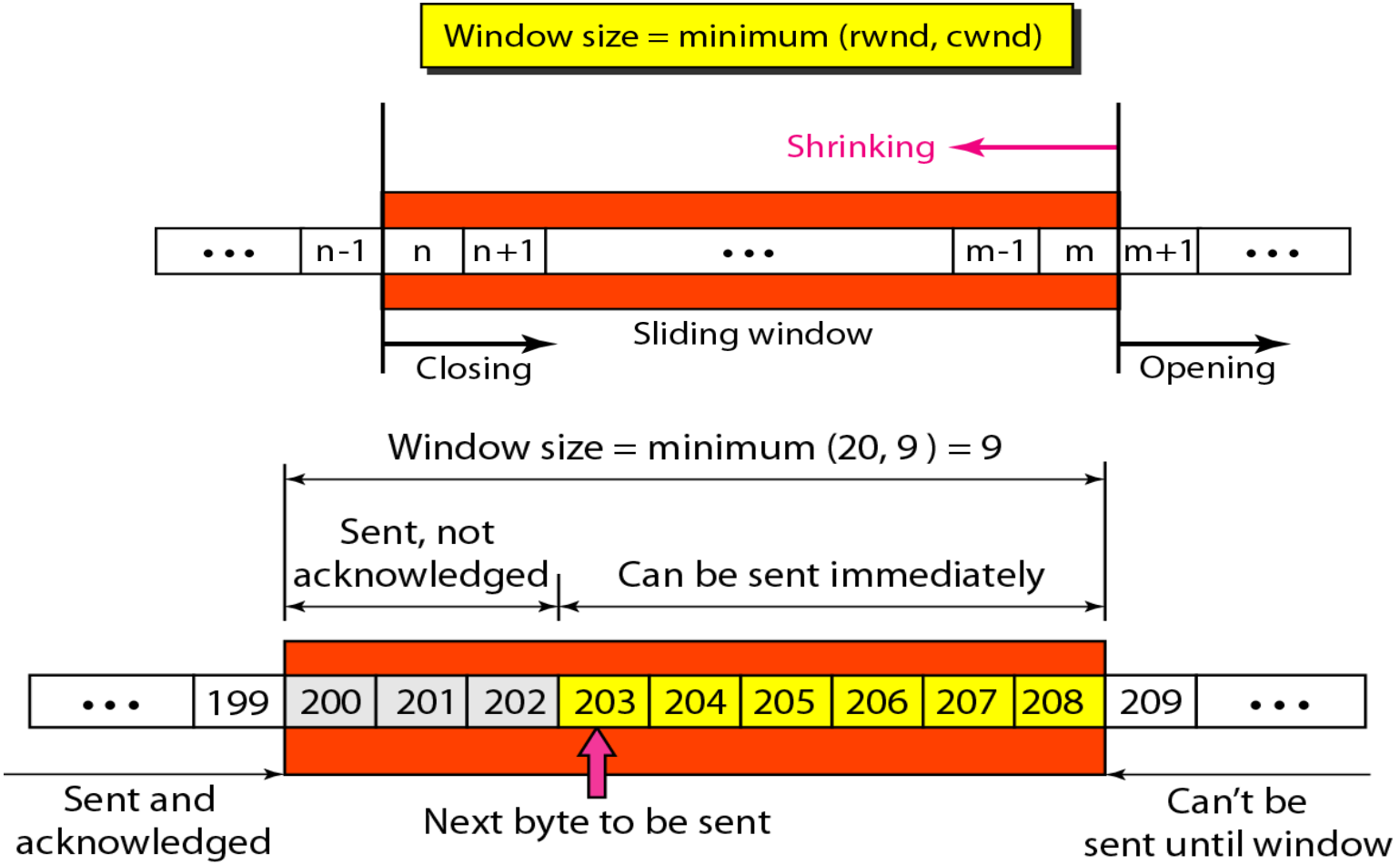


Connection Termination: Half-Close



Flow Control

- A sliding window is used to make transmission more efficient as well as to control the flow of data so that the destination does not become overwhelmed with data. TCP sliding windows are byte-oriented.



TCP Sliding Window

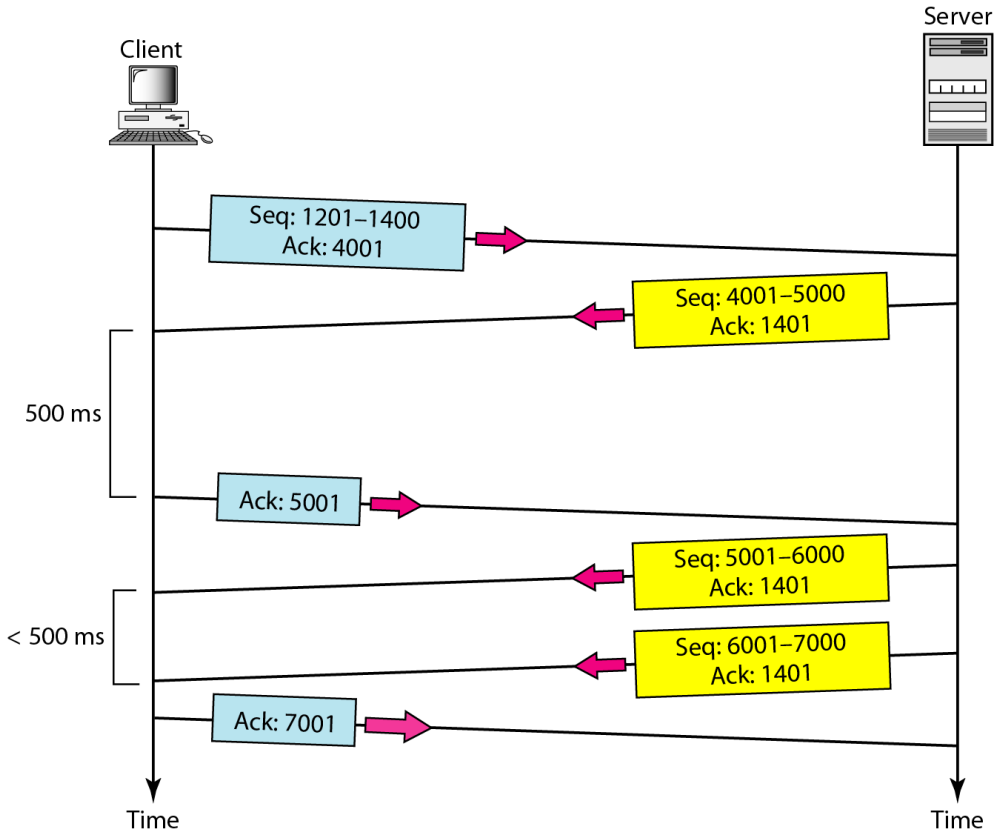
- The size of the window is the lesser of ***rwnd*** and ***cwnd***.
- The source does not have to send a full window's worth of data.
- The window can be opened or closed by the receiver, but should not be shrunk.
- The destination can send an acknowledgment at any time as long as it does not result in a shrinking window.
- The receiver can temporarily shut down the window; the sender, however, can always send a segment of 1 byte after the window is shut down.

Error Control

- Error detection and correction in TCP is achieved through the use of three simple tools: ***checksum, acknowledgment,*** and ***time-out***
- ***Checksum:*** If corrupted, it is discarded and considered as lost
- ***Acknowledgment:***
  - ACK segments do not consume sequence numbers and are not acknowledged
- ***Retransmission:***
  - In modern implementations, a retransmission occurs if the retransmission timer expires or three duplicate ACK segments have arrived
  - No retransmission timer is set for an ACK segment
  - Retransmission after RTO (Retransmission Time-Out): RTO is updated based on the RTT (Round Trip Time)
  - Retransmission after three duplicate ACK segments
- ***Out-of-order segments***
  - Data may arrive out of order and be temporarily stored by the receiving TCP, but TCP guarantees that no out-of-order segment is delivered to the process

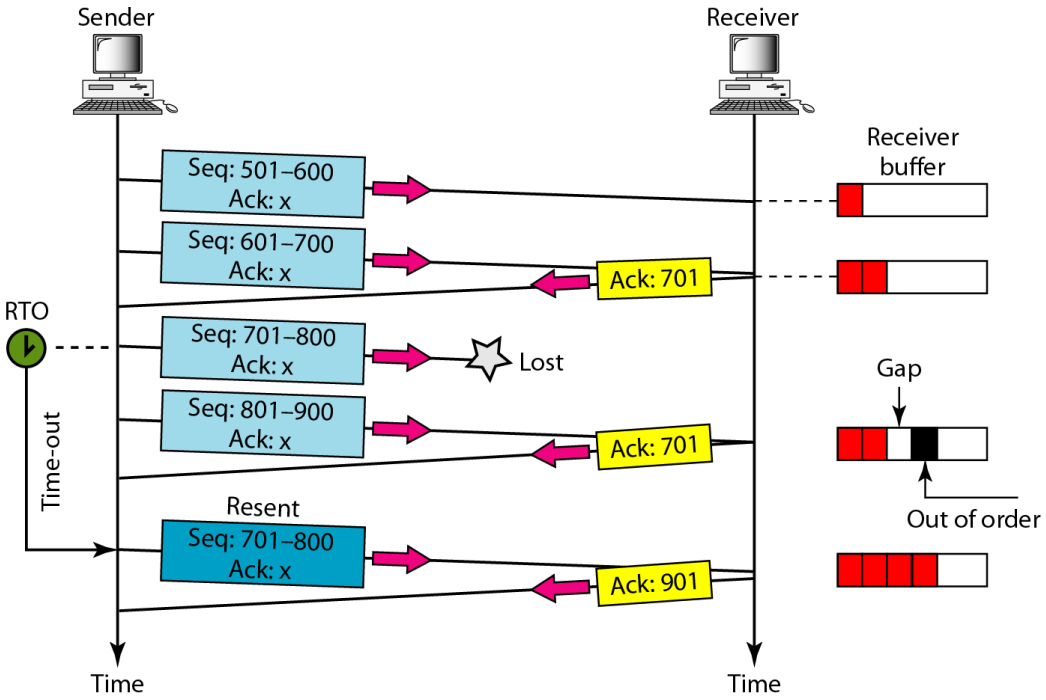
TCP Scenarios

- Normal Operation



TCP Scenarios

- Lost Segment
- The receiver TCP delivers only ordered data to the process



TCP Scenarios

- Fast retransmission