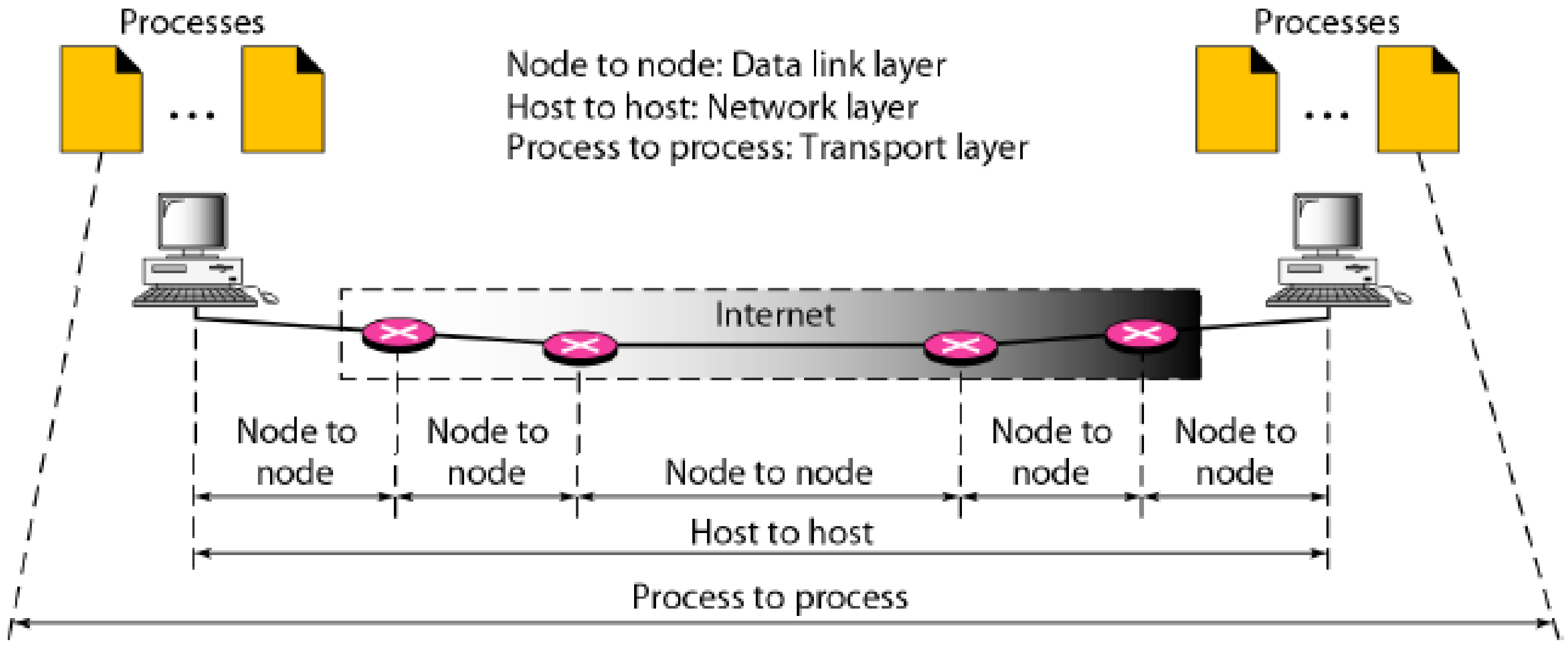
Transport Layer

Transport Layer

The transport layer is responsible for the delivery of a message from one process to another

Types of Data Deliveries

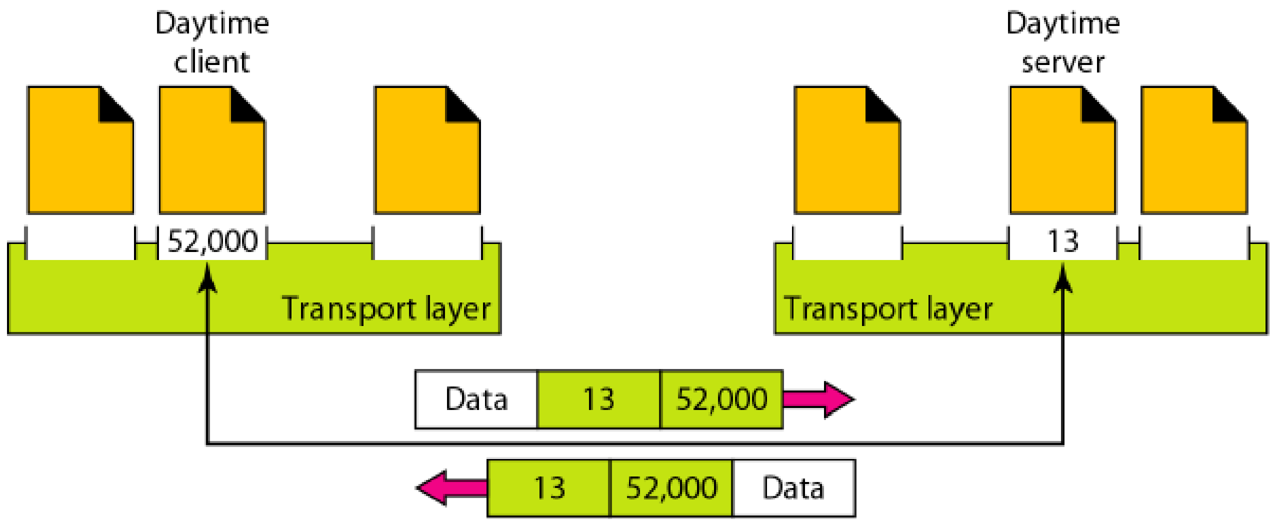


Client/Server Paradigm

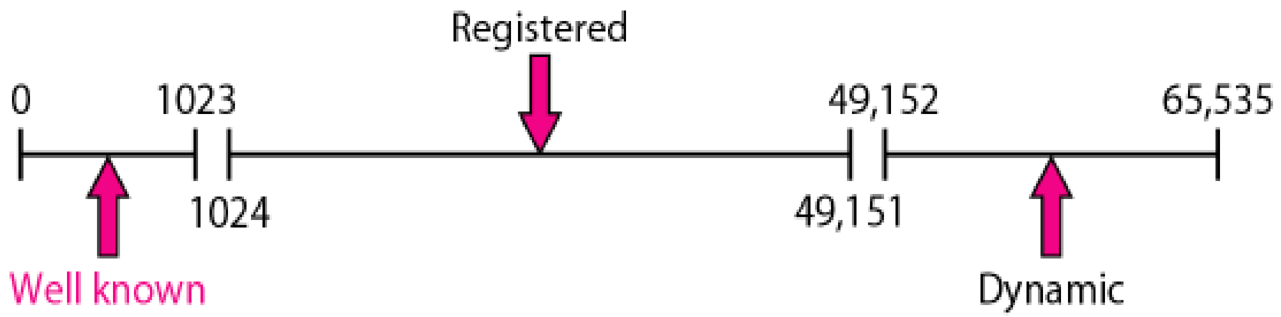
- An application program on the local host, called the client, needs services from an application program on the remote host, called a server

Port Number

- 16-bit integers between 0 and 65,535
- Client program defines itself with a port number (**ephemeral port number**), chosen randomly by the transport layer software running in the client host
- Servers use universal port numbers (**well-known port number**)



Port Number (2)



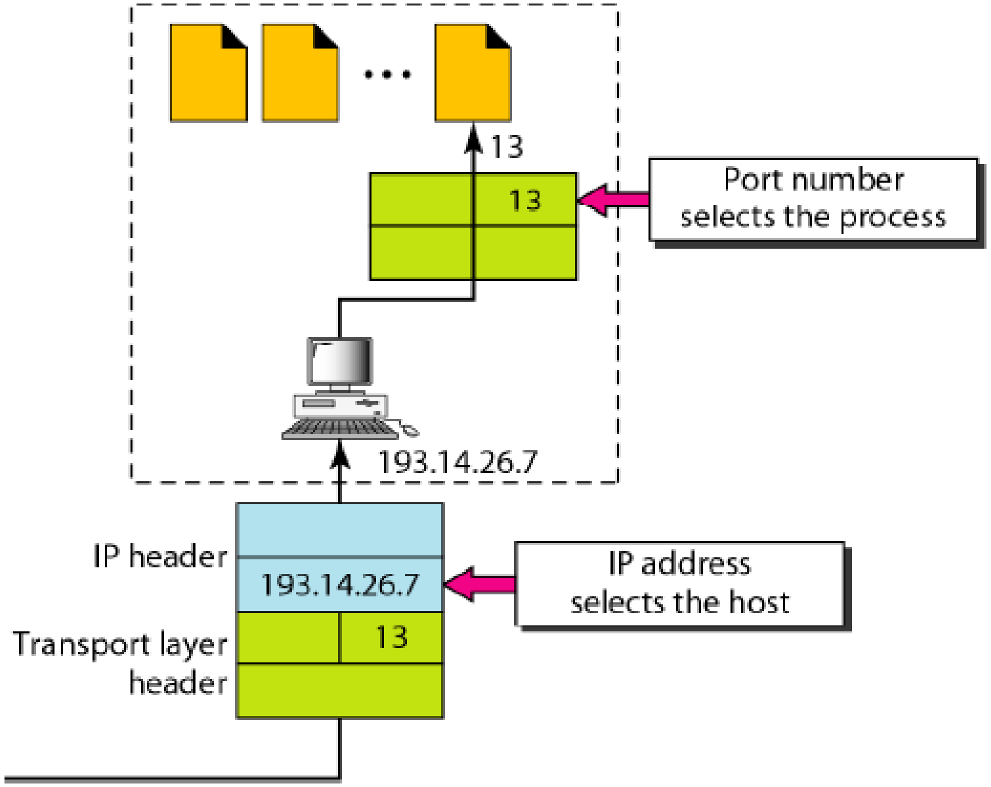
- **Well-known ports** – ports ranging from 0 to 1023 and are assigned and controlled by IANA
- **Registered ports** – ports ranging from 1024 to 49,151 and are not assigned or controlled by IANA. They can only be registered to prevent duplication
- **Dynamic** (or private) **ports** – ports ranging from 49,152 to 65,535 and are neither controlled nor registered. They can be used by any process

Port Number (3)

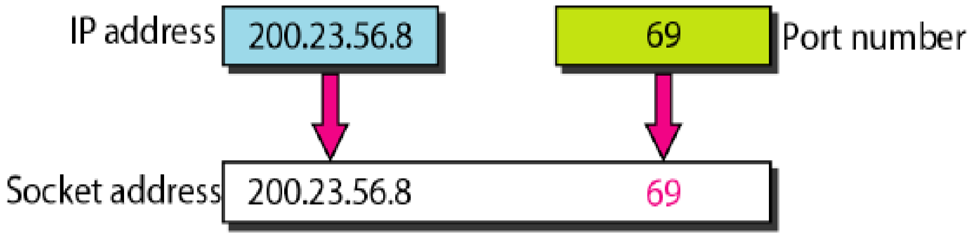
- In UNIX, the well-known ports are stored in a file called */etc/services*. Each line gives the name of the server and the well-know port number. Use grep utility to extract the line to get corresponding desired application

Socket Address

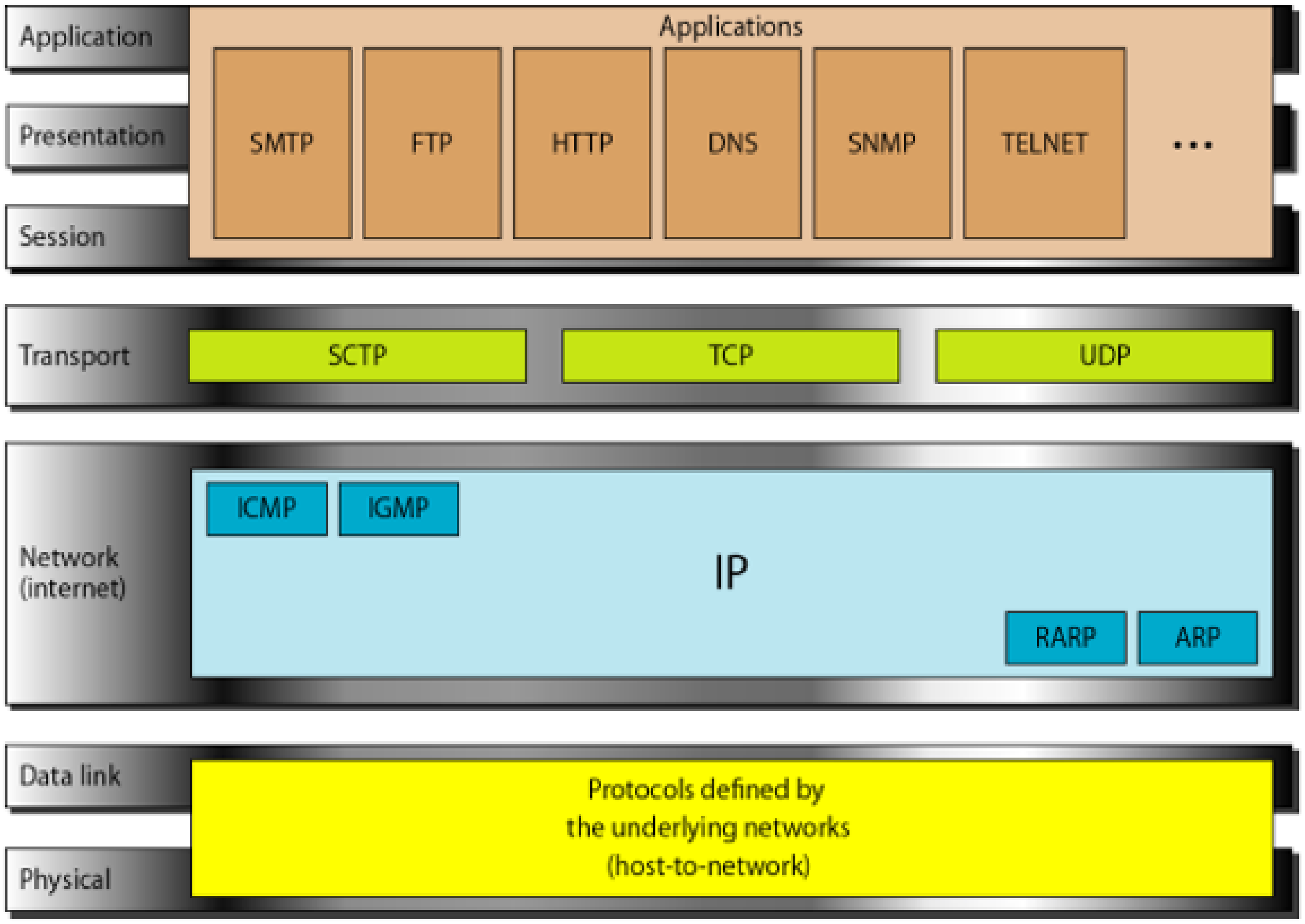
IP Address vs Port Numbers



Socket Address



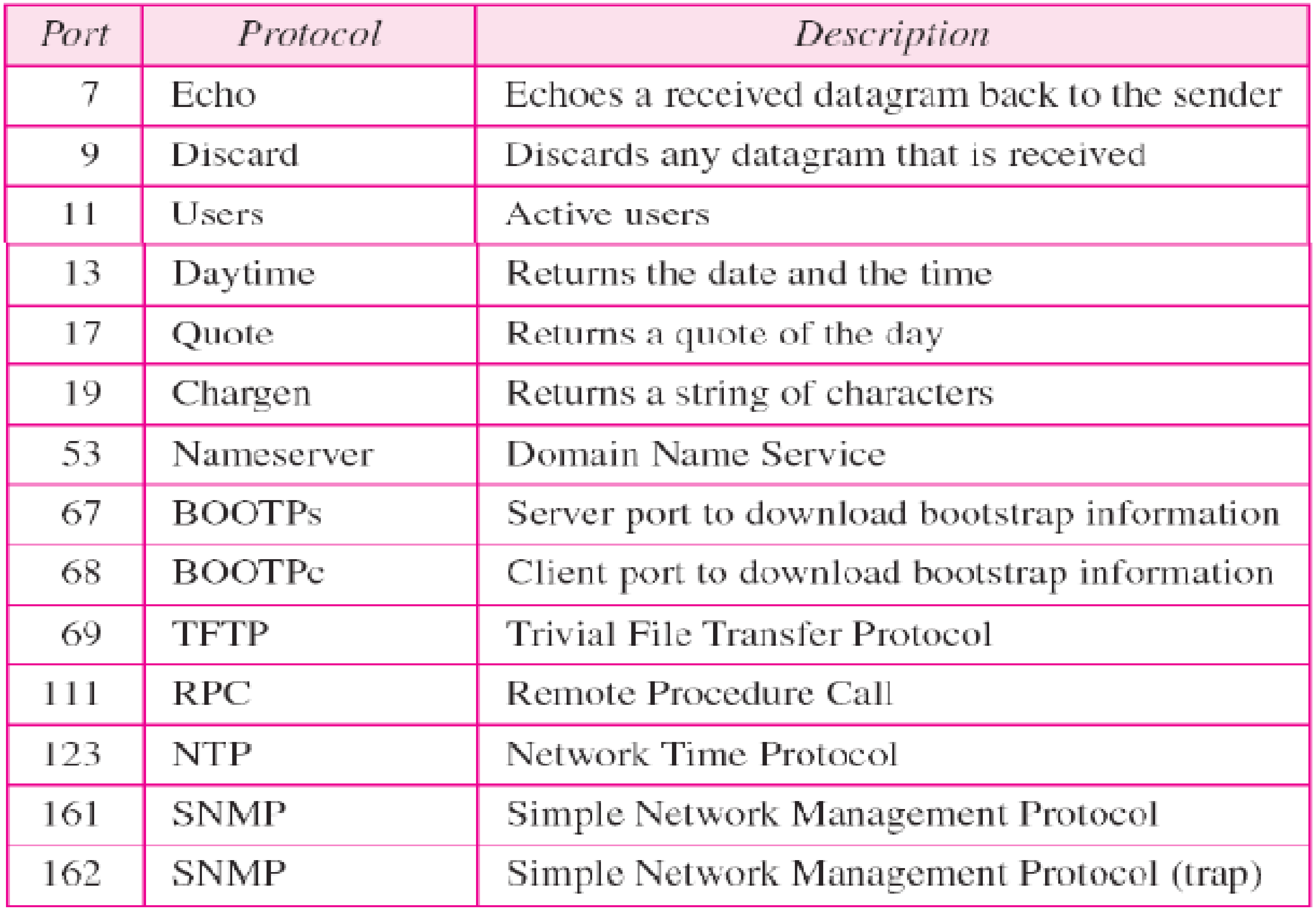
UDP, TCP, SCTP in TCP/IP Protocol Suite



User Datagram Protocol

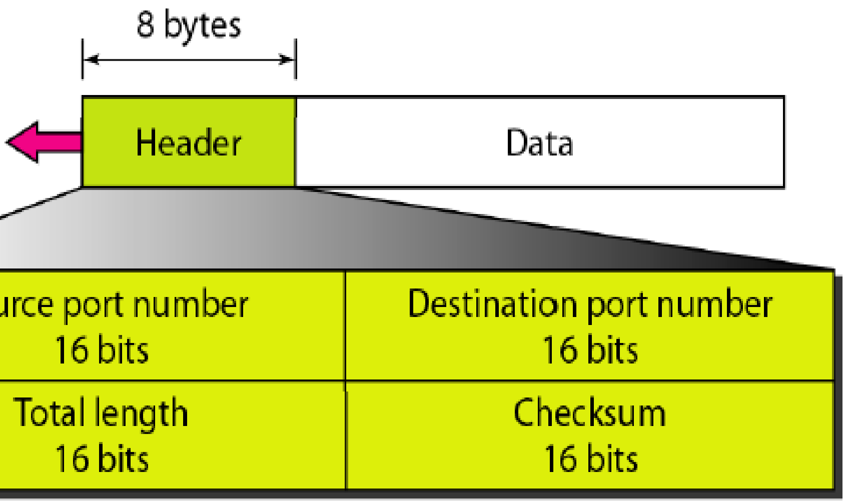
- Connectionless, unreliable transport protocol
- Does not add anything to IP except provide process-to-process communication
- Simple protocol using minimum overhead
- RFC 768

User Datagram Protocol (2)



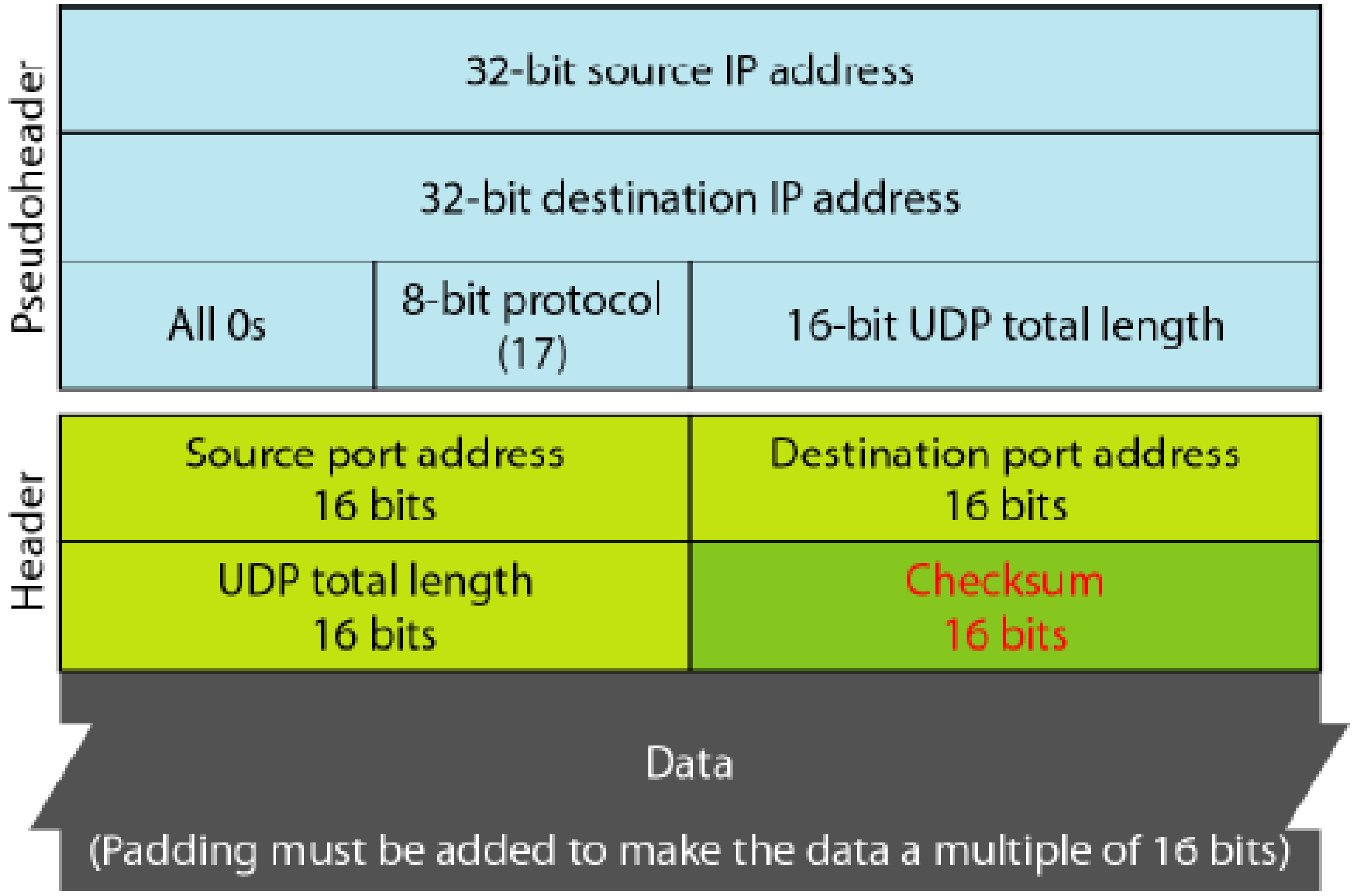
Port	*Protocol*	*Description*
7	Echo	Echoes a received datagram back to the sender
9	Discard	Discards any datagram that is received
11	Users	Active users
13	Daytime	Returns the date and the time
17	Quote	Returns a quote of the day
19	Chargen	Returns a string of characters
53	Nameserver	Domain Name Service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

UDP Format



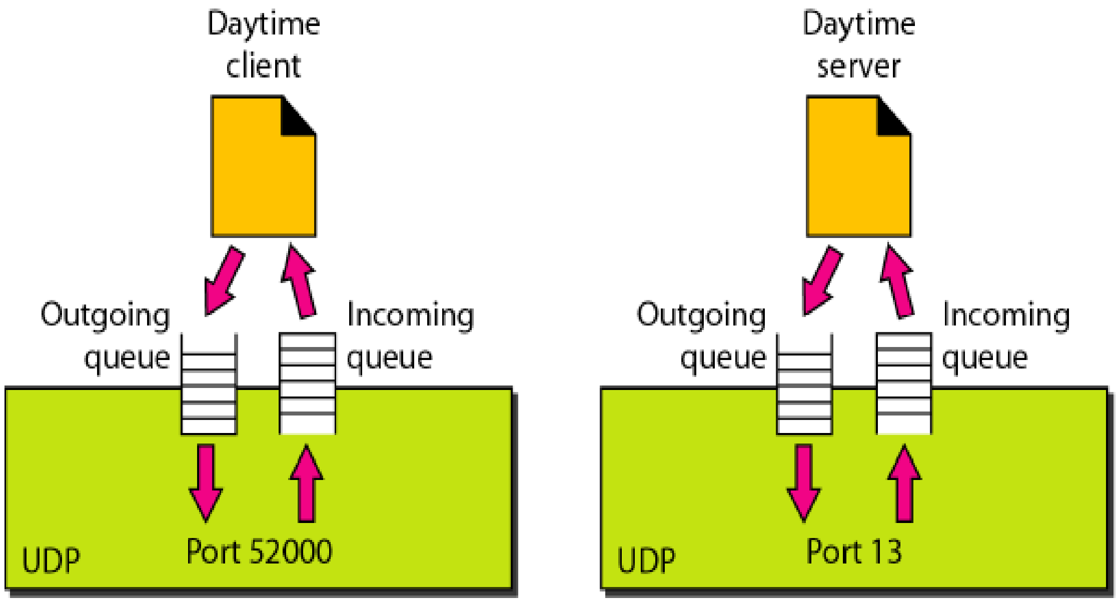
- **Source port number** – 16-bit long number used by the process running on the source host
- **Destination port number** - 16-bit long number used by the process running on the destination host
- **Length** – 16-bit field that defines the total length of the user datagram, header plus data
- **Checksum** – used to detect errors over the entire user datagram

Pseudoheader for checksum calculation


Pseudoheader
32-bit source IP address
32-bit destination IP address
All 0s
8-bit protocol (17)
16-bit UDP total length
Header
Source port address 16 bits
Destination port address 16 bits
UDP total length 16 bits
Checksum 16 bits
Data
(Padding must be added to make the data a multiple of 16 bits)

Queuing

- Some implementation create both an incoming and outgoing queue associated with each process; some only create an incoming queue
- If no queue, UDP discards the user datagram and ask ICMP to send an port unreachable message to the server

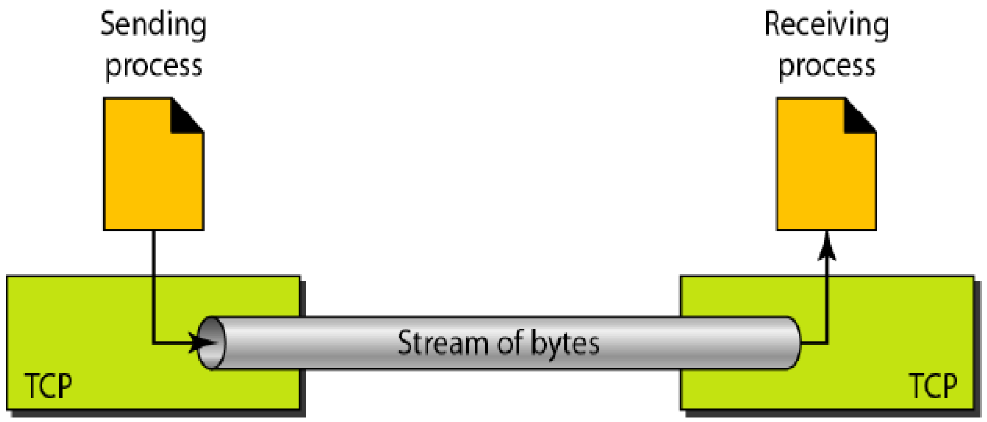


Use of UDP

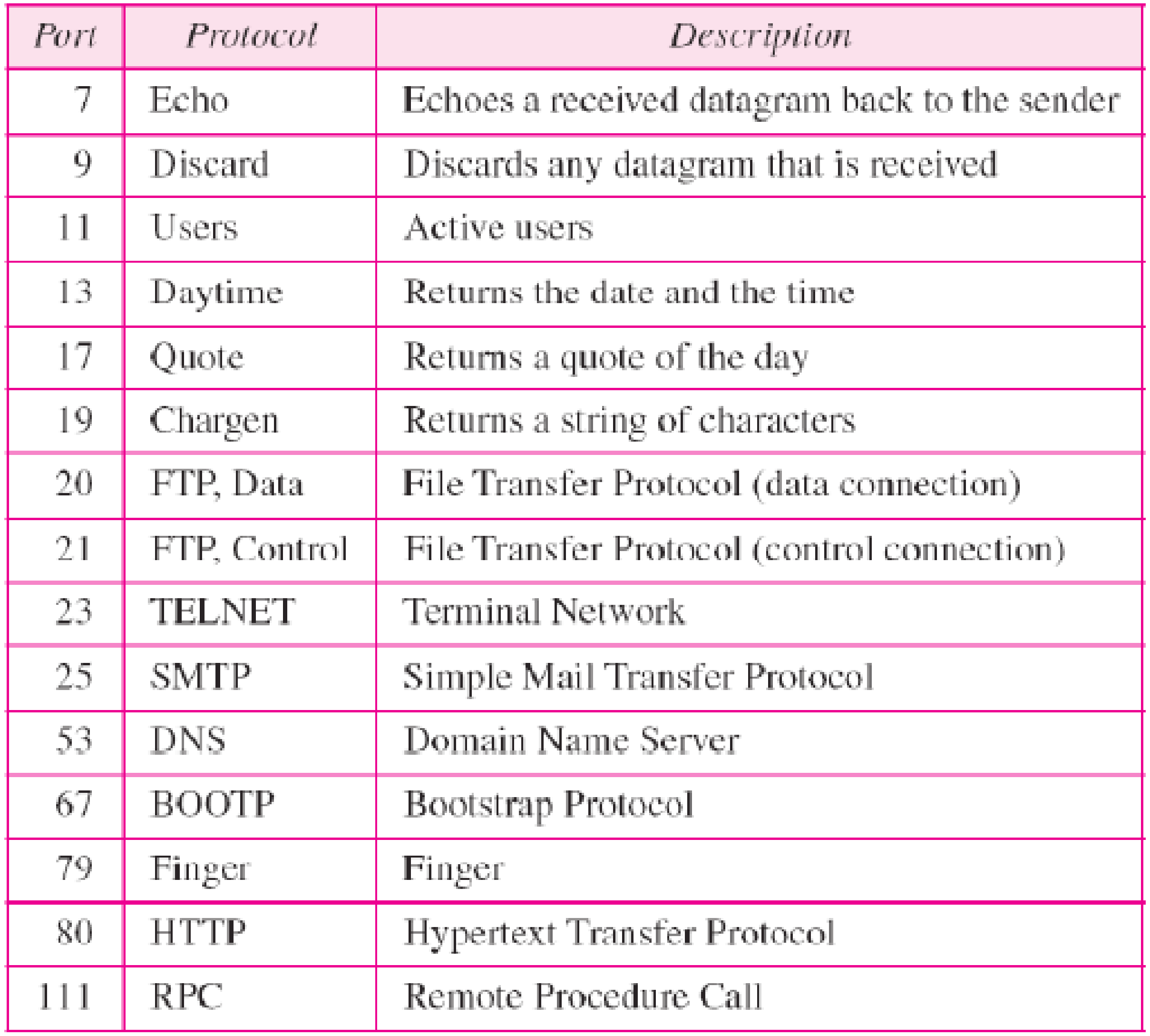
- Suitable for a process that requires simple request-response communication
- Suitable for a process with internal flow and control mechanism
- Suitable for multicasting
- Used for management processes such as SNMP
- Used for some routing protocols such as RIP

Transmission Control Protocol

- Connection-oriented, reliable transport protocol
- Unlike UDP, it allows the sending process to deliver data as a stream of bytes and allow the receiving process to obtain data as stream of bytes
- RFC 793

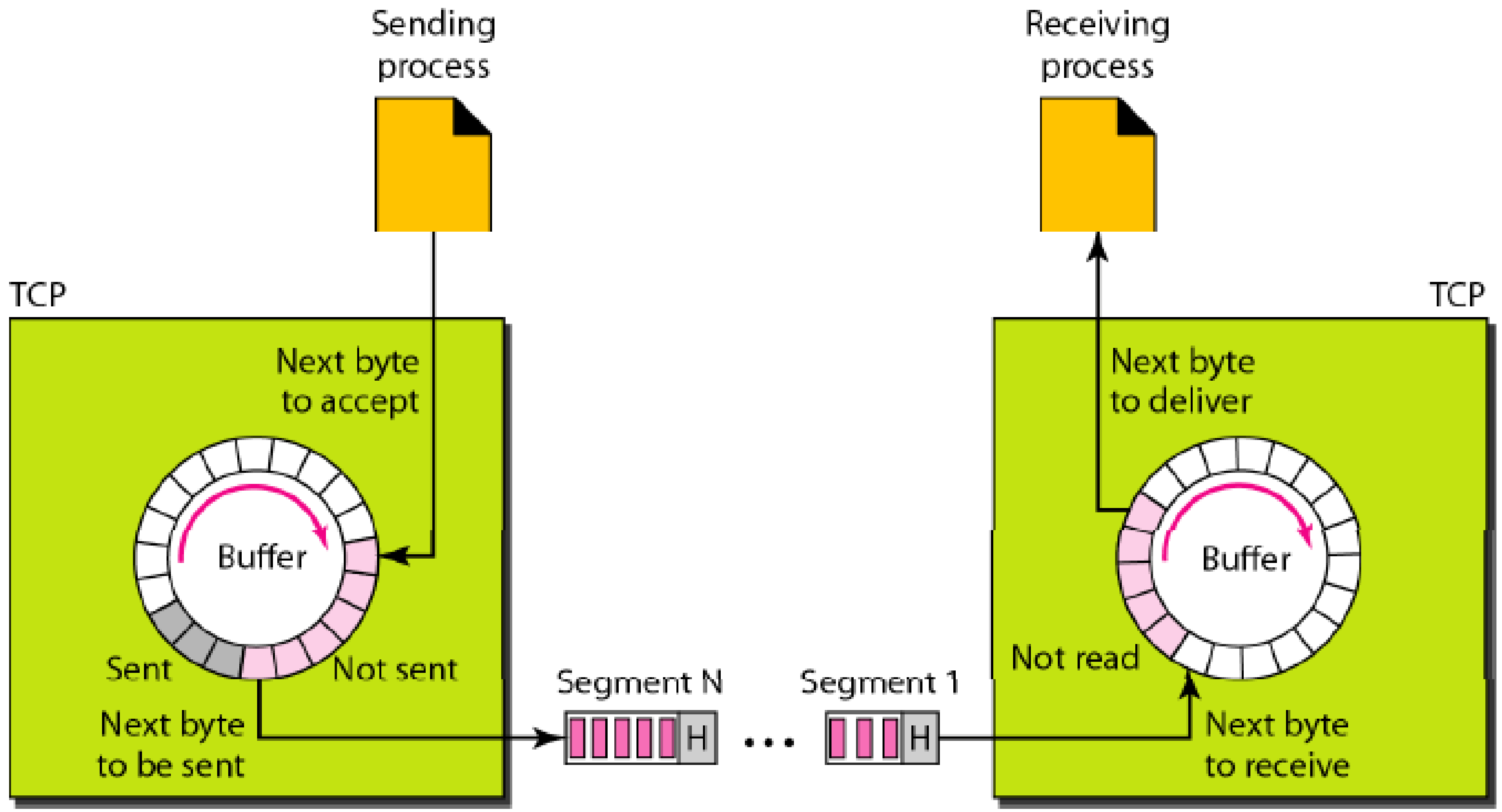


Transmission Control Protocol (2)



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

Buffer



Sending process
Receiving process
TCP
TCP
Next byte to accept
Next byte to deliver
Buffer
Buffer
Sent
Not sent
Not read
Next byte to be sent
Next byte to receive
Segment N
Segment 1
H
H

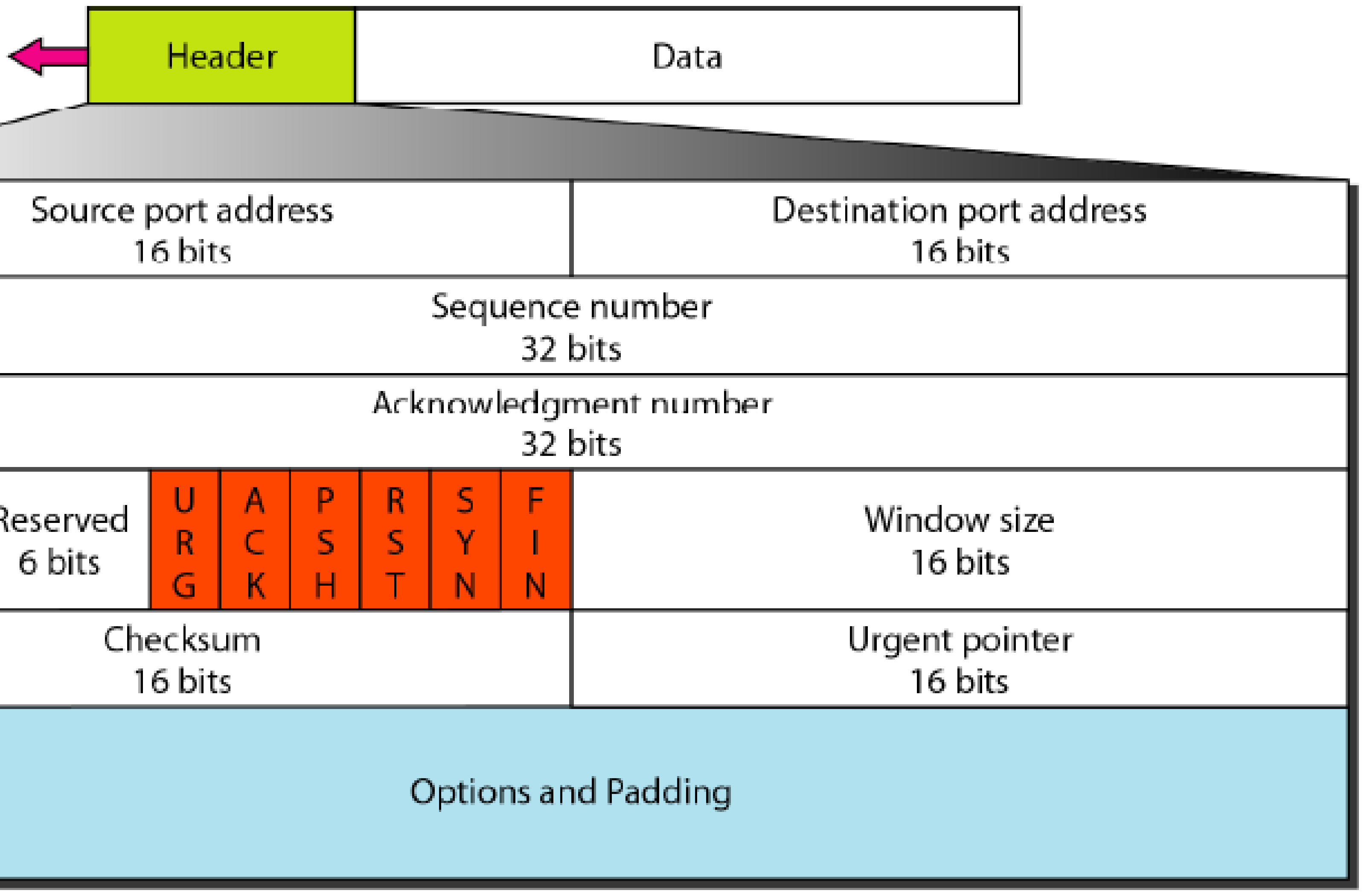
TCP Features

- **Numbering System**
  - Byte number – TCP numbers all data bytes that are transmitted in a connection. Numbering is independent in each direction and starts with a randomly generated number between 0 and 2^{32-1}
  - TCP assigns a **sequence number** to each segment that is being sent. The sequence number for each segment is the number of the first byte carried in that segment
  - When a connection is established, both parties can send and receive data at the same time. Each party uses an **acknowledge number** to confirm the bytes it has received

TCP Features (2)

- **Flow control** – receiver of data controls the amount of data that are to be sent by the sender
- **Error control** – to provide reliable service, TCP implements an error control mechanism
- **Congestion control** – amount of data sent by the sender is not only controlled by the receiver, but is also determined by the level of congestion in the network

TCP Segment Format

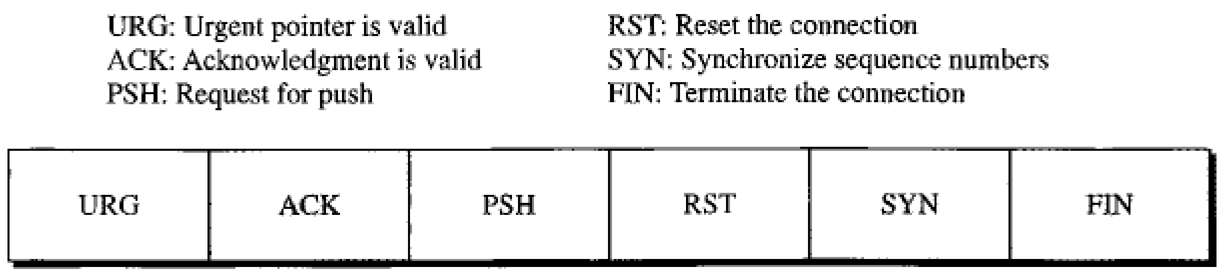


TCP Segment Format (2)

- **Source port address** – 16-bit field that defines the port number of the application program in the host that is sending the segment
- **Destination port address** – 16-bit field that defines the port number of the application program in the host that is receiving the segment
- **Sequence number** – 32-bit field that defines the number assigned to the first byte of data contained in the segment. During connection establishment, each party uses a random number generator to create an **initial sequence number** (ISN)

TCP Segment Format (3)

- **Acknowledgment number** – 32-bit field that defines the byte number that the receiver of the segment is expecting to receive from the other party
- **Header length** – 4-bit field indicates the number of the 4-byte word in the TCP header
- **Reserved** – 6-bit field for future use
- **Control** – 6 different control bits or flags


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

TCP Segment Format (4)

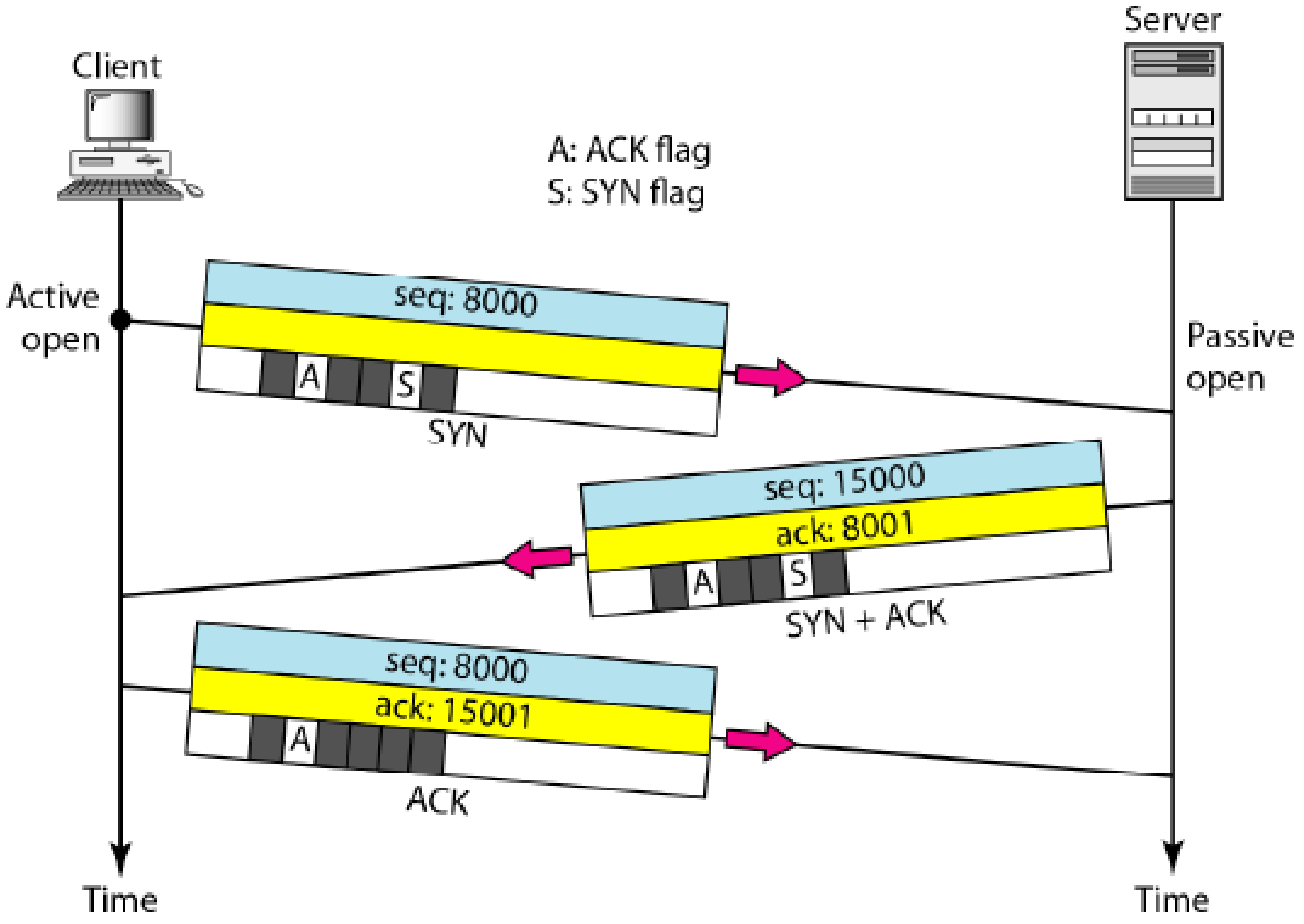
- **Window size** – 16-bit field that defines the size of the window, in bytes, that the other party must maintain. Normally referred to as the receiving window (rwnd) and has max value of 65,535 bytes
- **Checksum** – 16-bit field where the calculation is same as UDP. Inclusion of the checksum in TCP is mandatory while it is optional in UDP
- **Urgent pointer** – 16-bit field that defines 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** – Can be up to 40-bytes of optional information

TCP Connection

- TCP establishes a virtual path between the source and destination
- TCP uses the services of IP to deliver individual segments to the receiver, but it controls the connection itself
- It requires 3 phases: connection establishment, data transfer, and connection termination

Connection Establishment

Three-way Handshaking



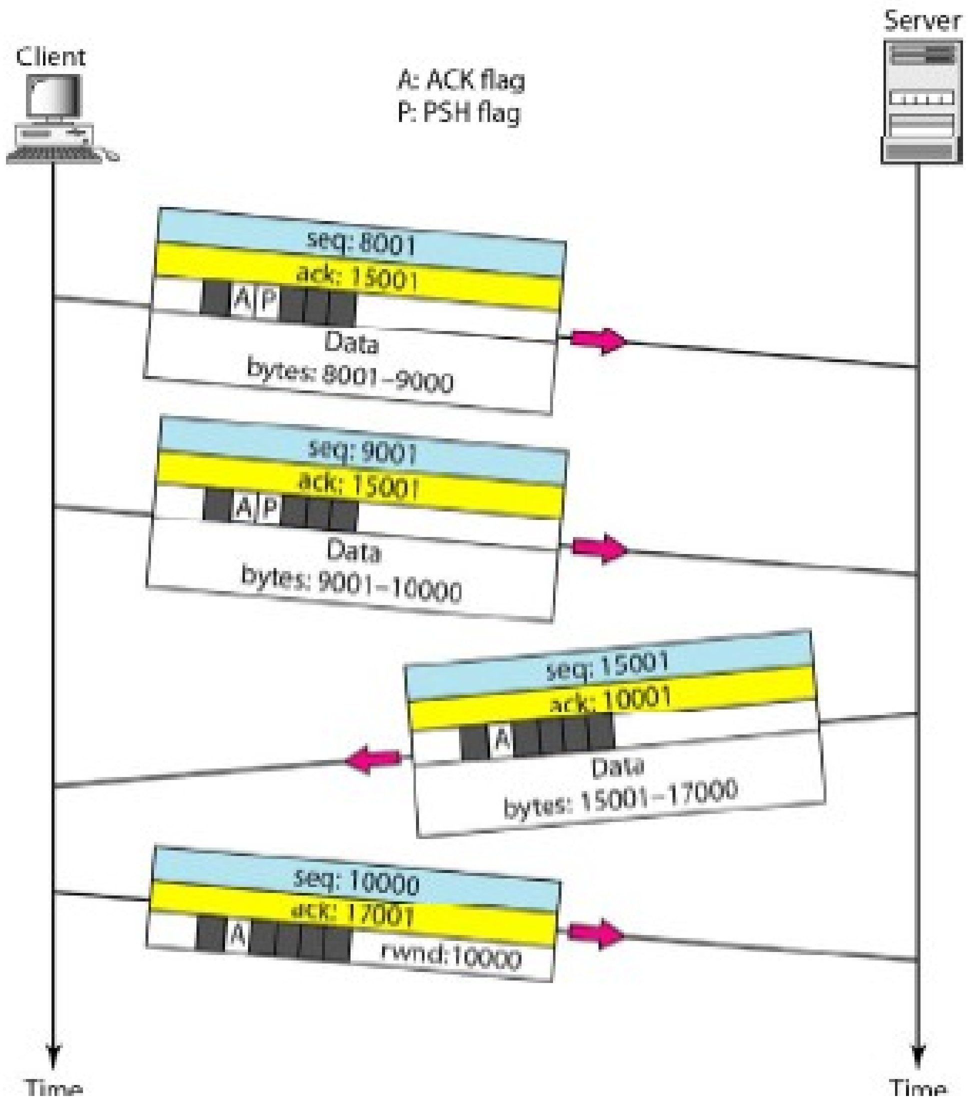
Connection Establishment (2)

- Process starts with the server where it tells its TCP that it is ready to start a connection (**passive open**)
- A client that wishes to connect to an open server tells it TCP that it needs to be connected to that particular server (**active open**)
- A rare situation (**simultaneous open**) may occur when both process issue an active open. Both TCP transmit a SYN + ACK segment to each other, and one single connection is established between them

Connection Establishment (3)

- Susceptible to the **SYN flooding attack**
  - Malicious attacker sends a large number of SYN segments to a server, pretending that each of them is coming from different clients by faking the source IP address in the datagrams
  - The server, assuming that the clients are issuing an active open, allocates the necessary resources and then sends the SYN + ACK segments to the fake clients which are lost

Data Transfer

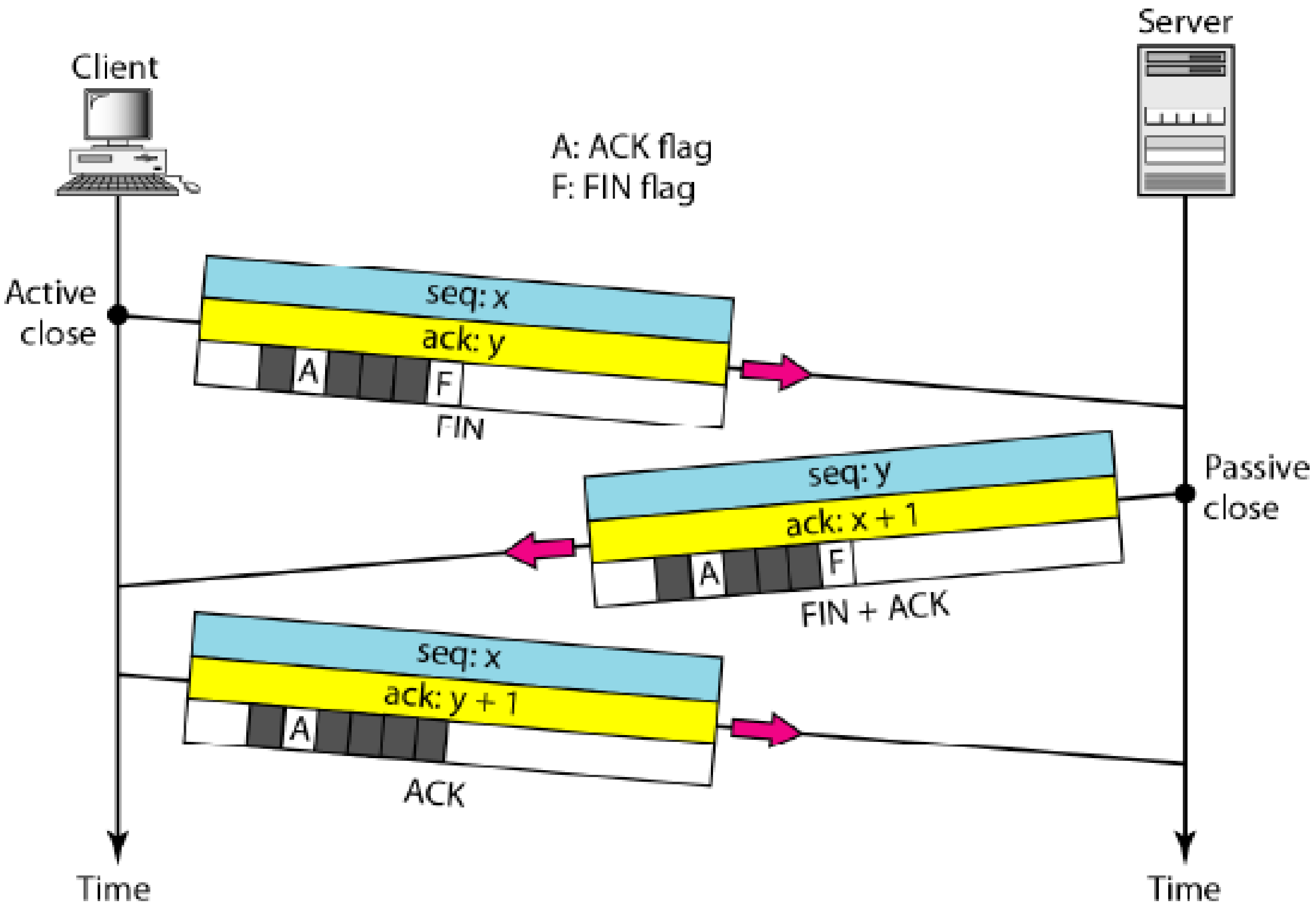


Data Transfer (2)

- After connection is established, bidirectional data transfer can take place. The client and server can both send data and acknowledgments
- **Pushing Data** – The sending TCP must not wait for the window to be filled, it must create a segment and send it immediately
- **Urgent Data –** the sending application program wants a piece of data to be read out of order by the receiving application program

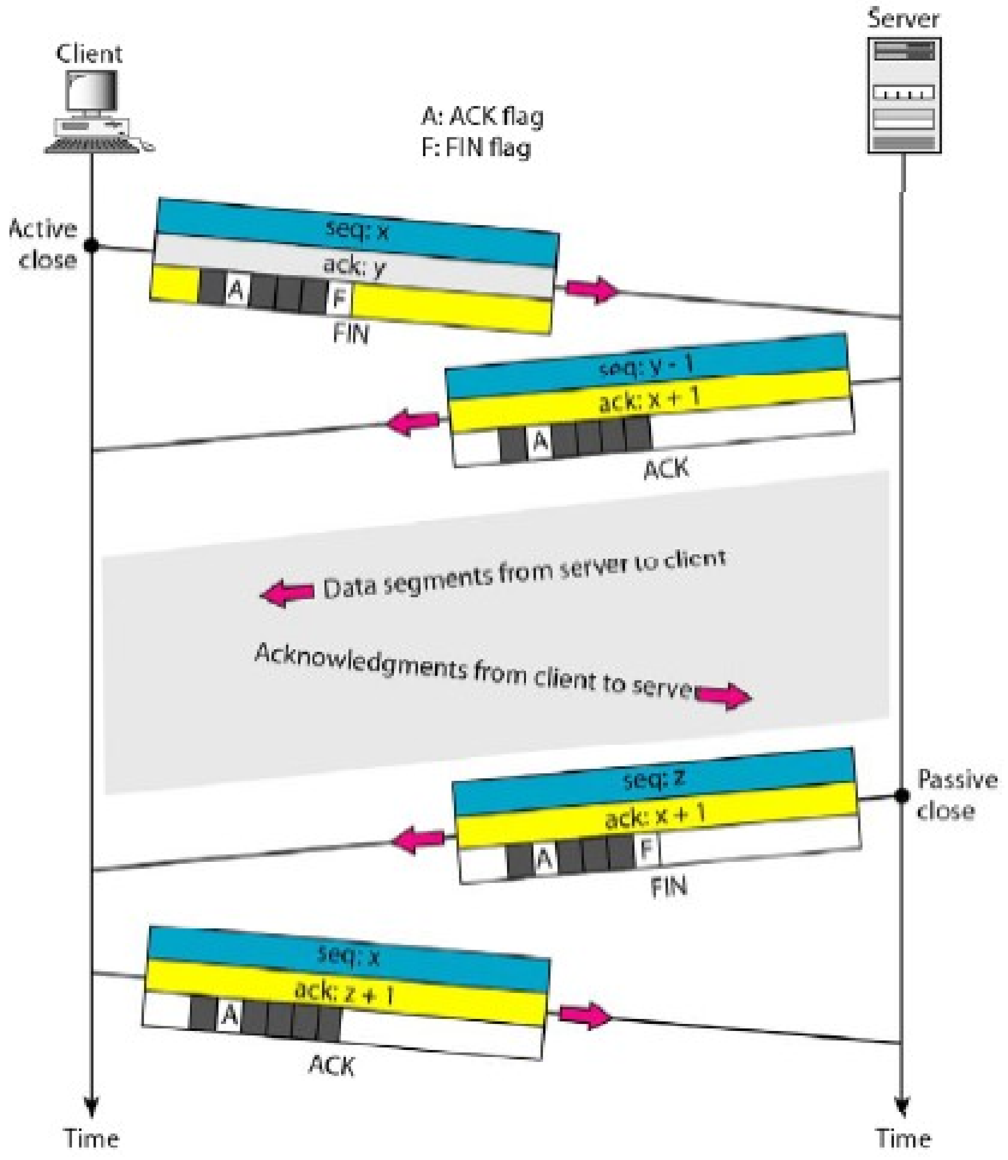
Connection Termination

Three-way Handshaking



Connection Termination (2)

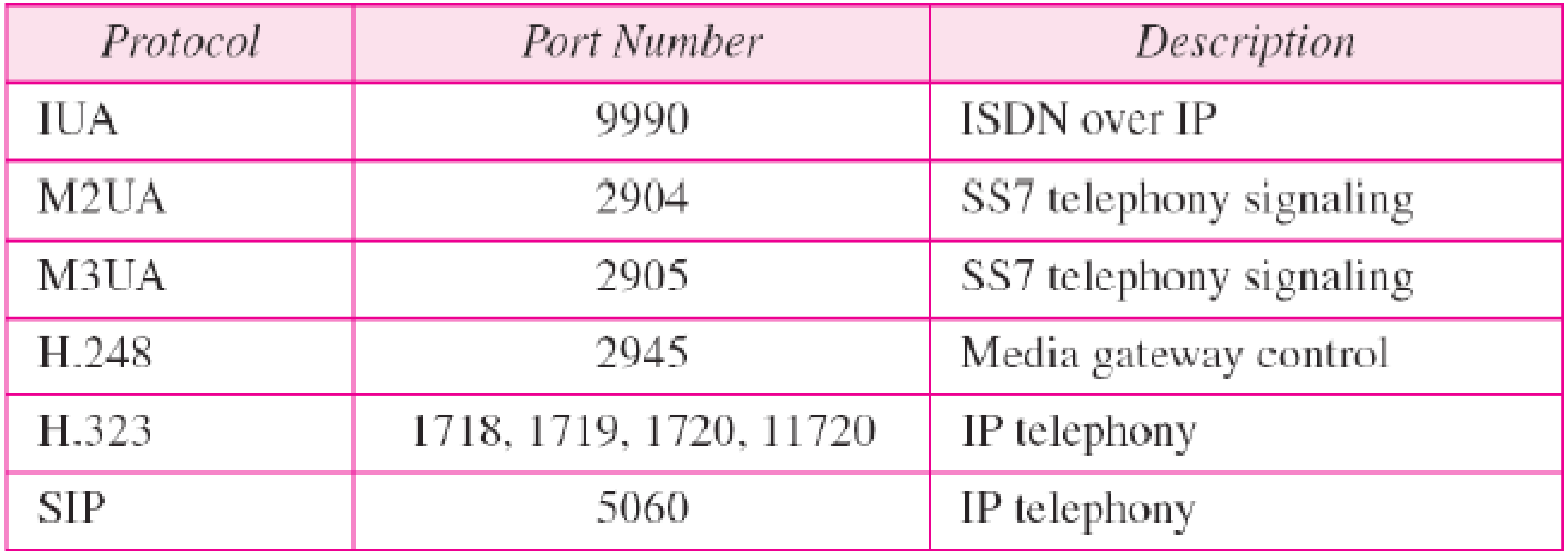
Four-way handshaking with half-close option



Stream Control Transmission Protocol

- New reliable, message-oriented transport layer protocol
- Preserves the message boundaries and at the same time detects lost data, duplicate data and out-of-order data
- It has also congestion control and flow control mechanisms

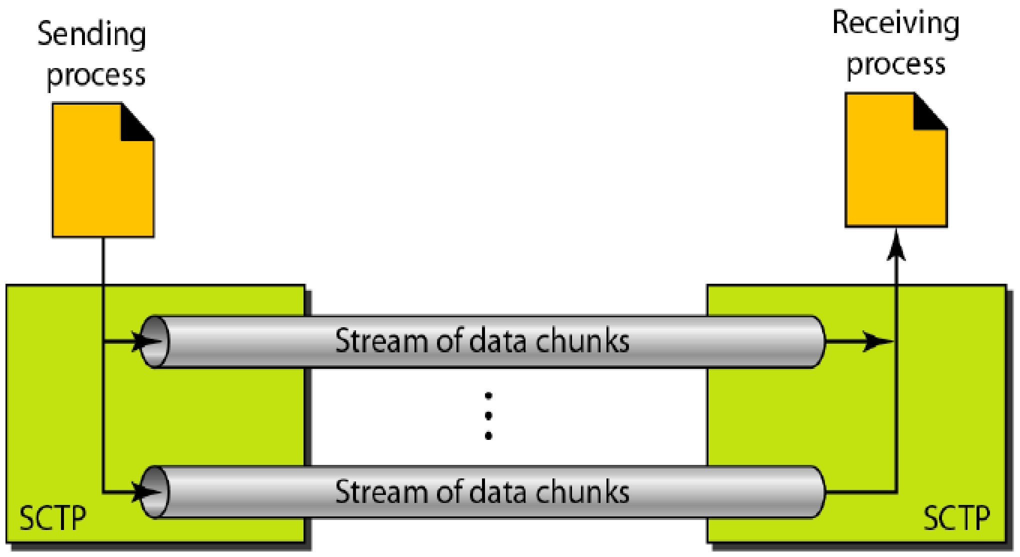
Stream Control Transmission Protocol (2)



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

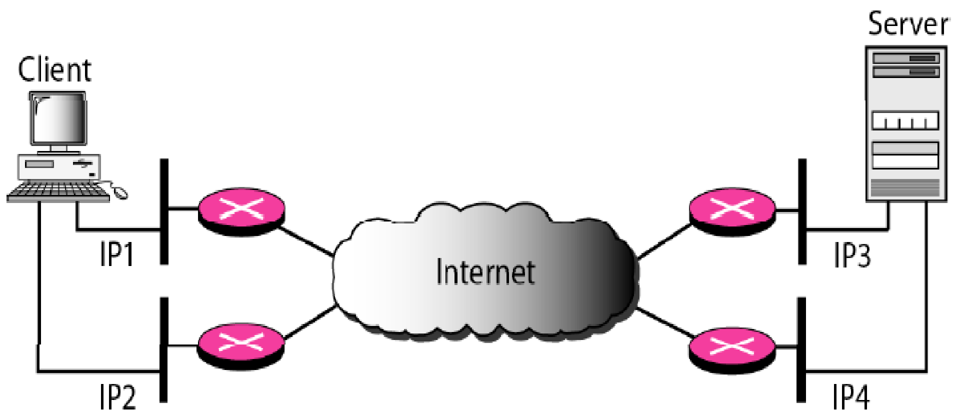
SCTP Services

- **Multiple streams**
  - *Association* in SCTP terminology



SCTP Services (2)

- **Multihoming**
  - Sending and receiving host can define multiple IP addresses in each assocation
  - When one path fails another interface can be used for data delivery without interruption



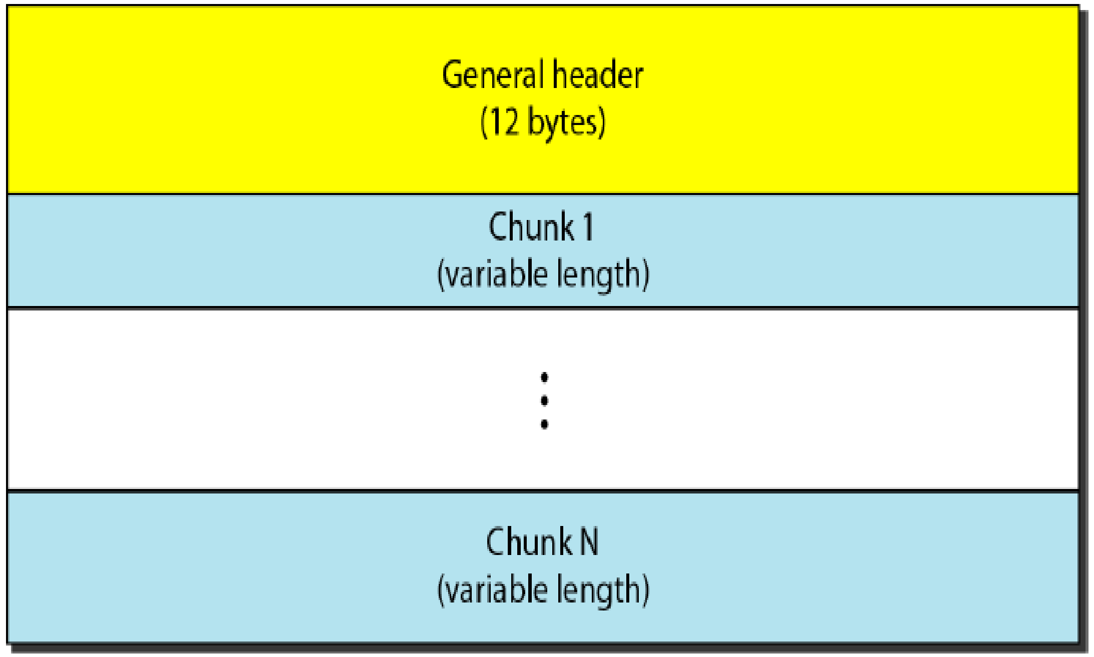
SCTP Feature

- **Transmission Sequence Number** (TSN) – unit of data in SCTP is a **data chunk**. Data transfer is controlled by numbering the data chunk using TSN
- **Stream Identifier** (SI) – each stream in SCTP needs to be identified
- **Stream Sequence Number** (SSN) – SCTP defines each data chunk in each stream

SCTP Feature (2)

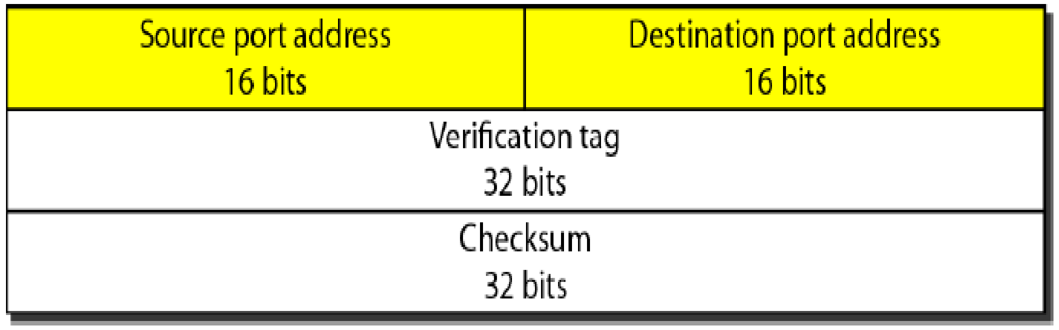
- **Packets** – data are carried as data chunks, control information is carried as control chunks. Several control chunks and data chunks can be packed together in a packet

SCTP Packet Format



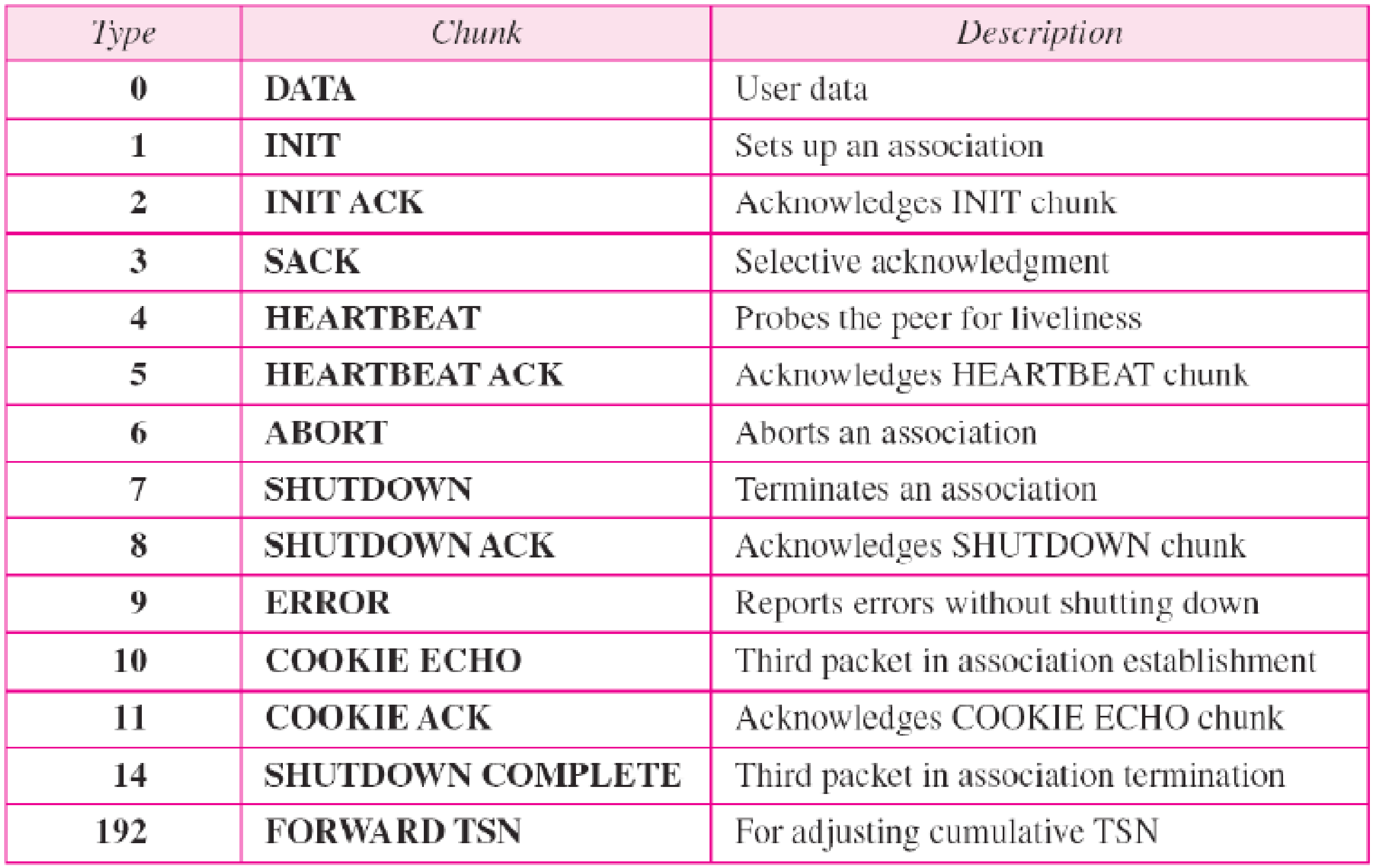
- In an SCTP, control chunks come before data chunks

SCTP General Header



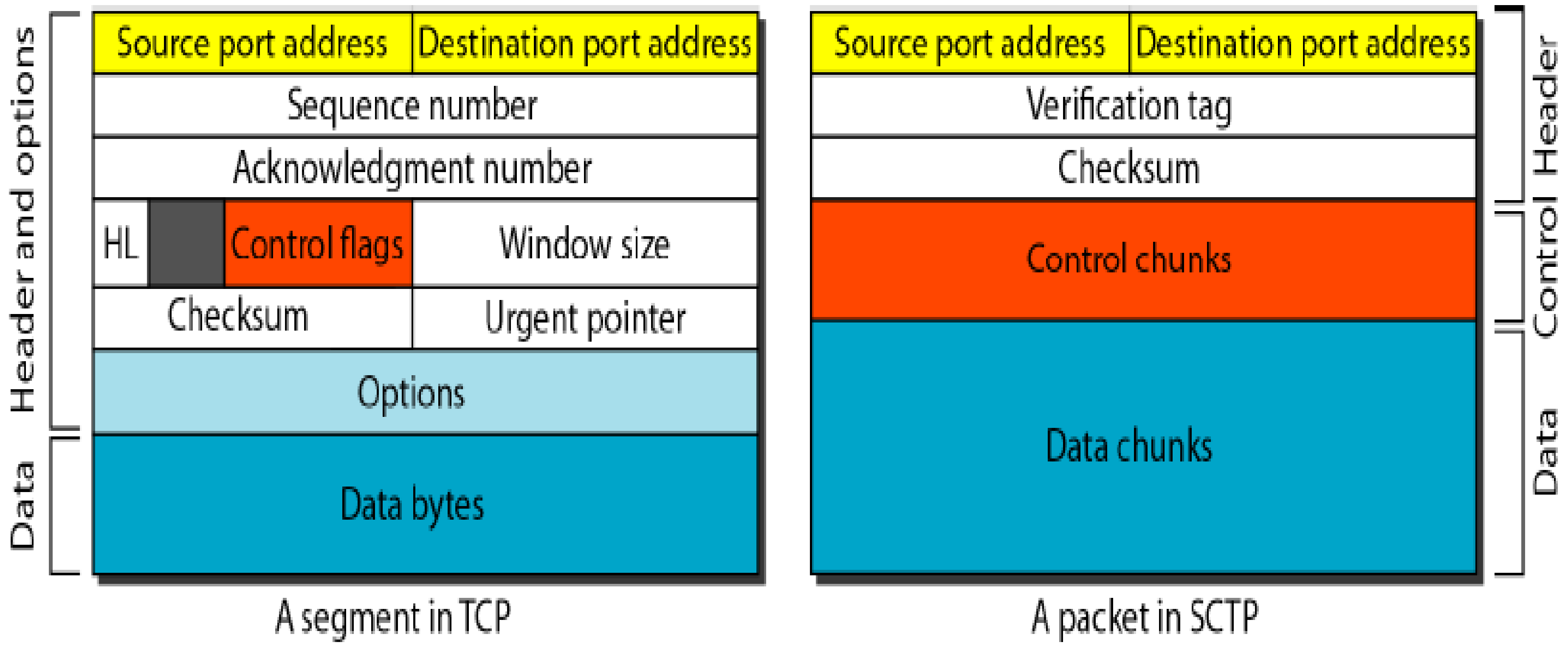
- **Source Port Address** and **Destination Address** – 16-bit field that defines the port numbers of the source and destination respectively
- **Verification tag** – number that matches a packet to an association. It serves as an identifier for the association and is repeated in every packet during the association
- **Checksum** – 32-bit field contains a CRC-32 checksum

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

Comparison between a TCP segment and a SCTP packet


Header and options
Data
Source port address
Destination port address
Sequence number
Acknowledgment number
HL
Control flags
Window size
Checksum
Urgent pointer
Options
Data bytes
A segment in TCP
Source port address
Destination port address
Verification tag
Checksum
Control chunks
Data chunks
A packet in SCTP
Header
Control
Data