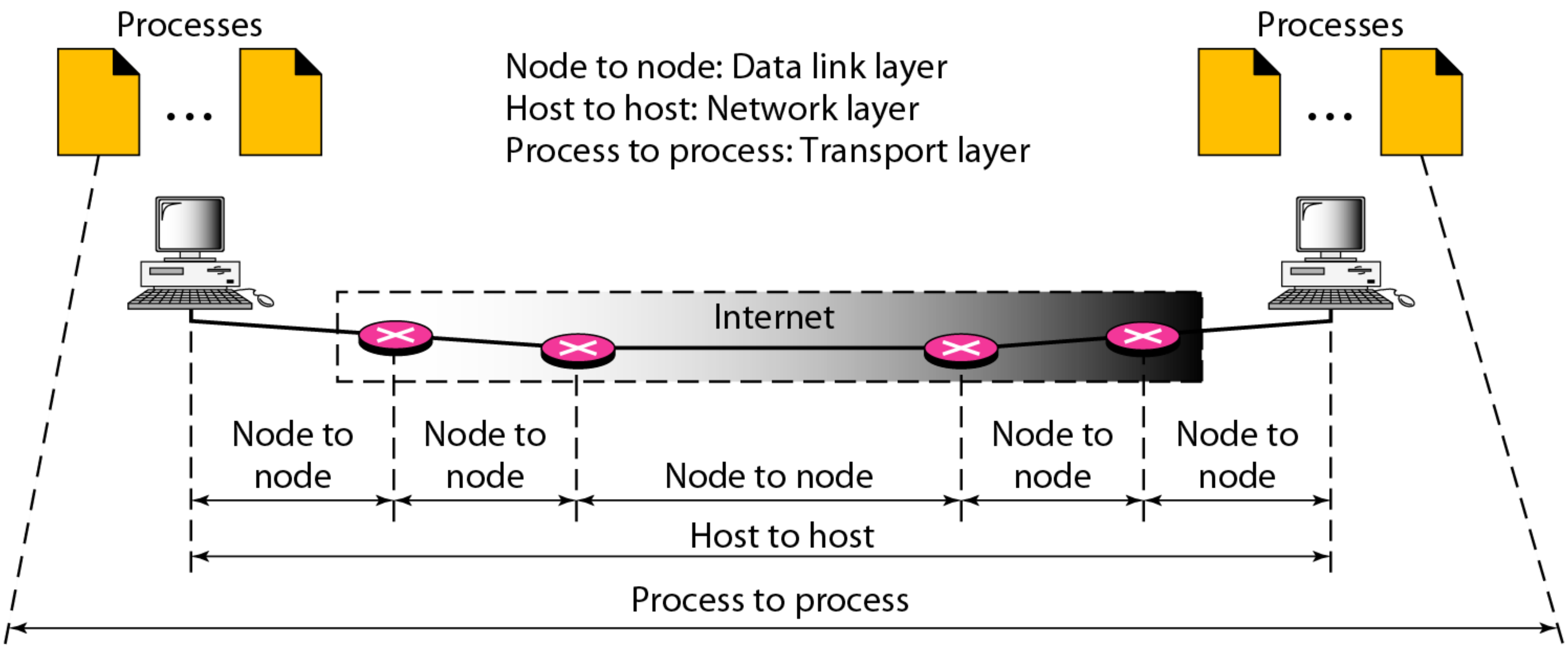
Chapter 23

Process-to-Process Delivery: UDP, TCP, and SCTP

23-1 PROCESS-TO-PROCESS DELIVERY

The 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/server relationship, as we will see later.

Figure 23.1 *Types of data deliveries*



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

Figure 23.2 *Port numbers*

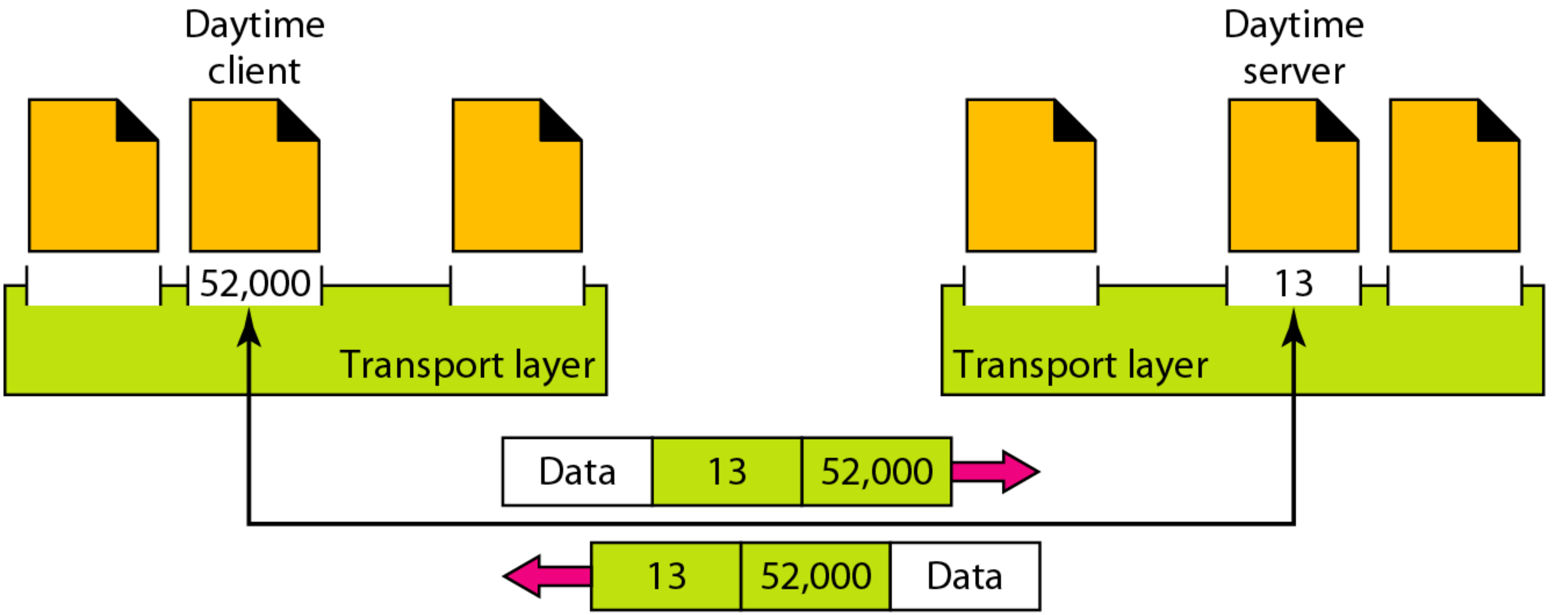


Figure 23.3 *IP addresses versus port numbers*

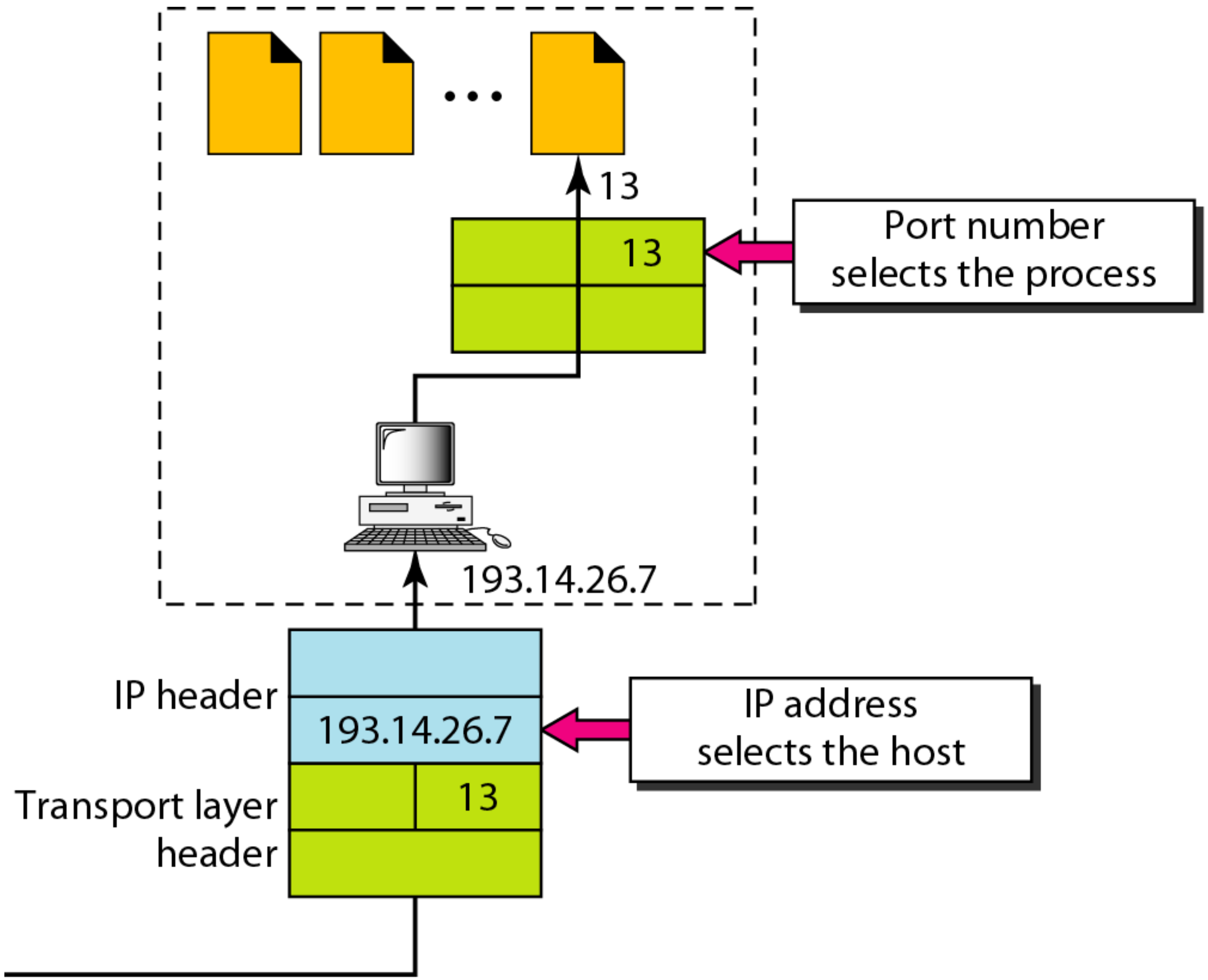


Figure 23.4 *IANA ranges*

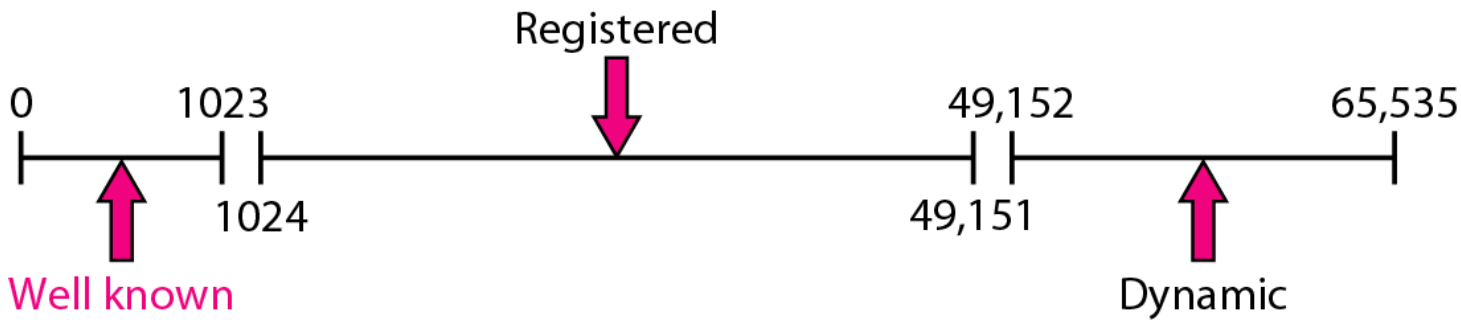


Figure 23.5 ***Socket address***

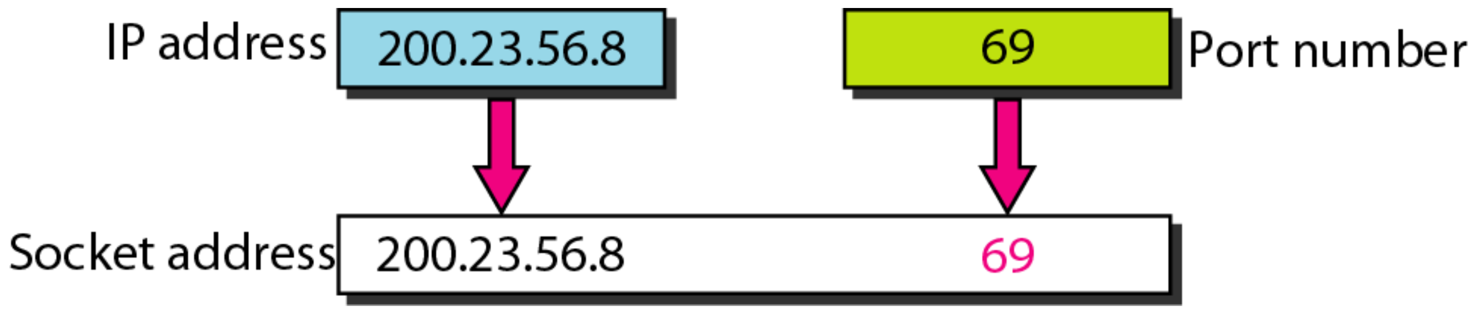
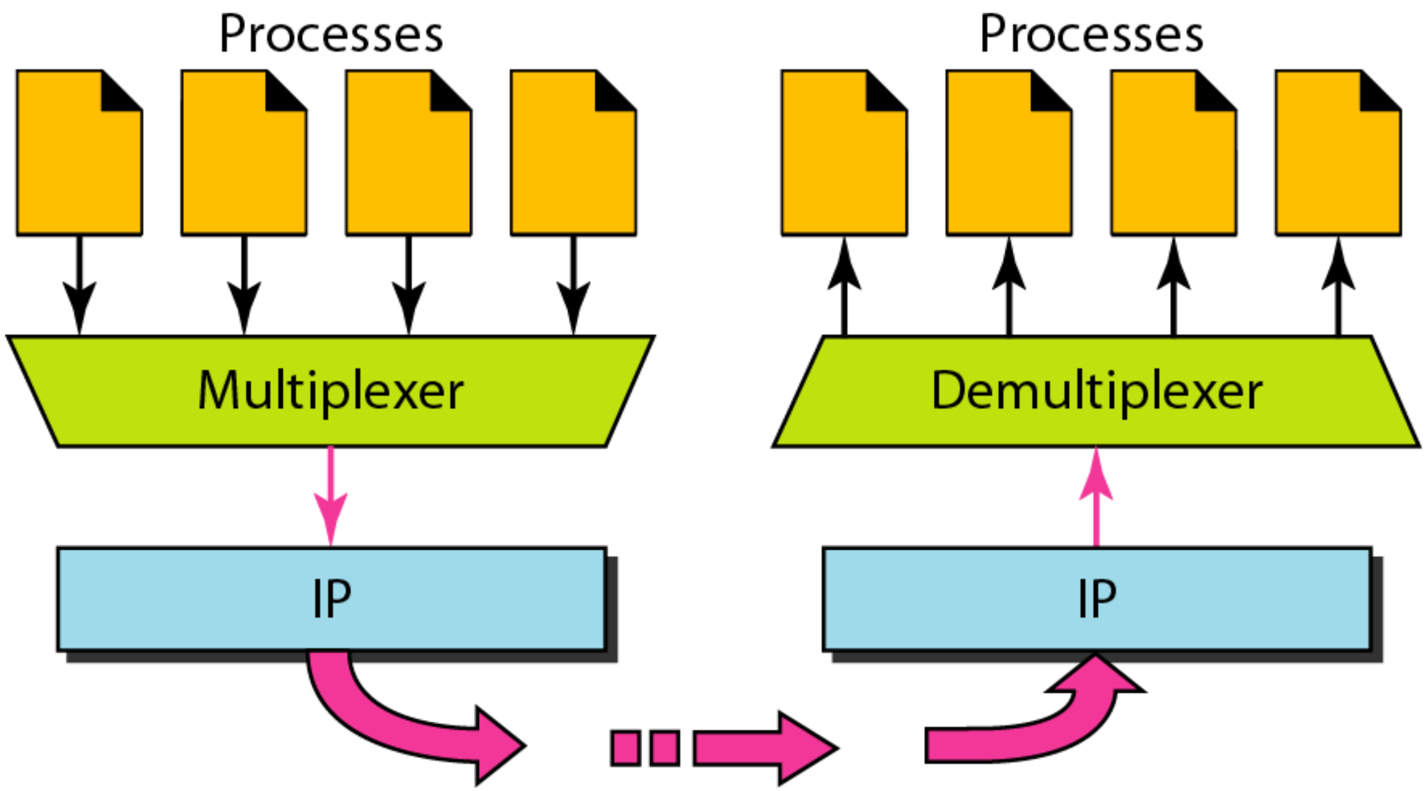


Figure 23.6 *Multiplexing and demultiplexing*



Flow and error control

- Flow control - buffers
- Error control - sequence numbers, acknowledgment

Combination of Flow and Error Control together - sliding window, congestion control

Figure 23.7 *Error control*

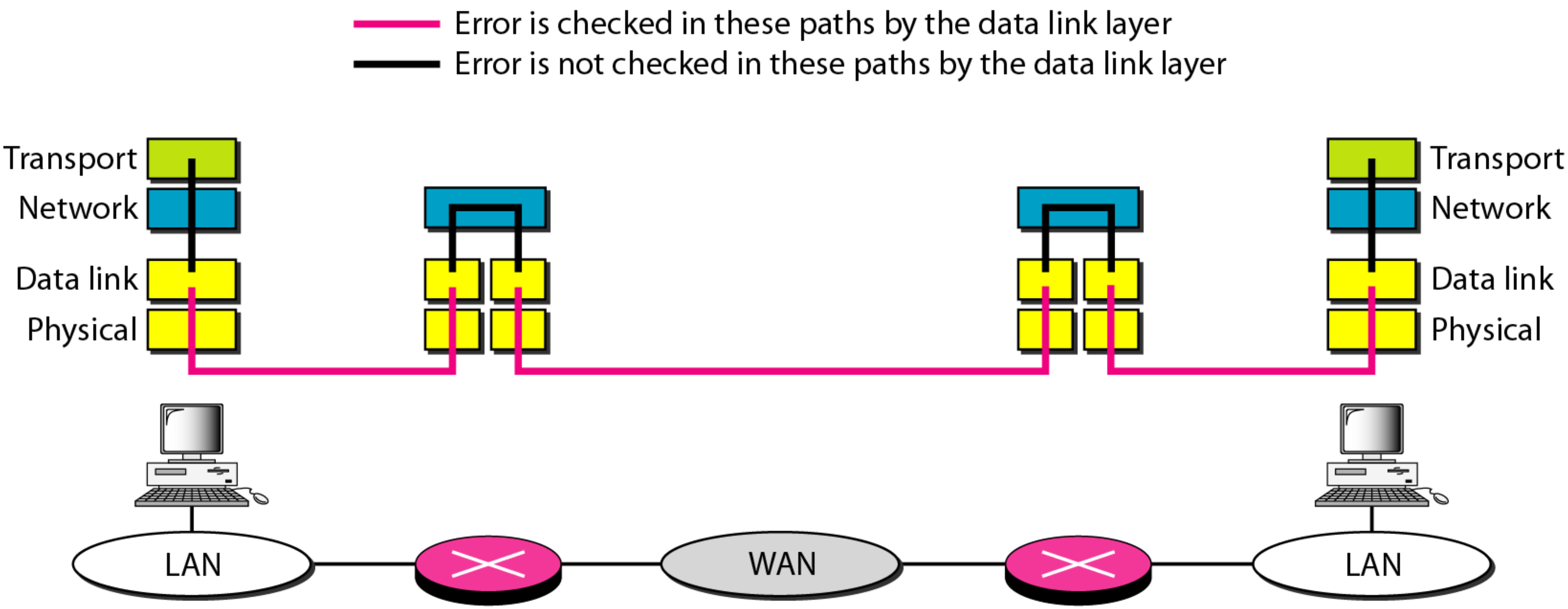
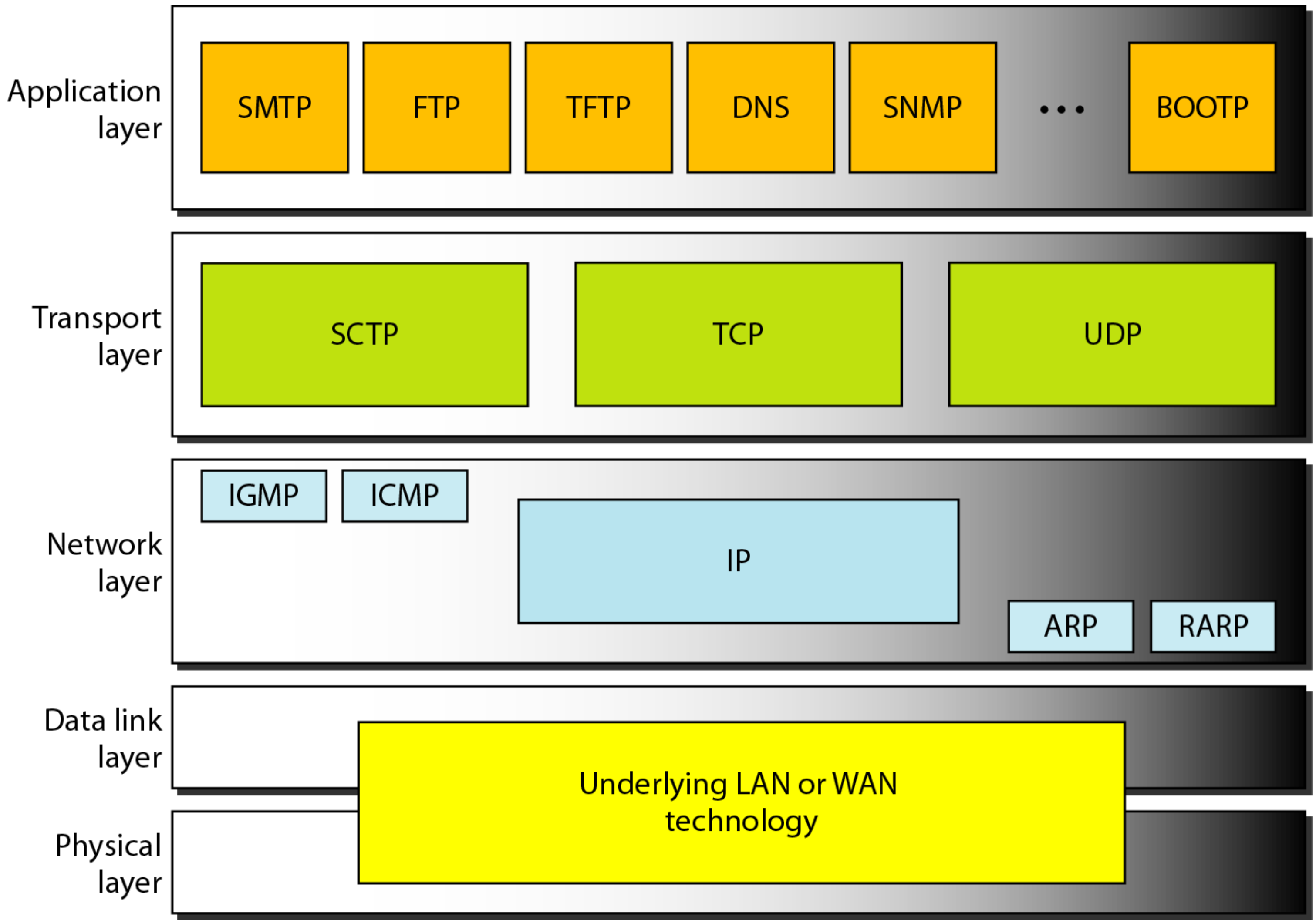


Figure 23.8 *Position of UDP, TCP, and SCTP in TCP/IP suite*



23-2 USER DATAGRAM PROTOCOL (UDP)

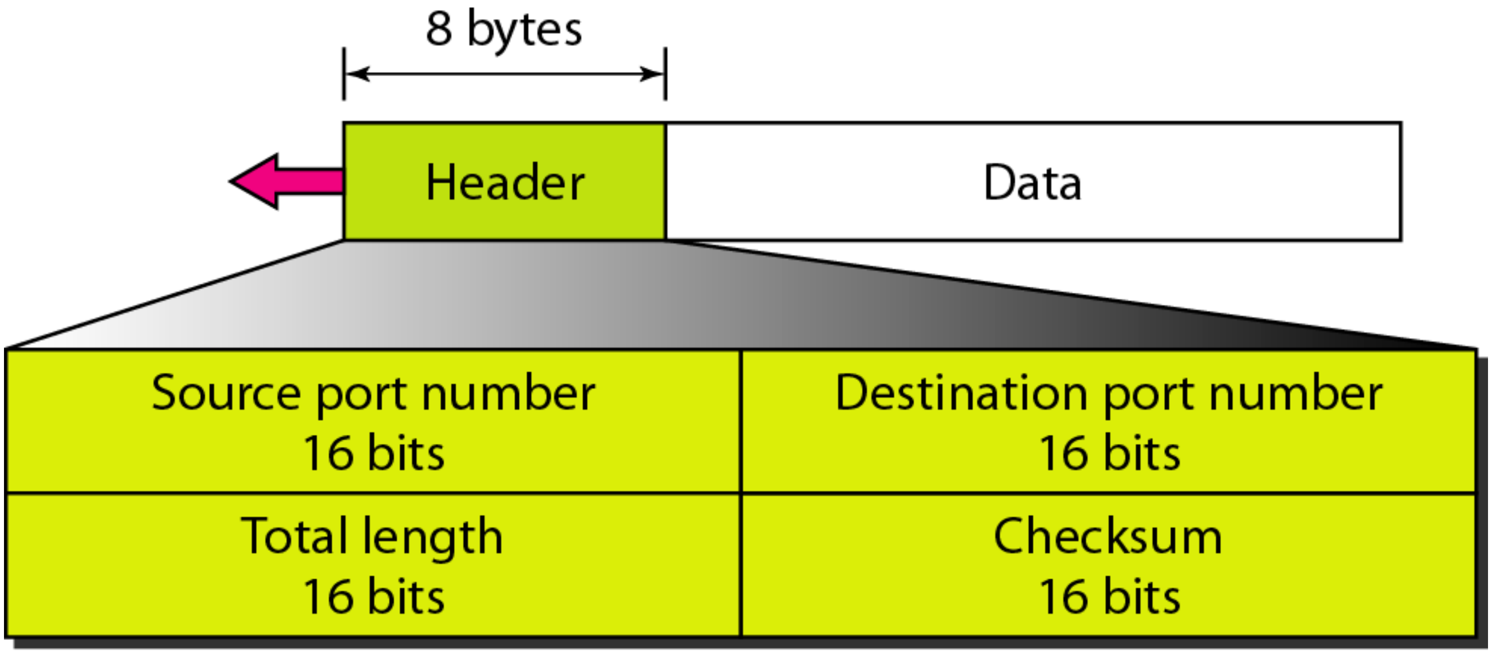
The User Datagram Protocol (UDP) is called a connectionless, unreliable transport protocol. It does not add anything to the services of IP except to provide process-to-process communication instead of host-to-host communication.

Well-Known Ports for UDP
User Datagram
Checksum
UDP Operation
Use of UDP

Table 23.1 *Well-known ports used with 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)

Figure 23.9 *User datagram format*



Note

UDP length
= IP length – IP header's length

Figure 23.10 *Pseudoheader for checksum calculation*

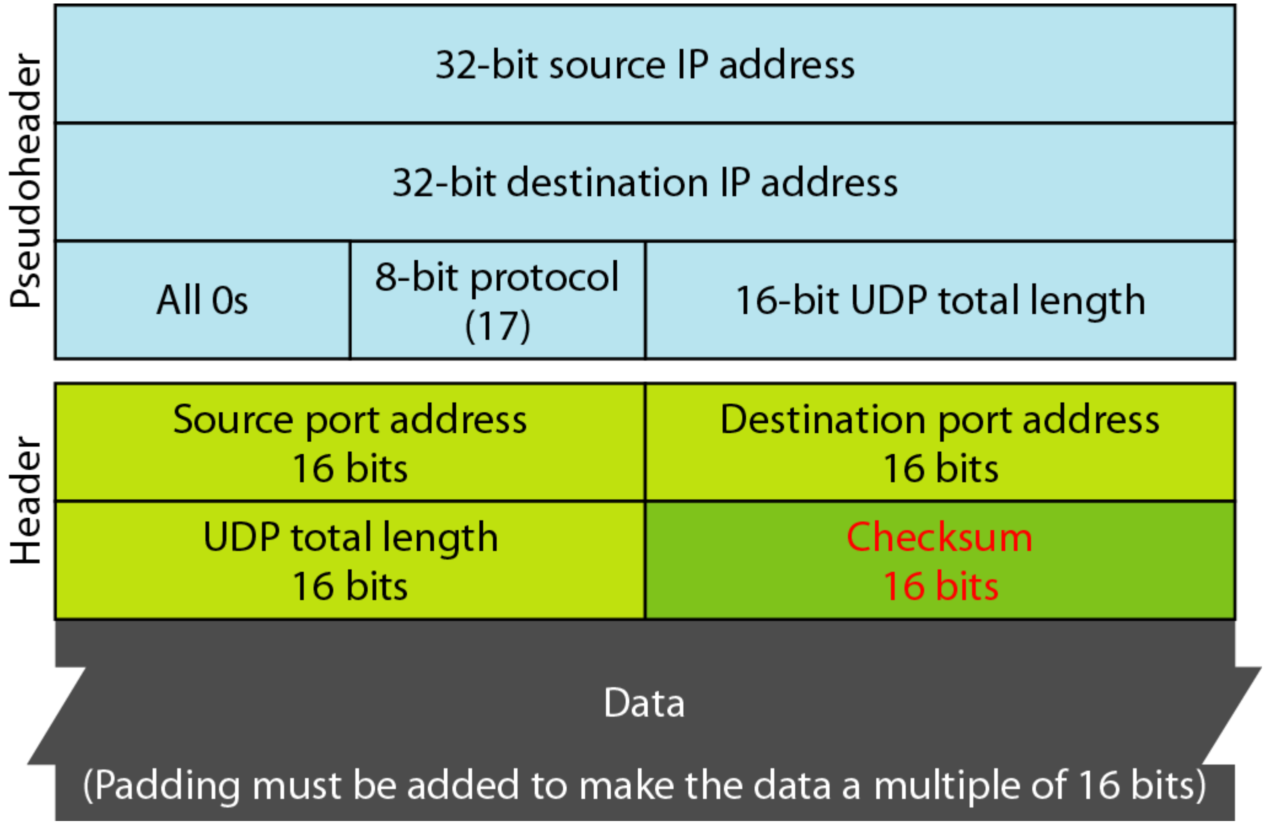
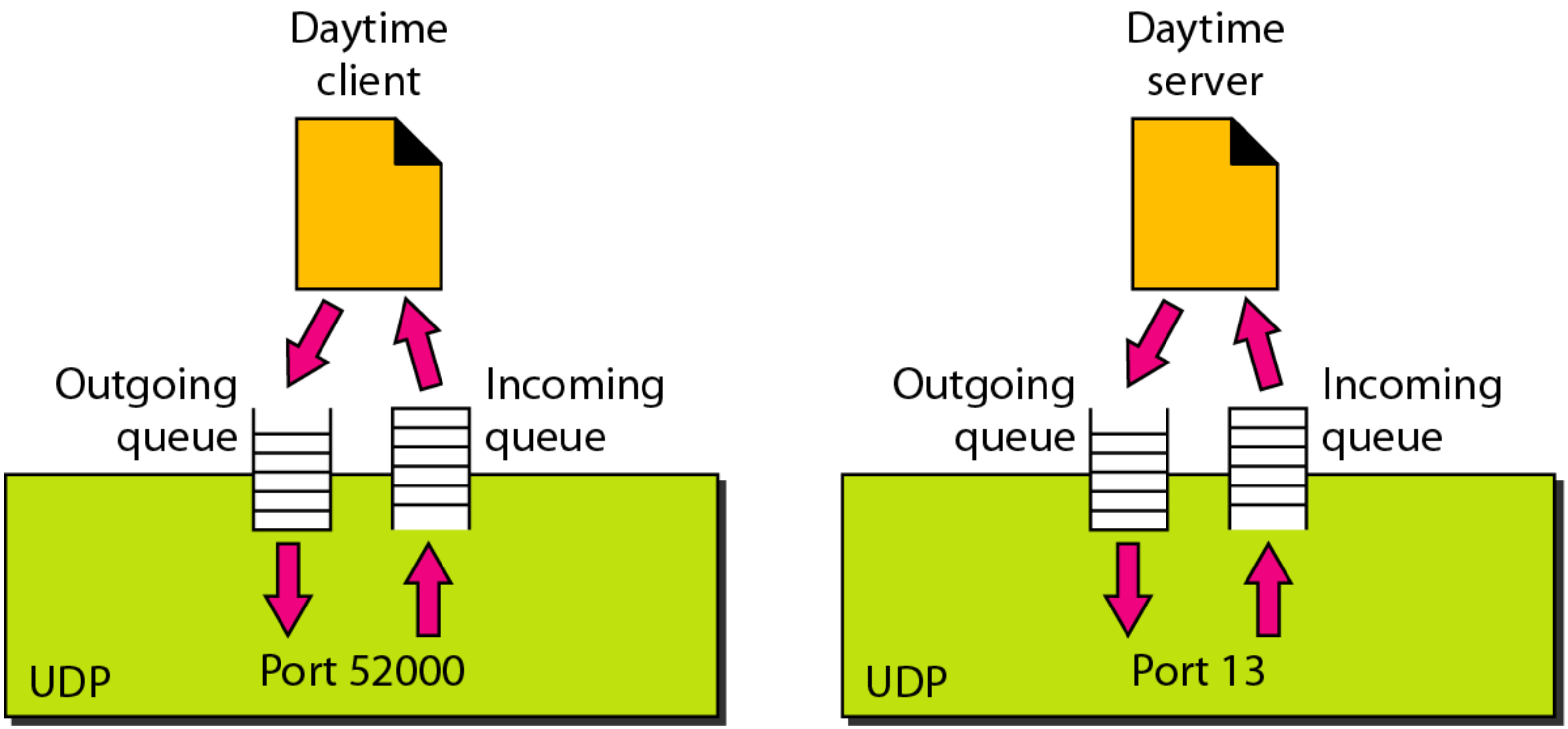


Figure 23.12 *Queues in UDP*



23-3 TCP

TCP is a connection-oriented protocol; it creates a virtual connection between two TCPs to send data. In addition, TCP uses flow and error control mechanisms at the transport level.

Table 23.2 *Well-known ports used by 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

Figure 23.13 *Stream delivery*

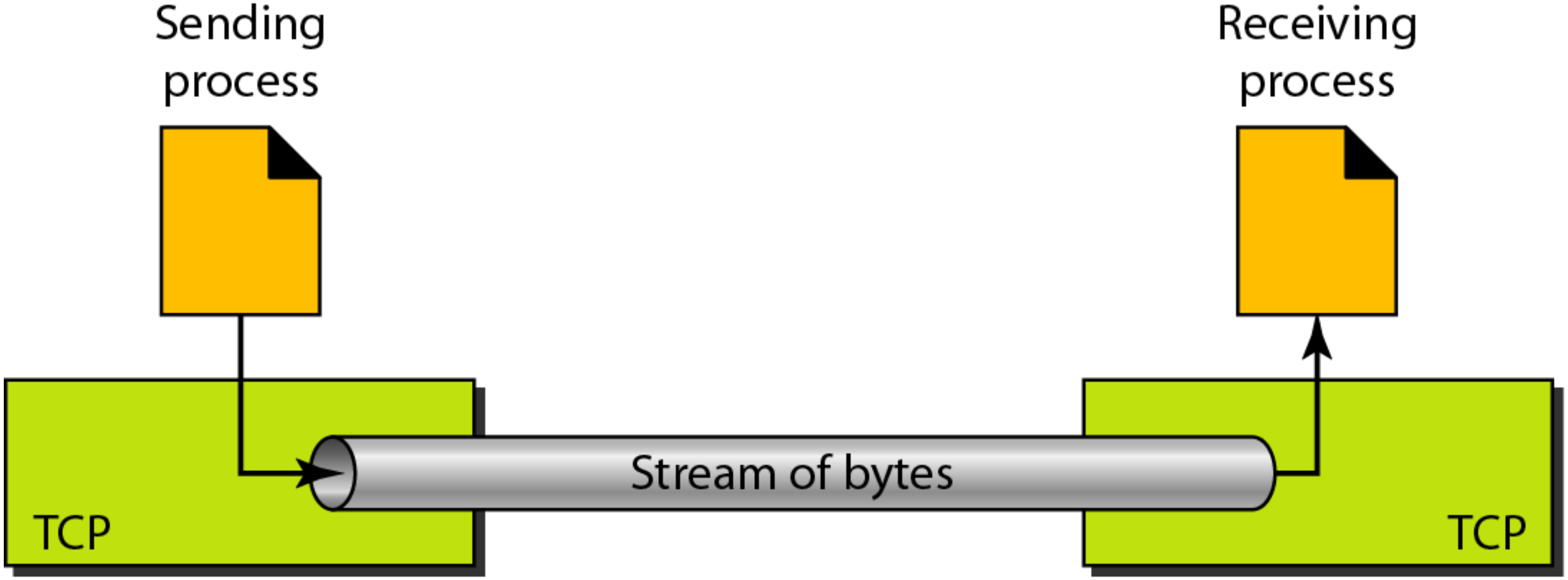


Figure 23.14 *Sending and receiving buffers*

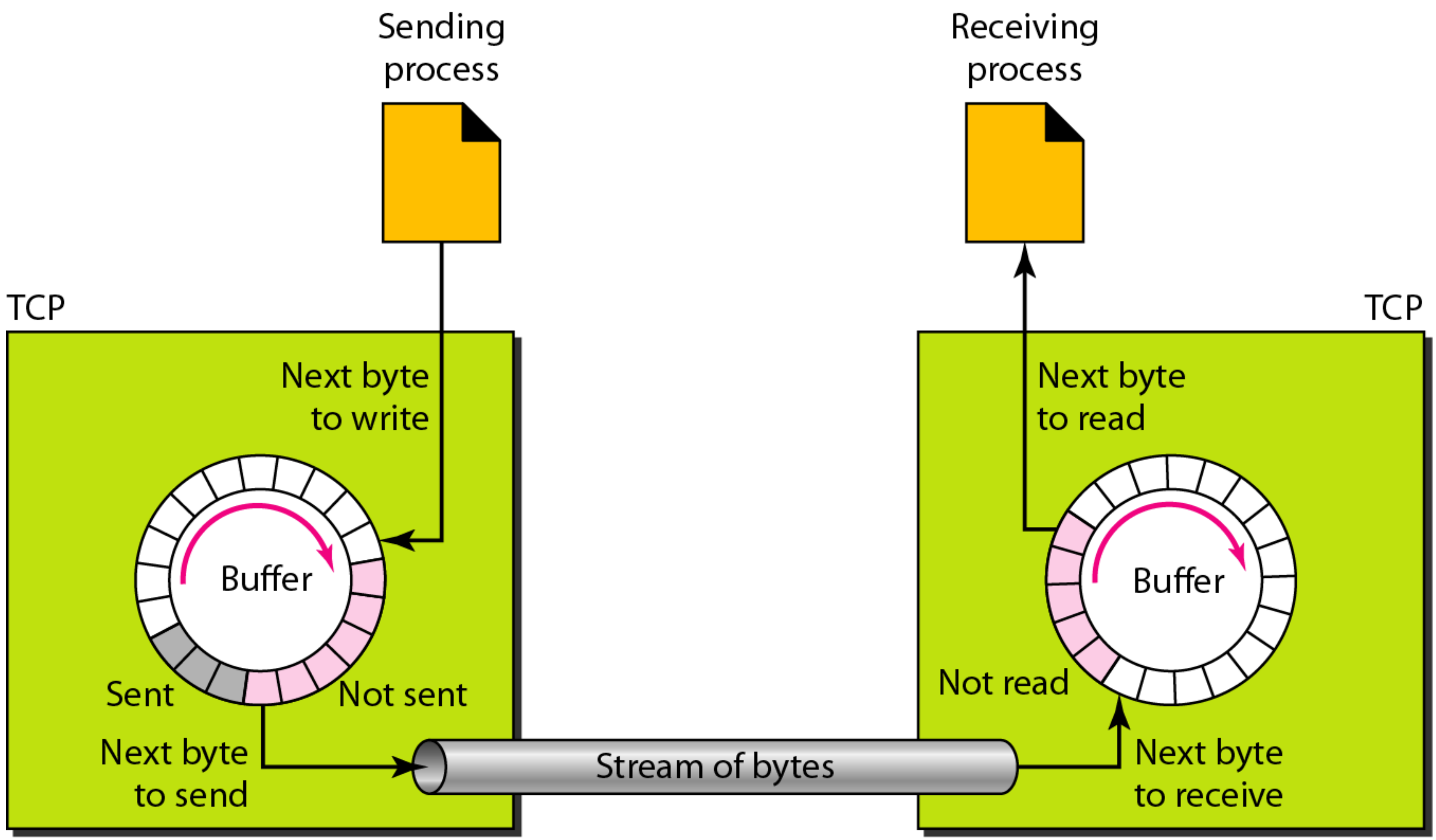
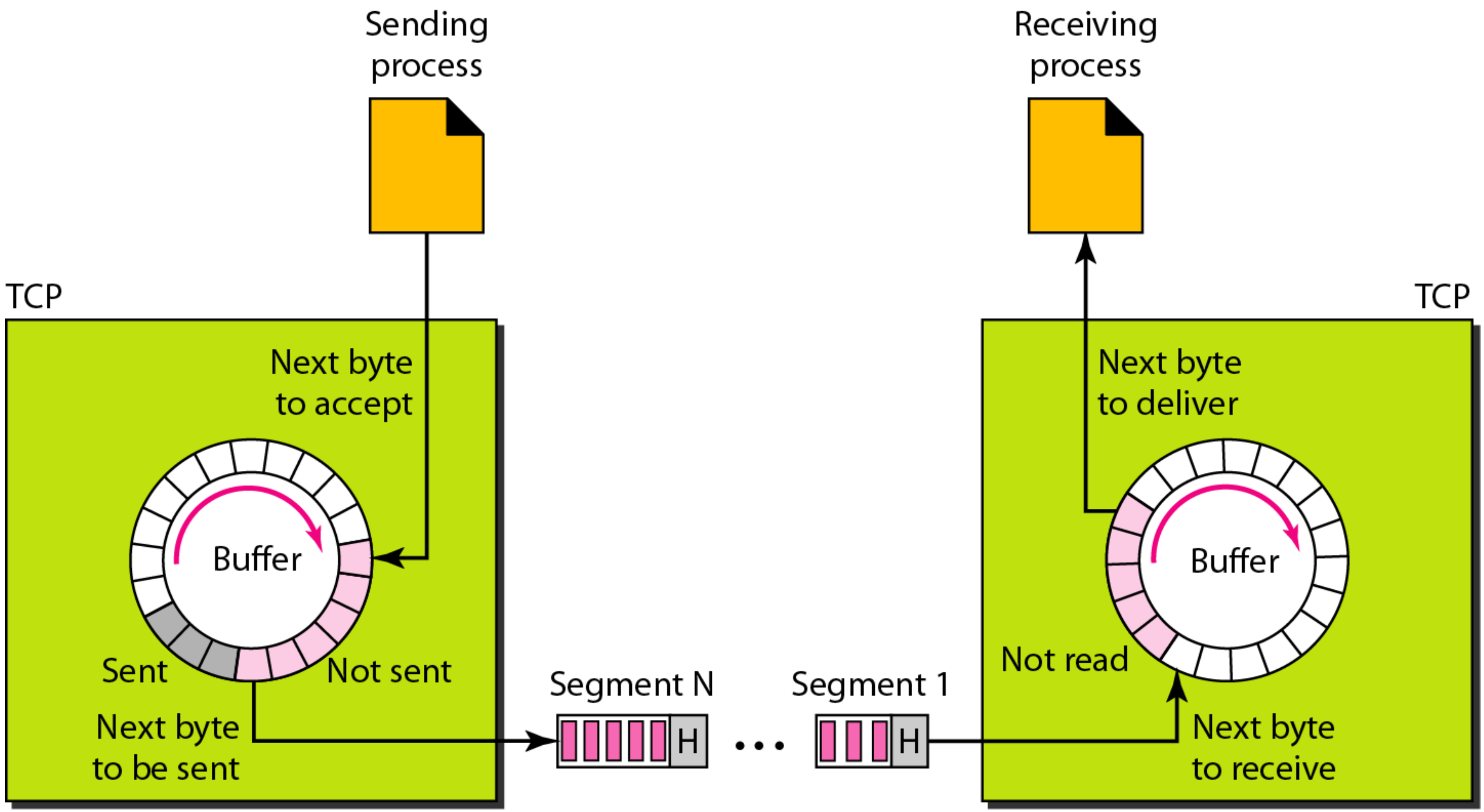


Figure 23.15 *TCP segments*



The bytes of data being transferred in each connection are numbered by TCP. The numbering starts with a randomly generated number.

Example 23.3

The following shows the sequence number for each segment:

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)

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.

Figure 23.16 *TCP segment format*

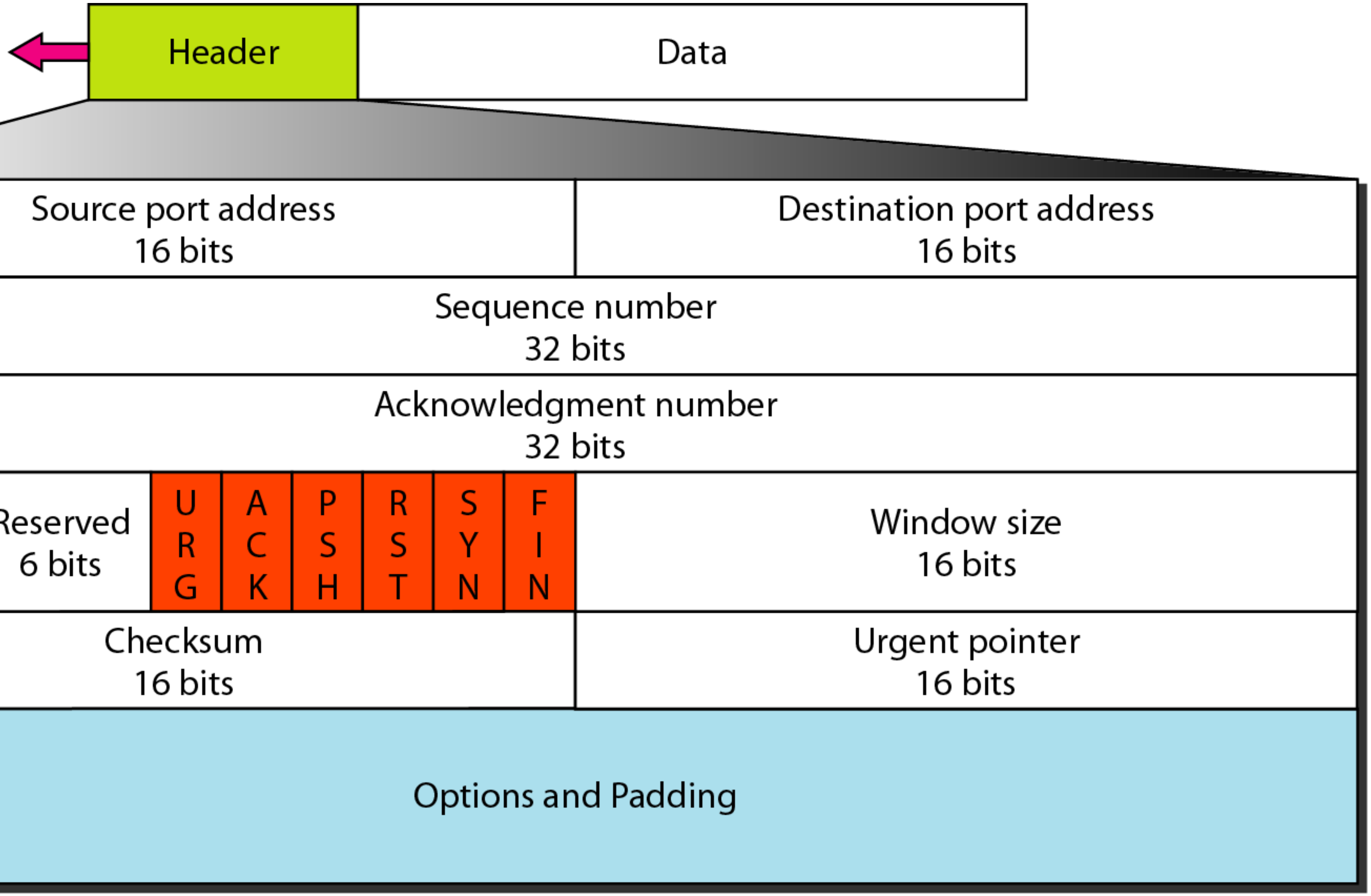


Figure 23.17 *Control field*

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

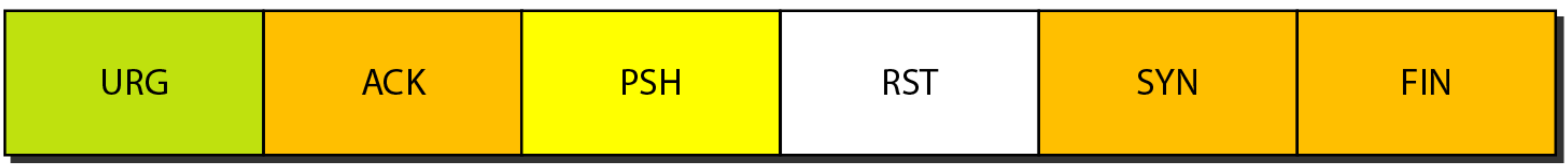
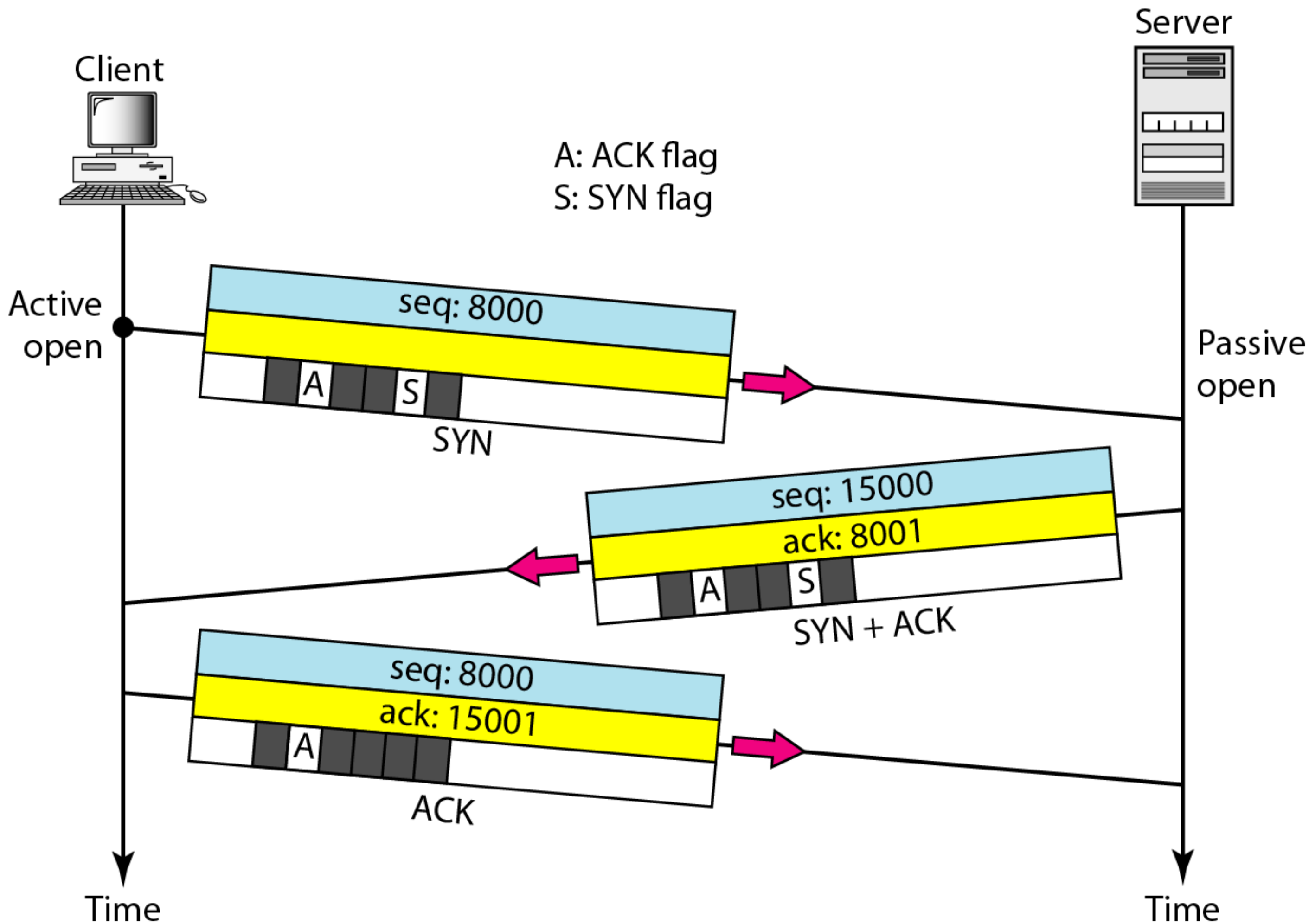


Table 23.3 ***Description of flags in the control field***

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

Figure 23.18 *Connection establishment using 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.

Figure 23.19 *Data transfer*

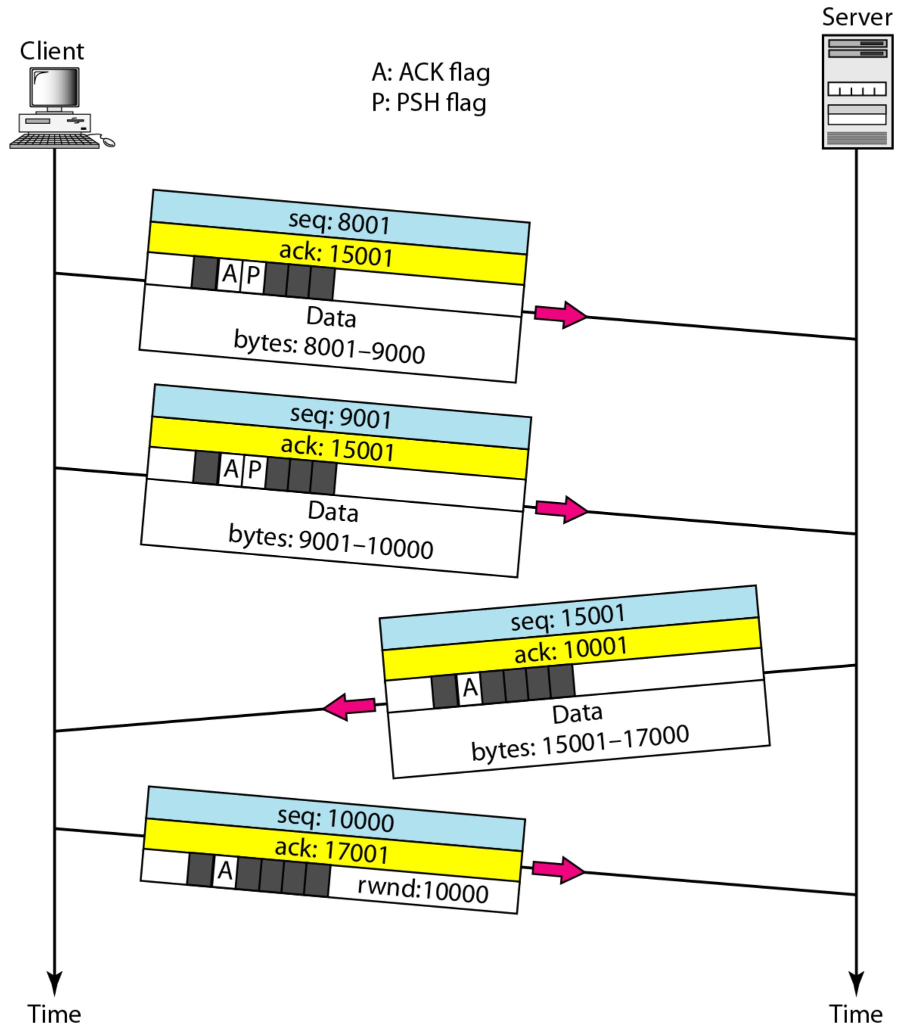
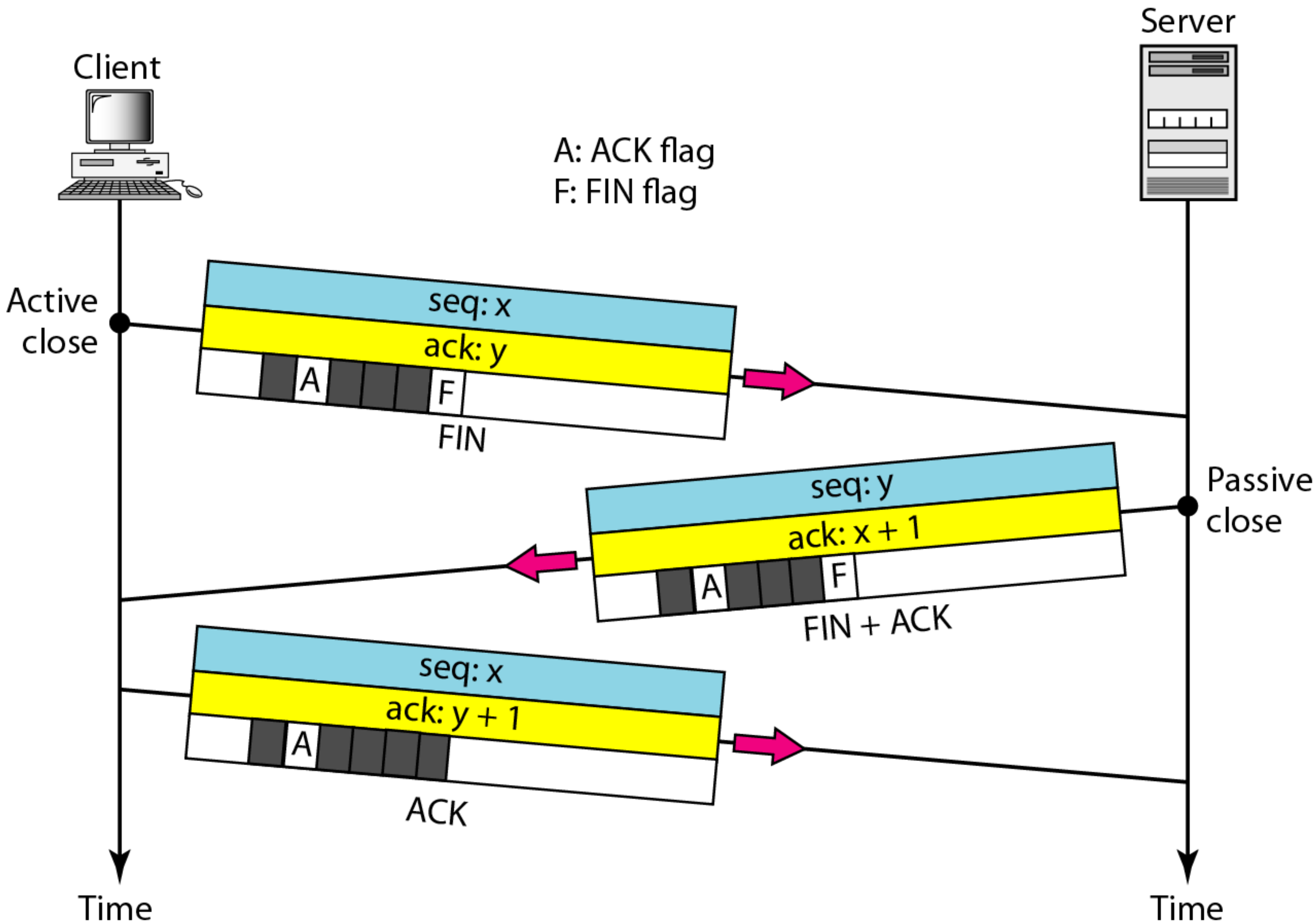


Figure 23.20 *Connection termination using three-way handshaking*



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.

Figure 23.21 *Half-close*

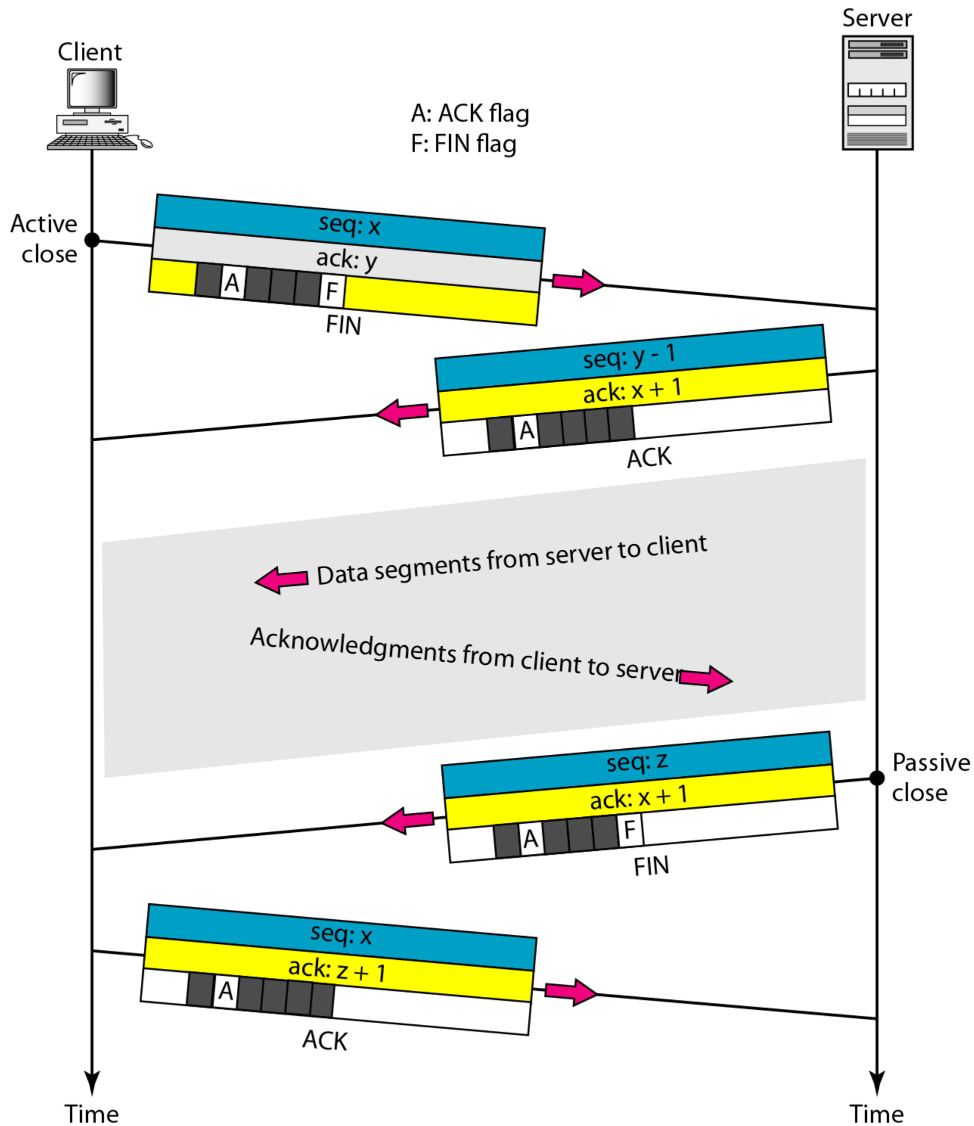
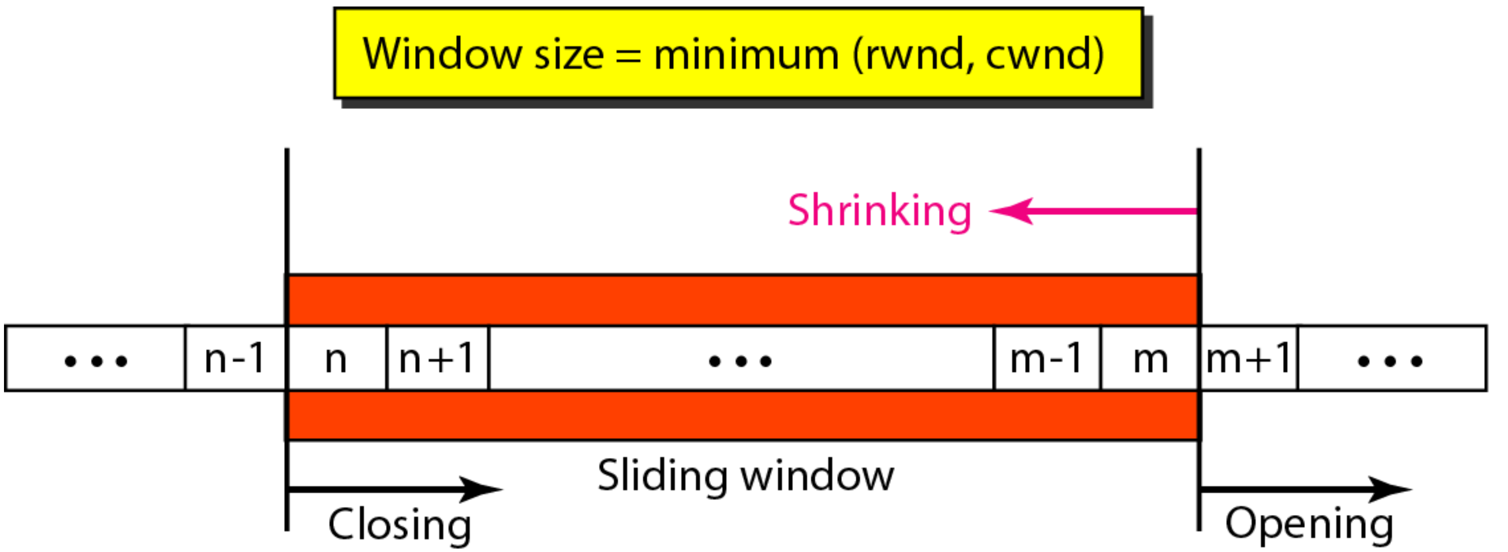


Figure 23.22 *Sliding window*



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.

ACK segments do not consume sequence numbers and are not acknowledged.

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.

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.

Figure 23.24 *Normal operation*

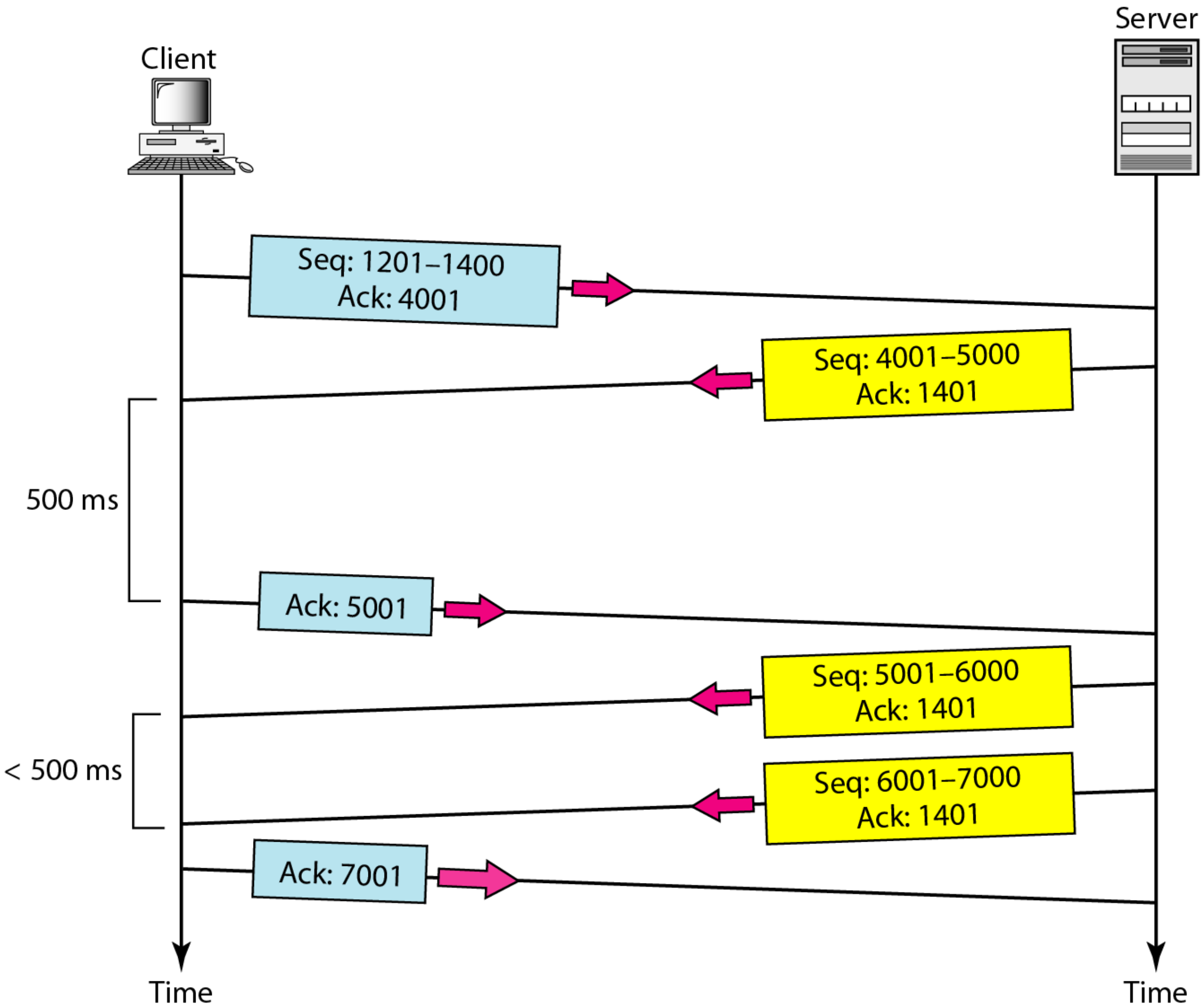
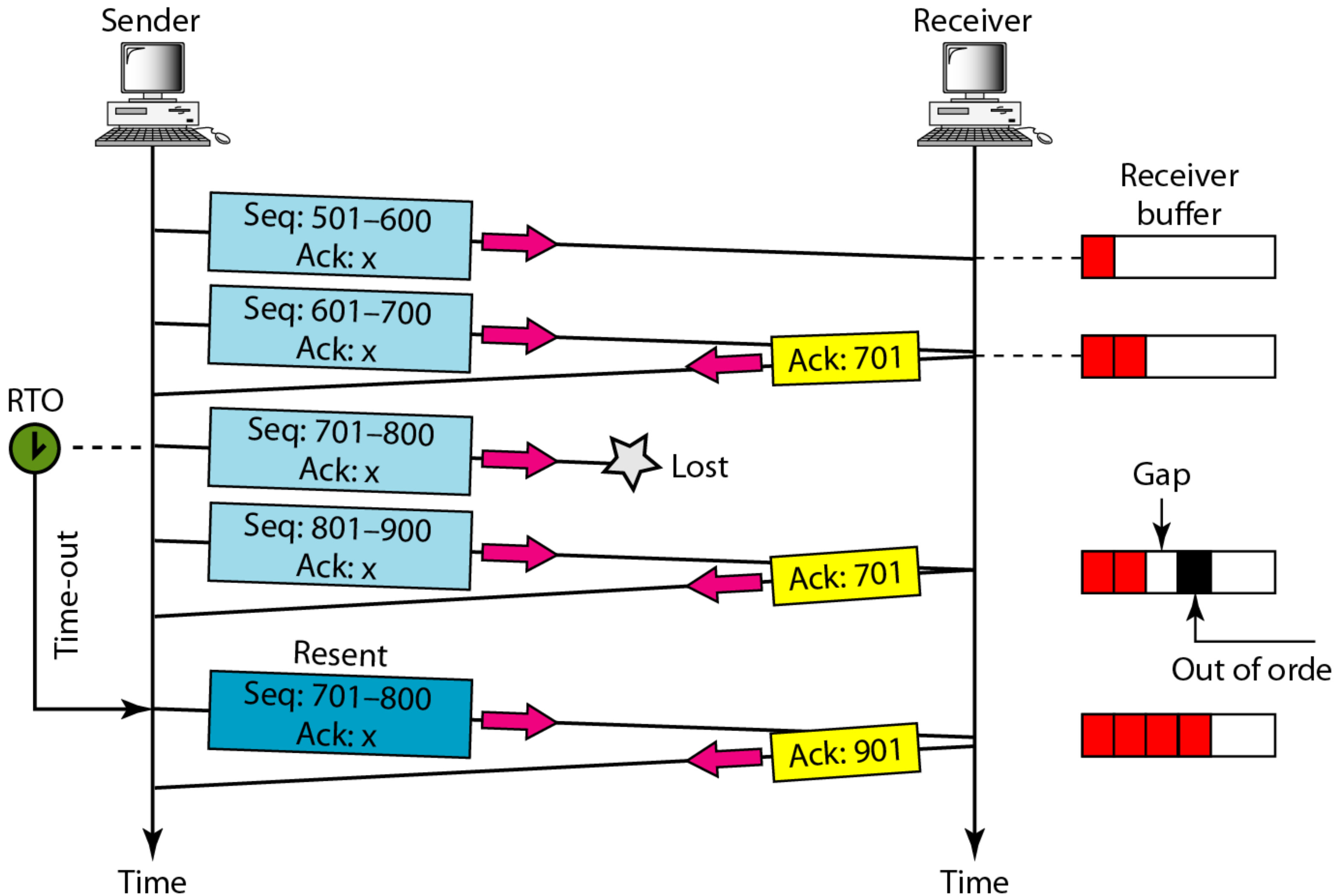
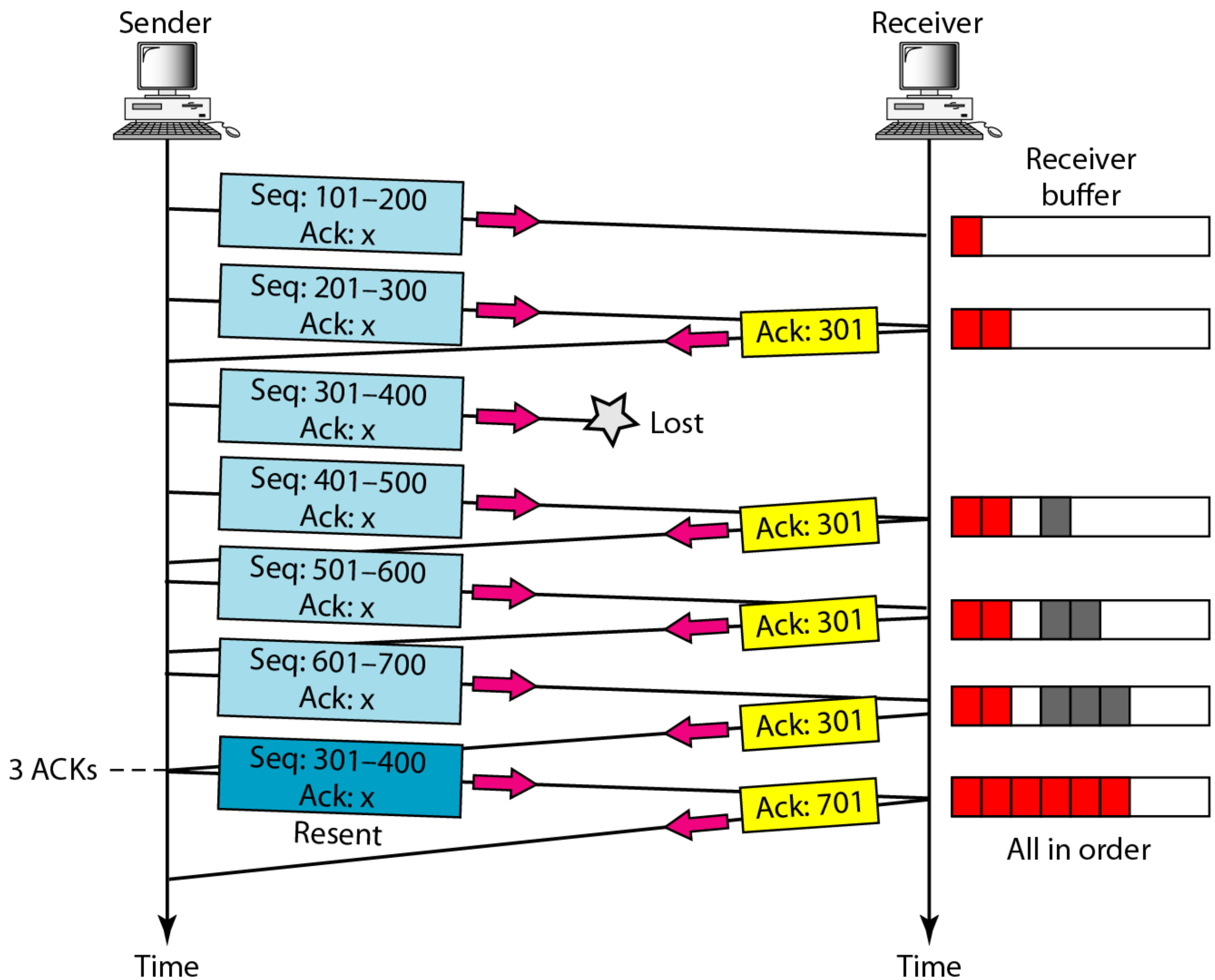


Figure 23.25 *Lost segment*



The receiver TCP delivers only ordered data to the process.

Figure 23.26 *Fast retransmission*



23-4 SCTP

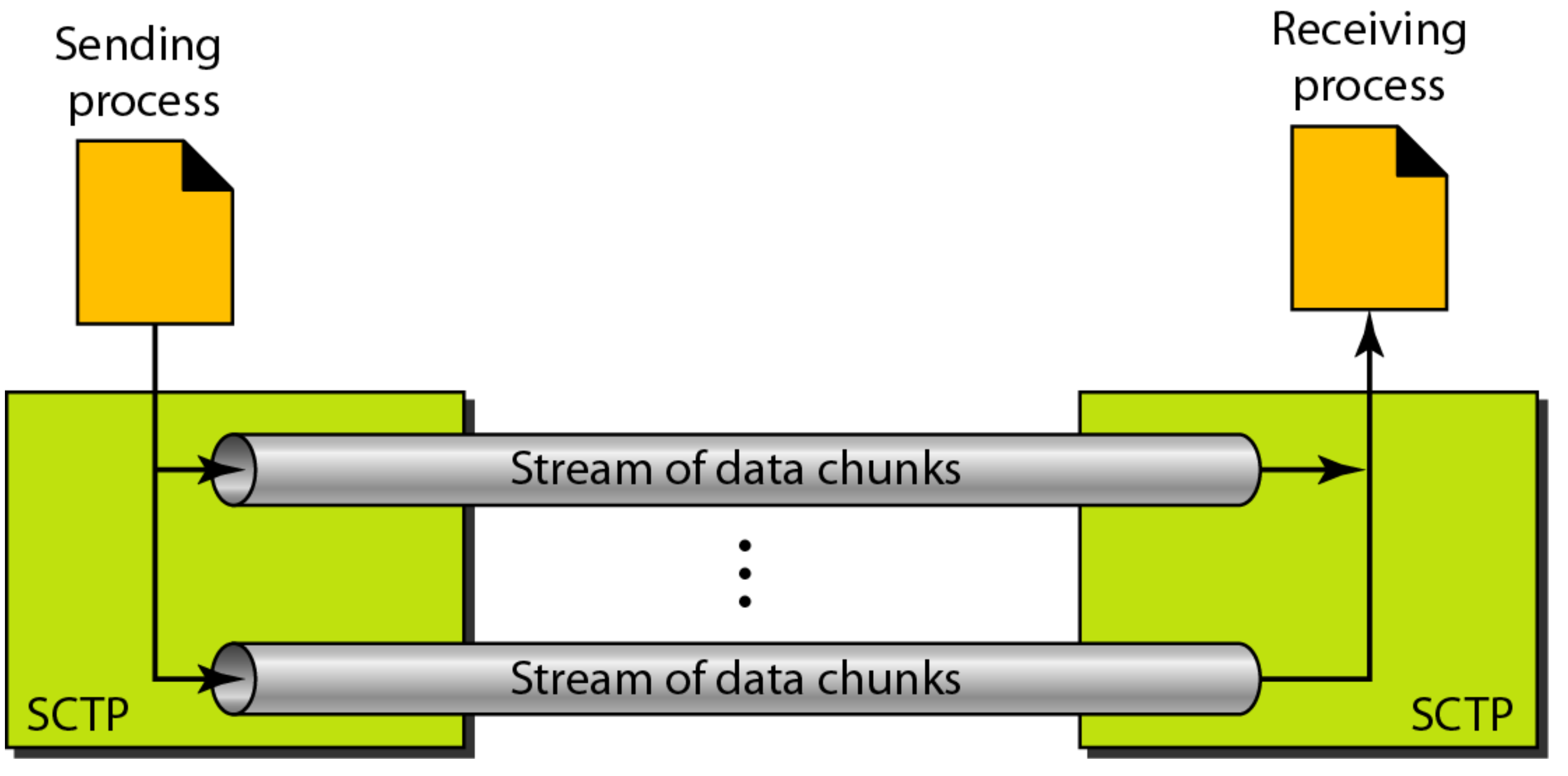
Stream Control Transmission Protocol (SCTP) is a new reliable, message-oriented transport layer protocol. SCTP, however, is mostly designed for Internet applications that have recently been introduced. These new applications need a more sophisticated service than TCP can provide.

SCTP is a message-oriented, reliable protocol that combines the best features of UDP and TCP.

Table 23.4 ***Some SCTP applications***

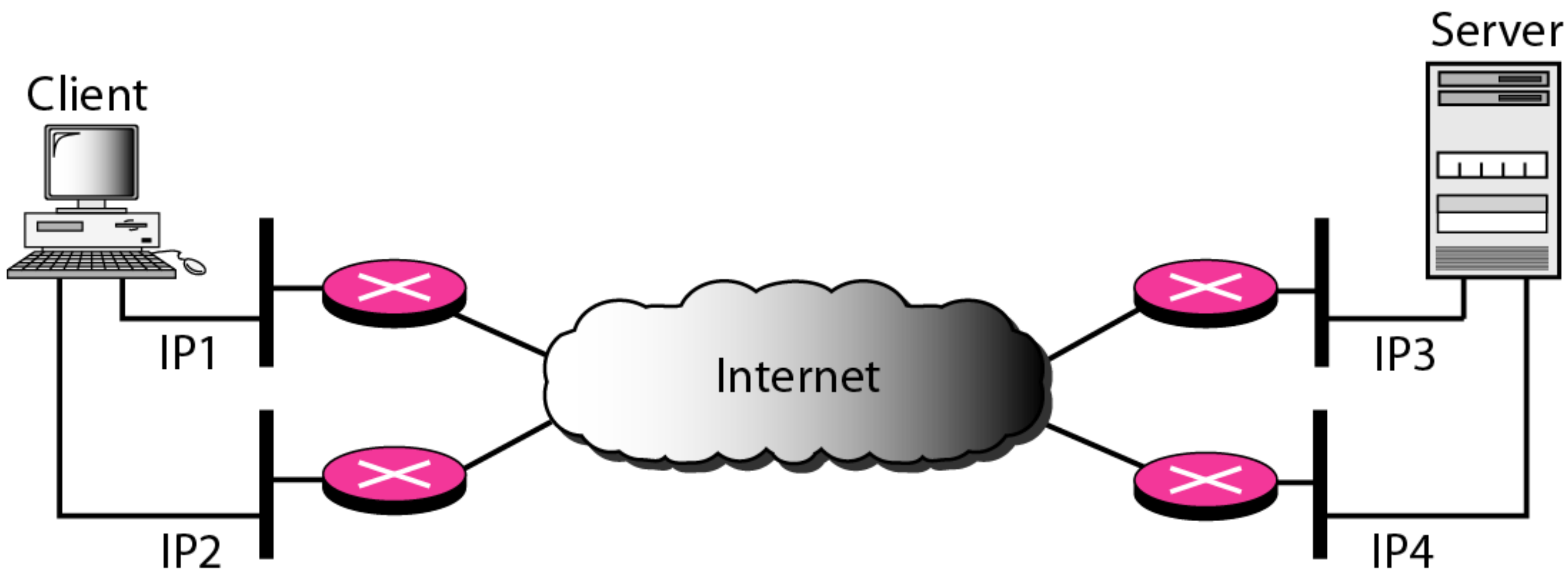
Protocol	*Port Number*	*Description*
IUA	9990	ISDN over IP
M2UA	2904	SS7 telephony signaling
M3UA	2905	SS7 telephony signaling
H.248	2945	Media gateway control
H.323	1718, 1719, 1720, 11720	IP telephony
SIP	5060	IP telephony

Figure 23.27 *Multiple-stream concept*



An association in SCTP can involve multiple streams.

Figure 23.28 *Multihoming concept*



SCTP association allows multiple IP addresses for each end.

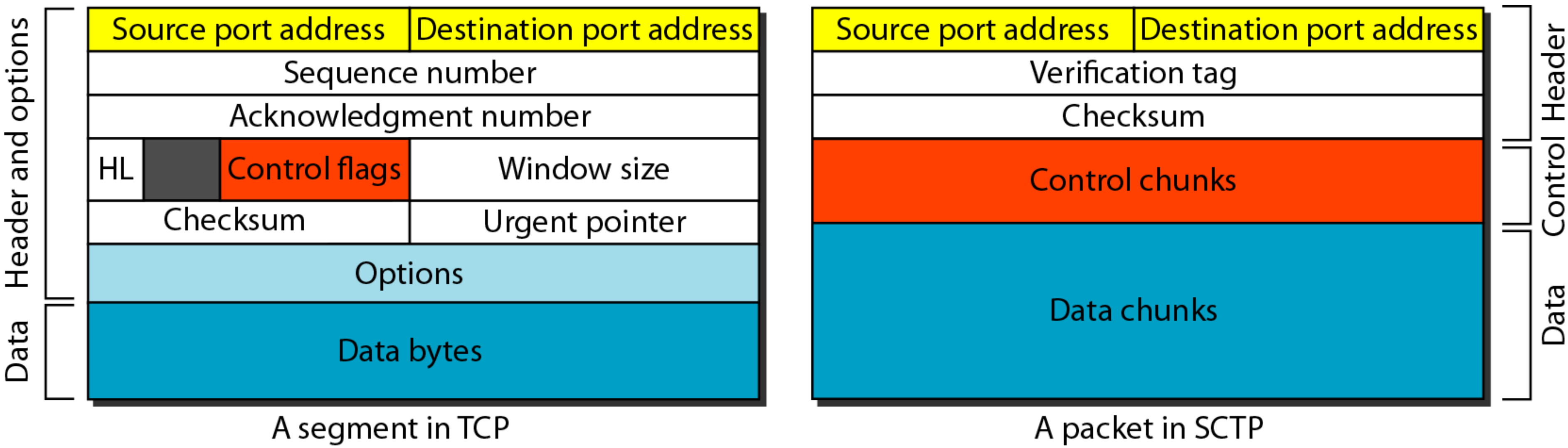
In SCTP, a data chunk is numbered using a TSN.

To distinguish between different streams, SCTP uses an SI.

To distinguish between different data chunks belonging to the same stream, SCTP uses SSNs.

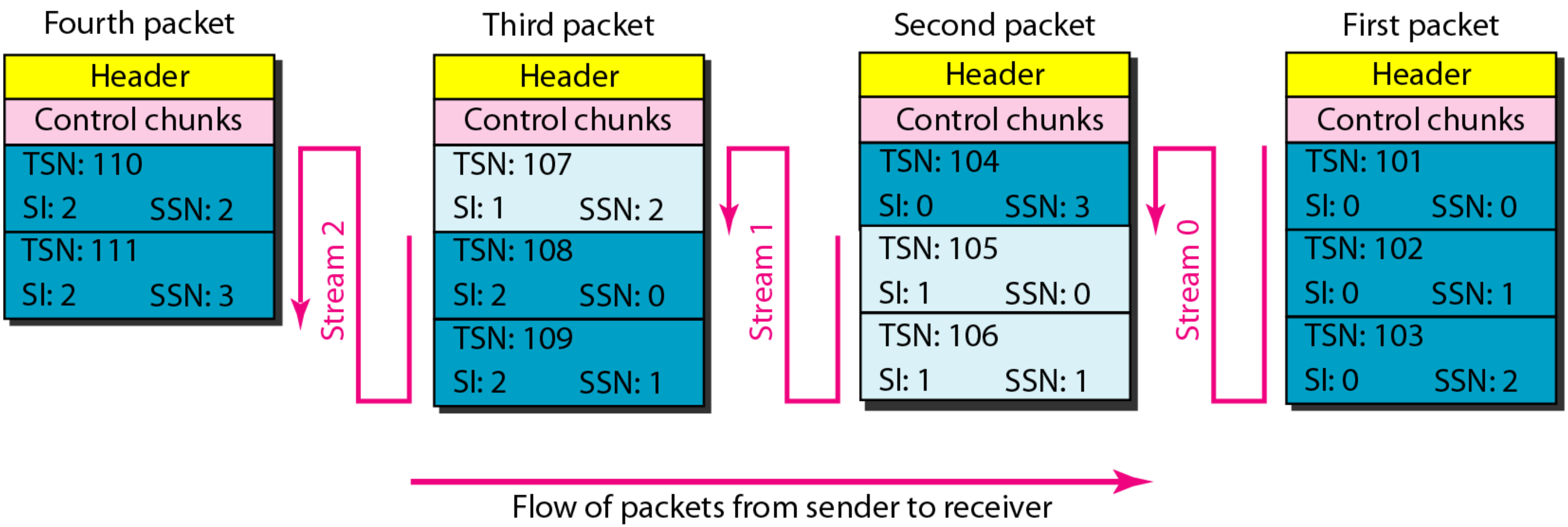
TCP has segments; SCTP has packets.

Figure 23.29 *Comparison between a TCP segment and an SCTP packet*



In SCTP, control information and data information are carried in separate chunks.

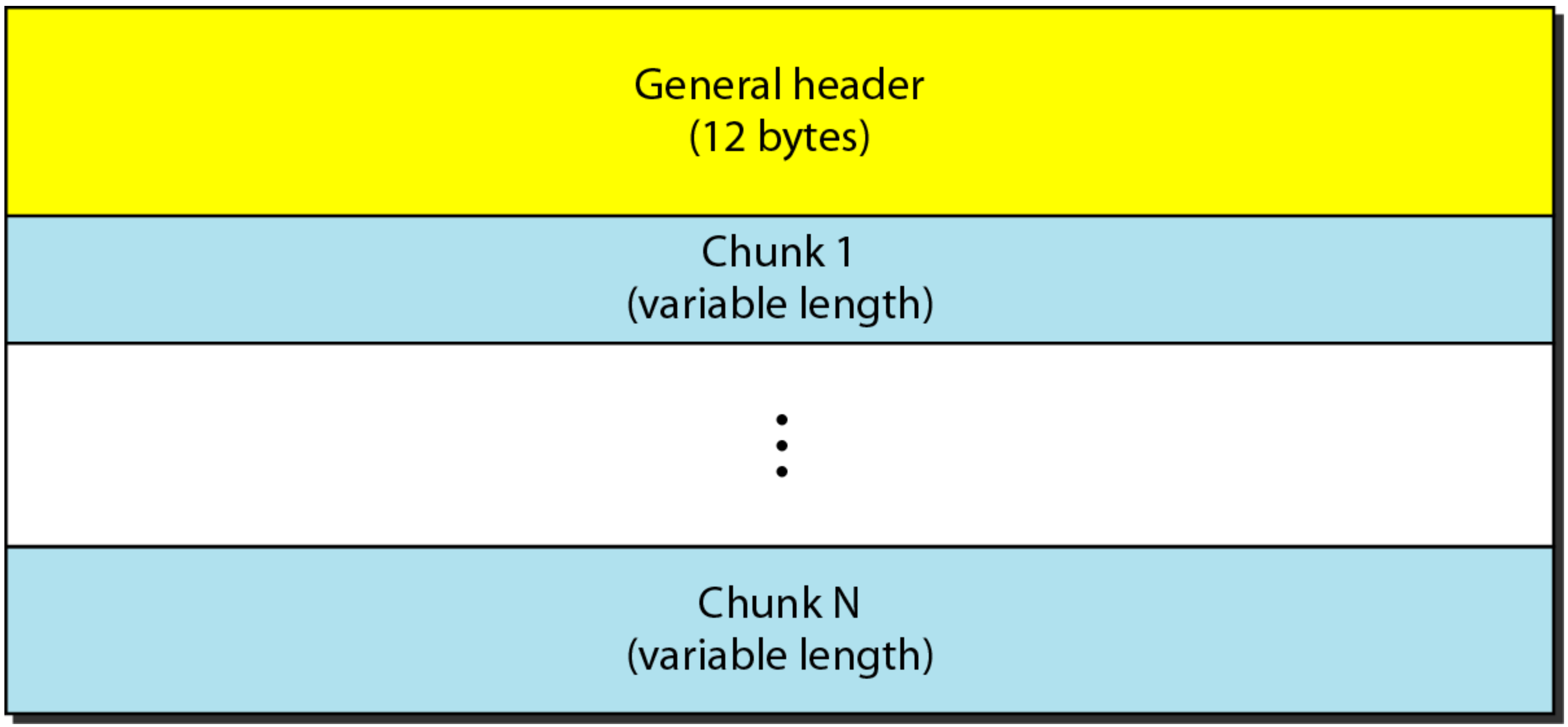
Figure 23.30 *Packet, data chunks, and streams*



Data chunks are identified by three items: TSN, SI, and SSN. TSN is a cumulative number identifying the association; SI defines the stream; SSN defines the chunk in a stream.

In SCTP, acknowledgment numbers are used to acknowledge only data chunks; control chunks are acknowledged by other control chunks if necessary.

Figure 23.31 *SCTP packet format*



In an SCTP packet, control chunks come before data chunks.

Figure 23.32 *General header*

Source port address 16 bits	Destination port address 16 bits
Verification tag 32 bits	
Checksum 32 bits	

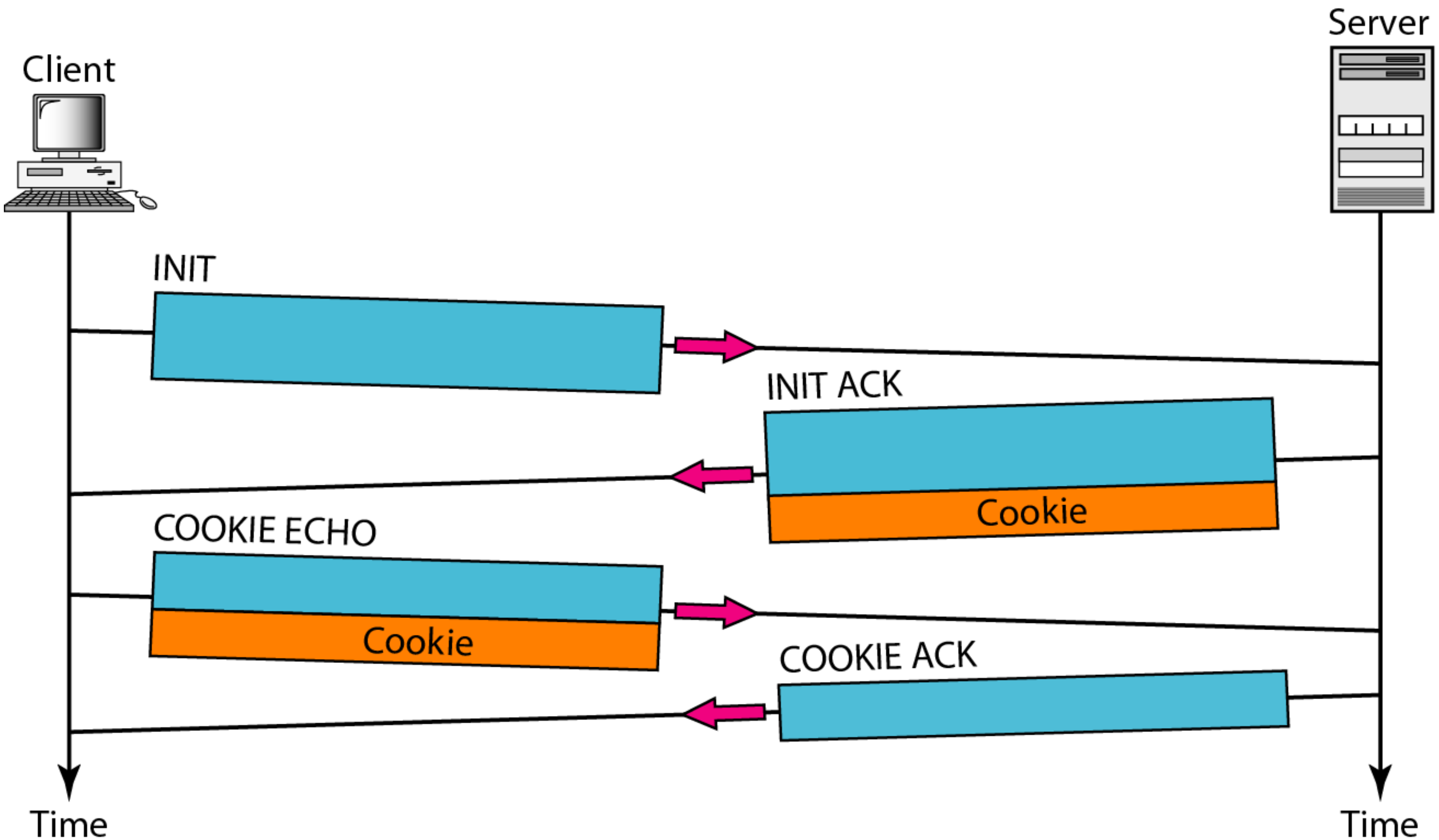
Table 23.5 *Chunks*

Type	*Chunk*	*Description*
0	**DATA**	User data
1	**INIT**	Sets up an association
2	**INIT ACK**	Acknowledges INIT chunk
3	**SACK**	Selective acknowledgment
4	**HEARTBEAT**	Probes the peer for liveliness
5	**HEARTBEAT ACK**	Acknowledges HEARTBEAT chunk
6	**ABORT**	Aborts an association
7	**SHUTDOWN**	Terminates an association
8	**SHUTDOWN ACK**	Acknowledges SHUTDOWN chunk
9	**ERROR**	Reports errors without shutting down
10	**COOKIE ECHO**	Third packet in association establishment
11	**COOKIE ACK**	Acknowledges COOKIE ECHO chunk
14	**SHUTDOWN COMPLETE**	Third packet in association termination
192	**FORWARD TSN**	For adjusting cumulative TSN

A connection in SCTP is called an association.

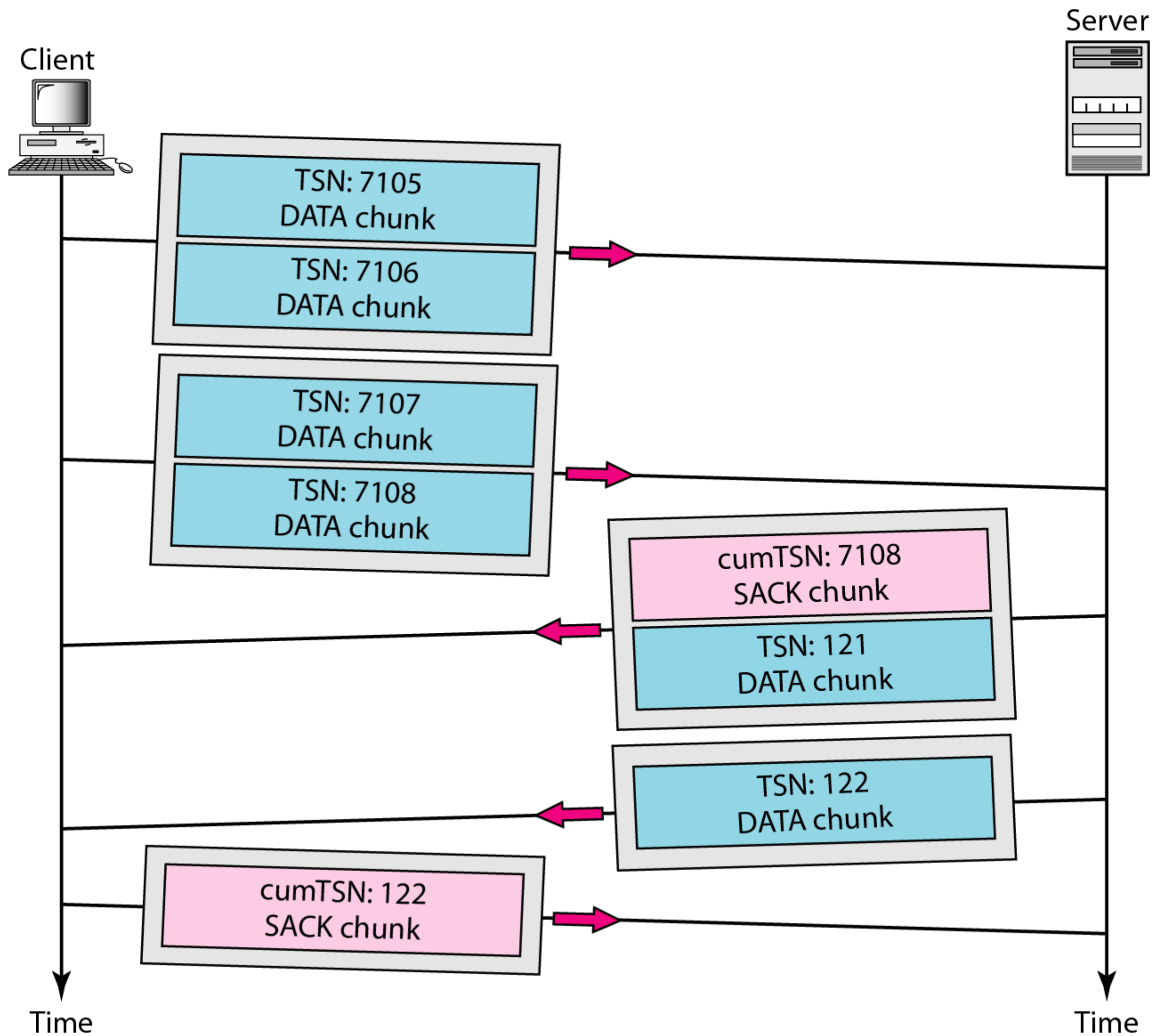
No other chunk is allowed in a packet carrying an INIT or INIT ACK chunk. A COOKIE ECHO or a COOKIE ACK chunk can carry data chunks.

Figure 23.33 *Four-way handshaking*



In SCTP, only DATA chunks consume TSNs;
DATA chunks are the only chunks that are acknowledged.

Figure 23.34 *Simple data transfer*



The acknowledgment in SCTP defines the cumulative TSN, the TSN of the last data chunk received in order.

Figure 23.35 *Association termination*