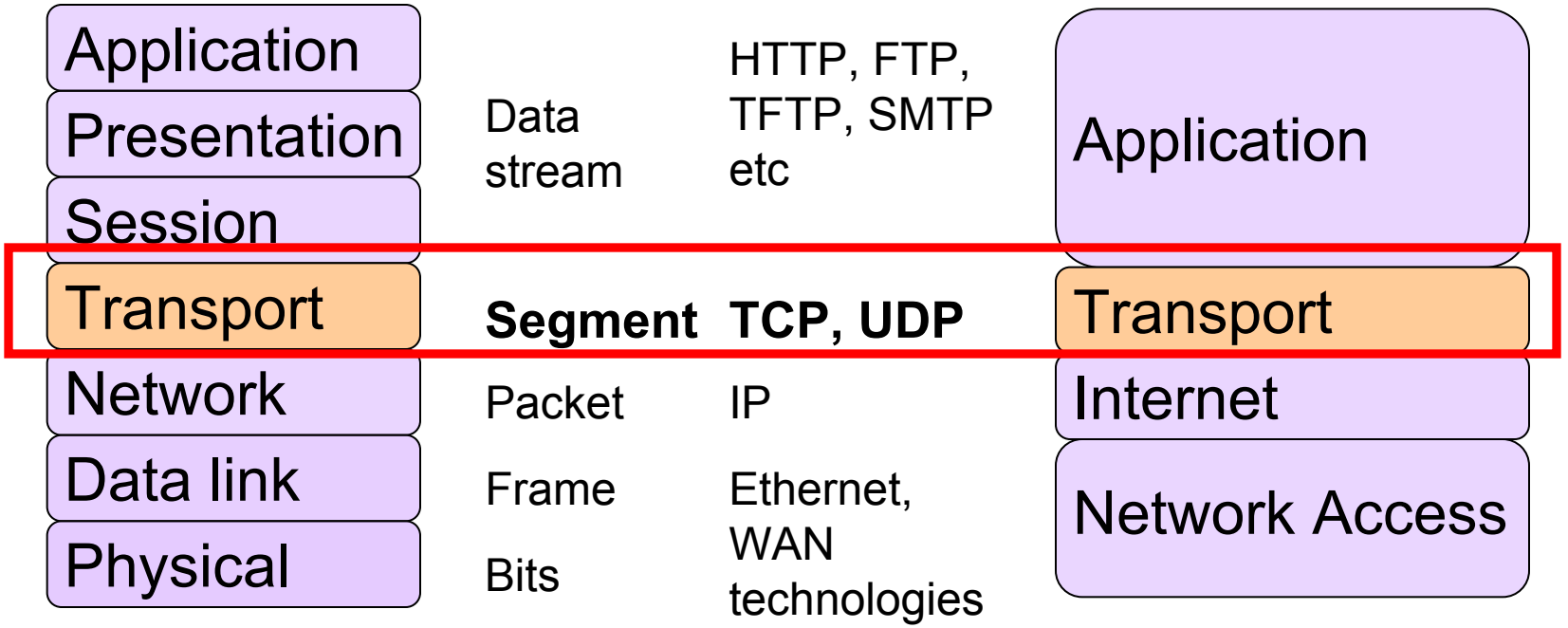
Transport Layer

OSI transport layer

- OSI model layer 4
- TCP/IP model Transport layer



Transport vs. Network Layer

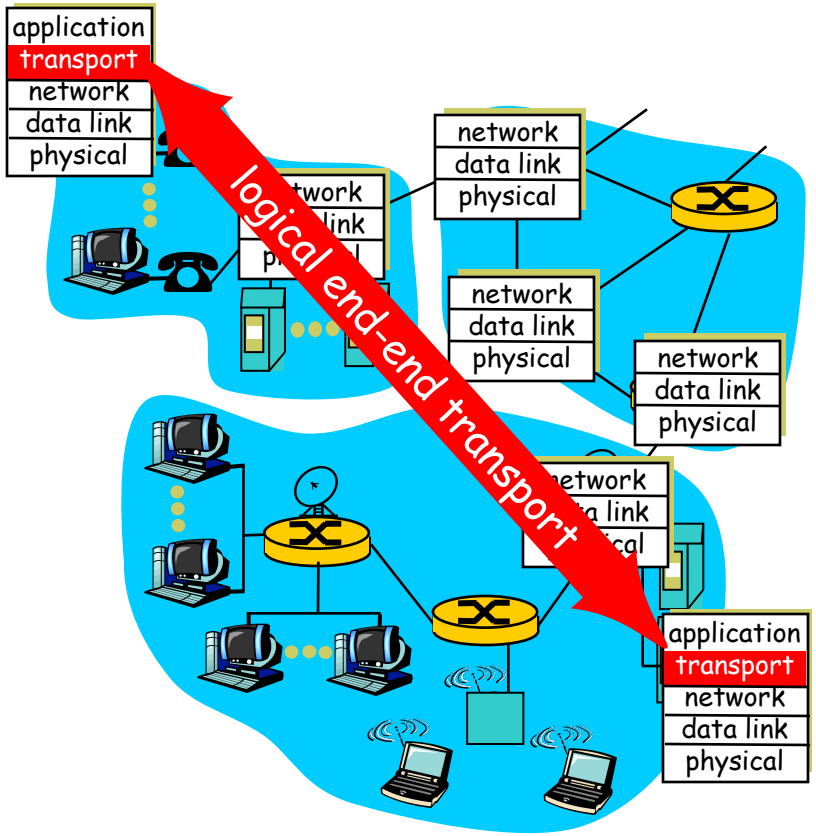
Household analogy:

12 kids in Dhaka sending letters to 12 kids in Khulna.

- processes = kids
- app messages = letters in envelopes
- hosts = houses
- transport protocol = Arif and Babar
- network-layer protocol = postal service

Transport Services and Protocols

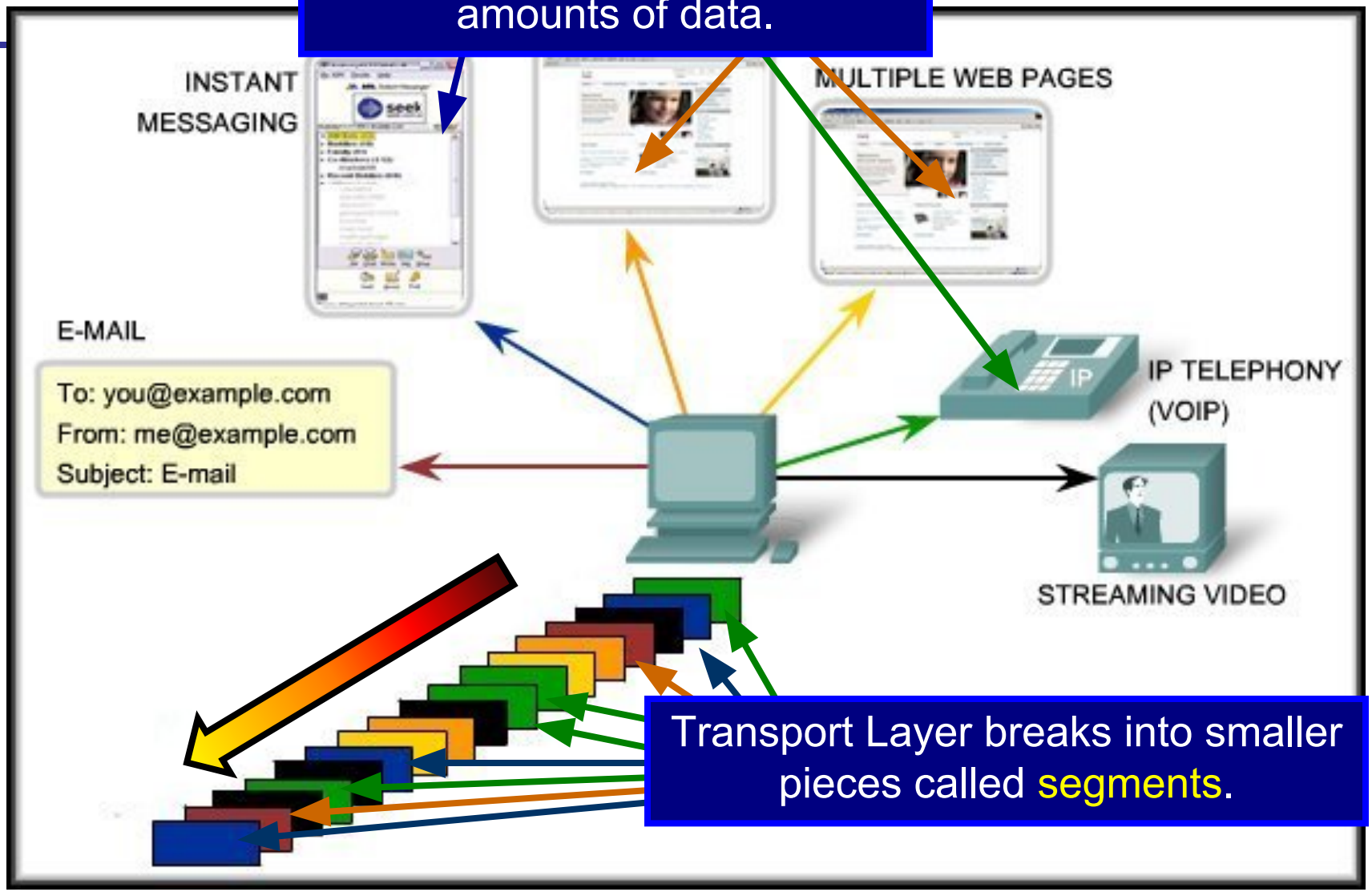
- **network layer***:* logical communication between **hosts.**
- **transport layer**: logical communication between **processes.**



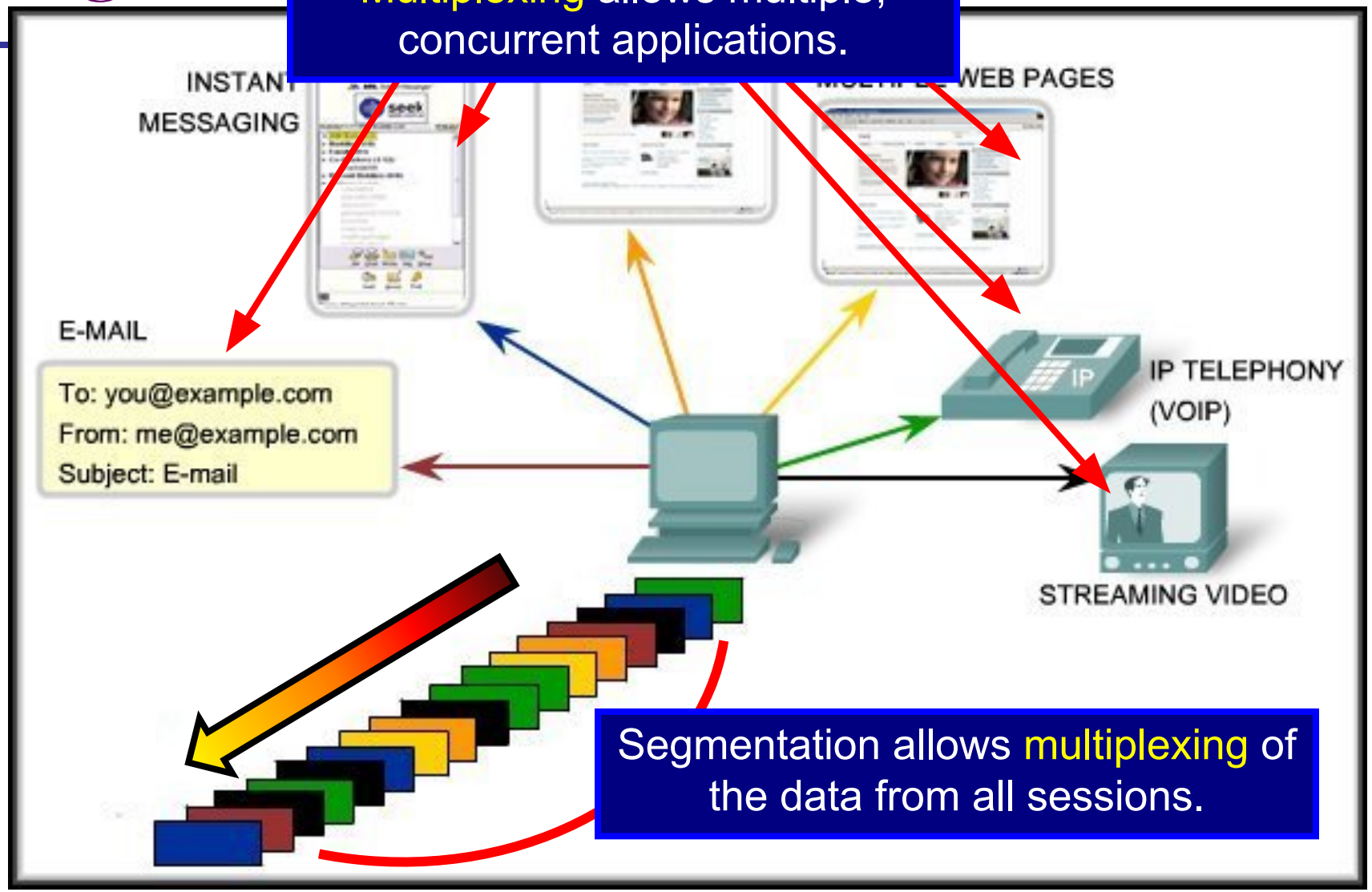
Primary Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

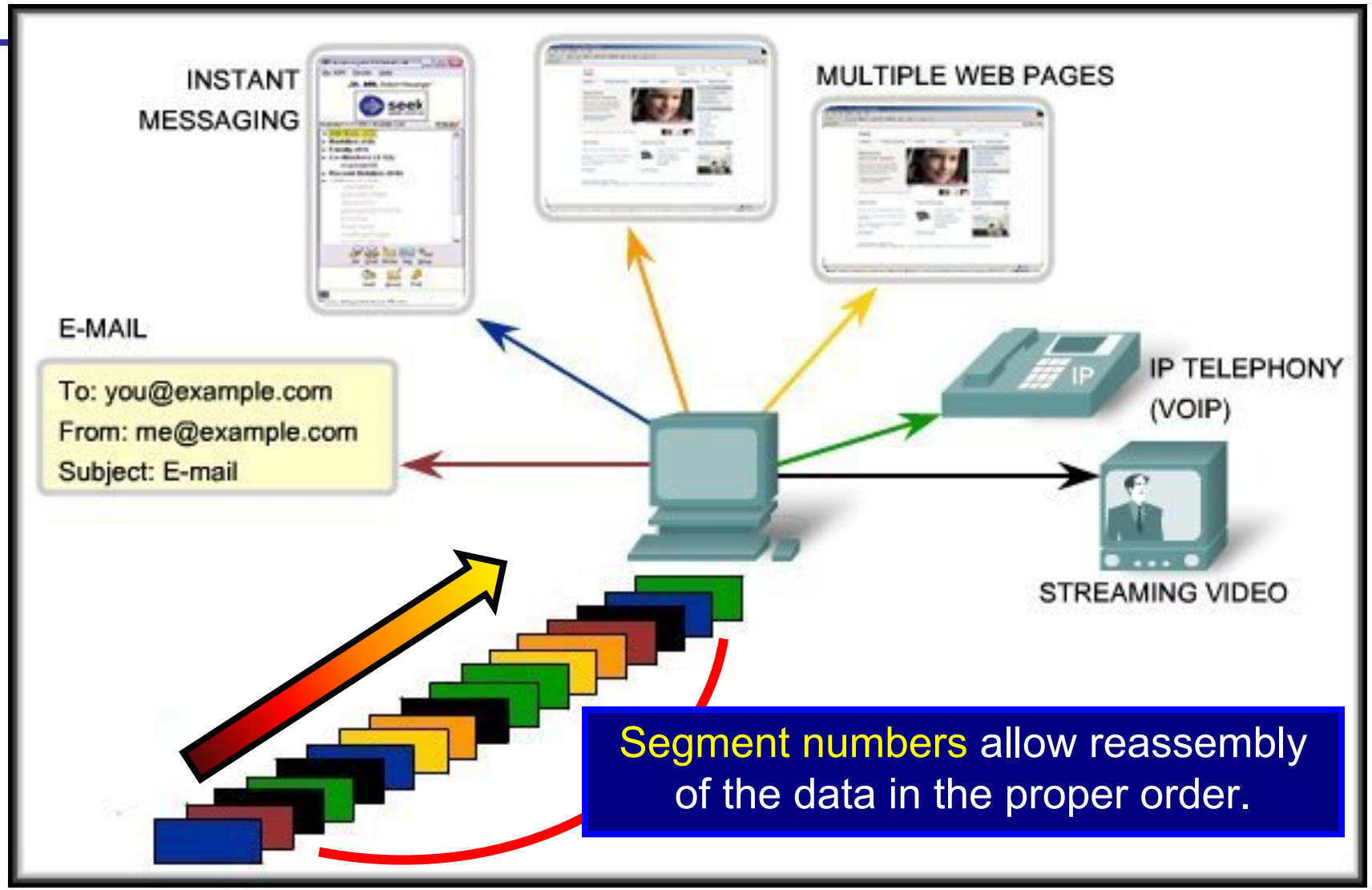
Segmenting Data



Segmenting and Managing Data

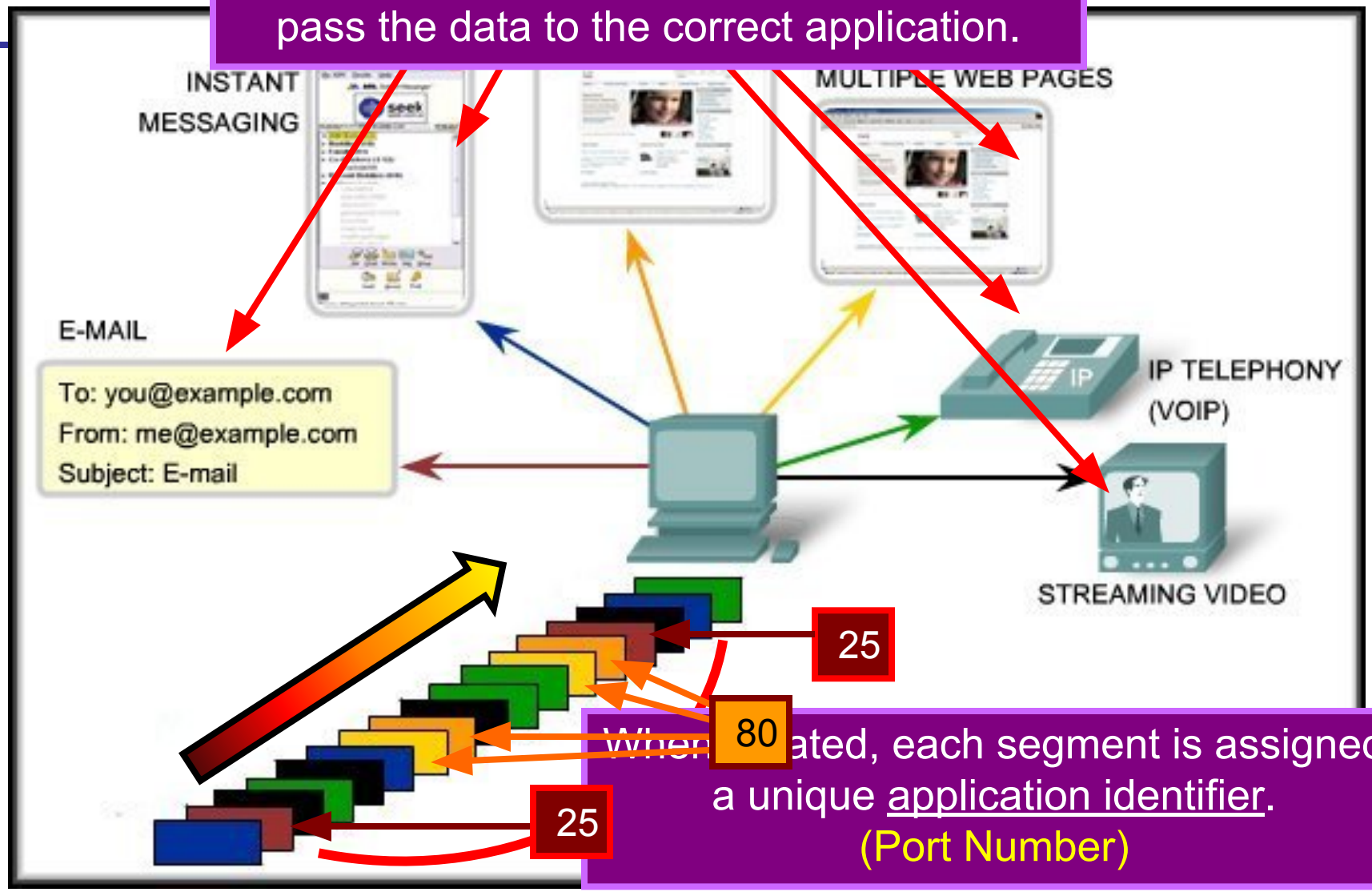


Reassembling Segments



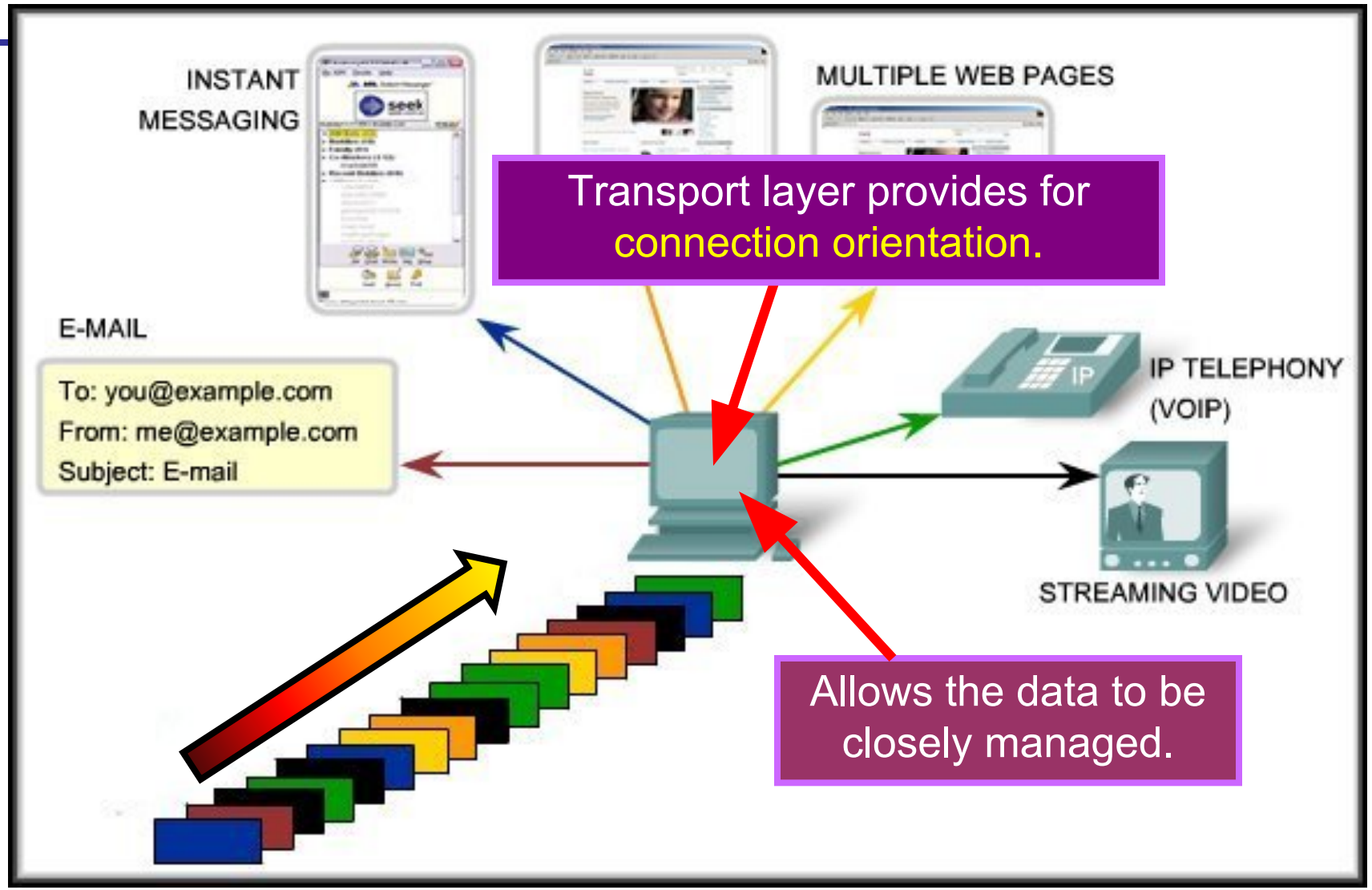
Identifying Applications

After reassembly, the port number is used to pass the data to the correct application.

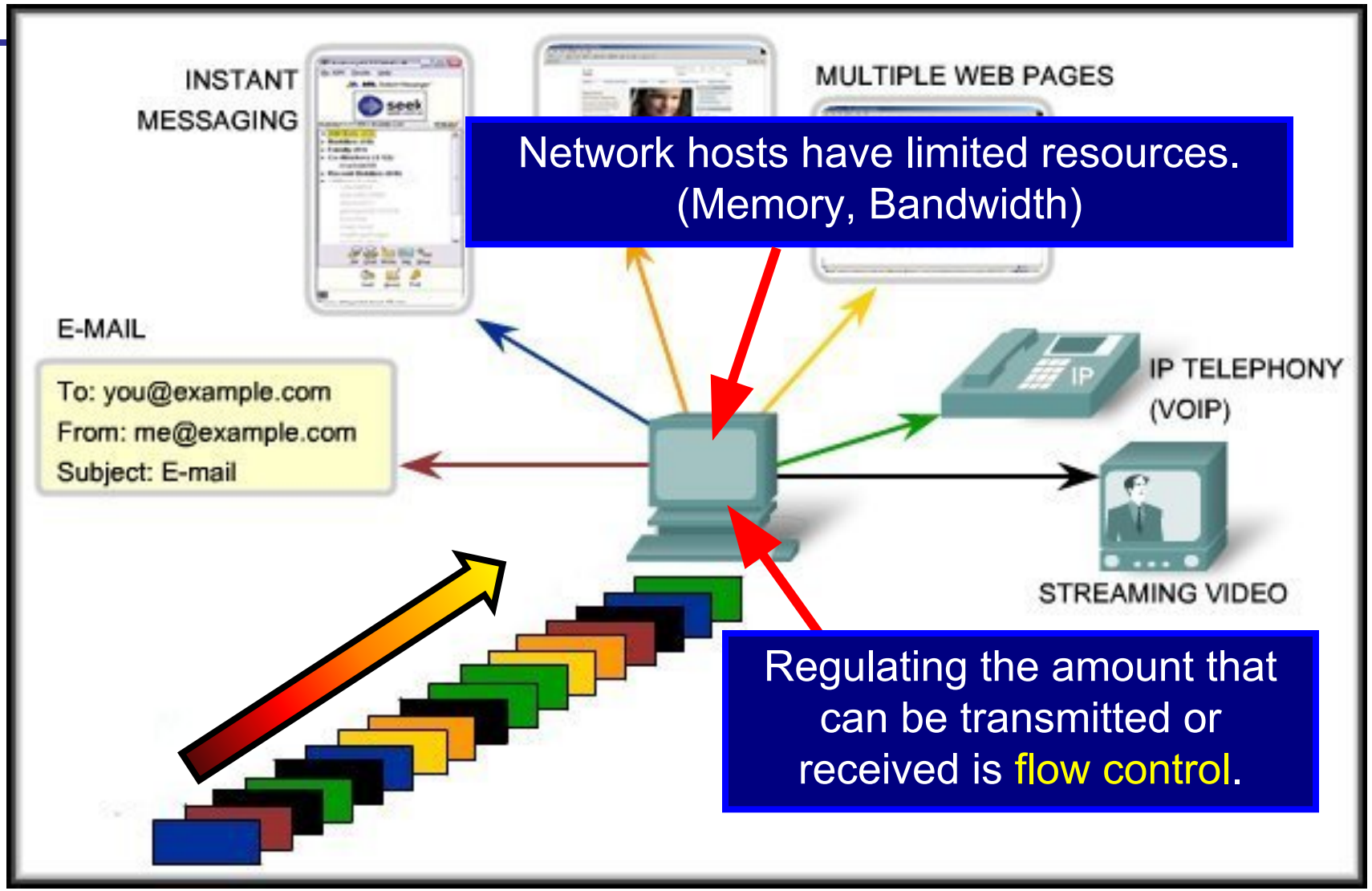


When created, each segment is assigned a unique application identifier. (Port Number)

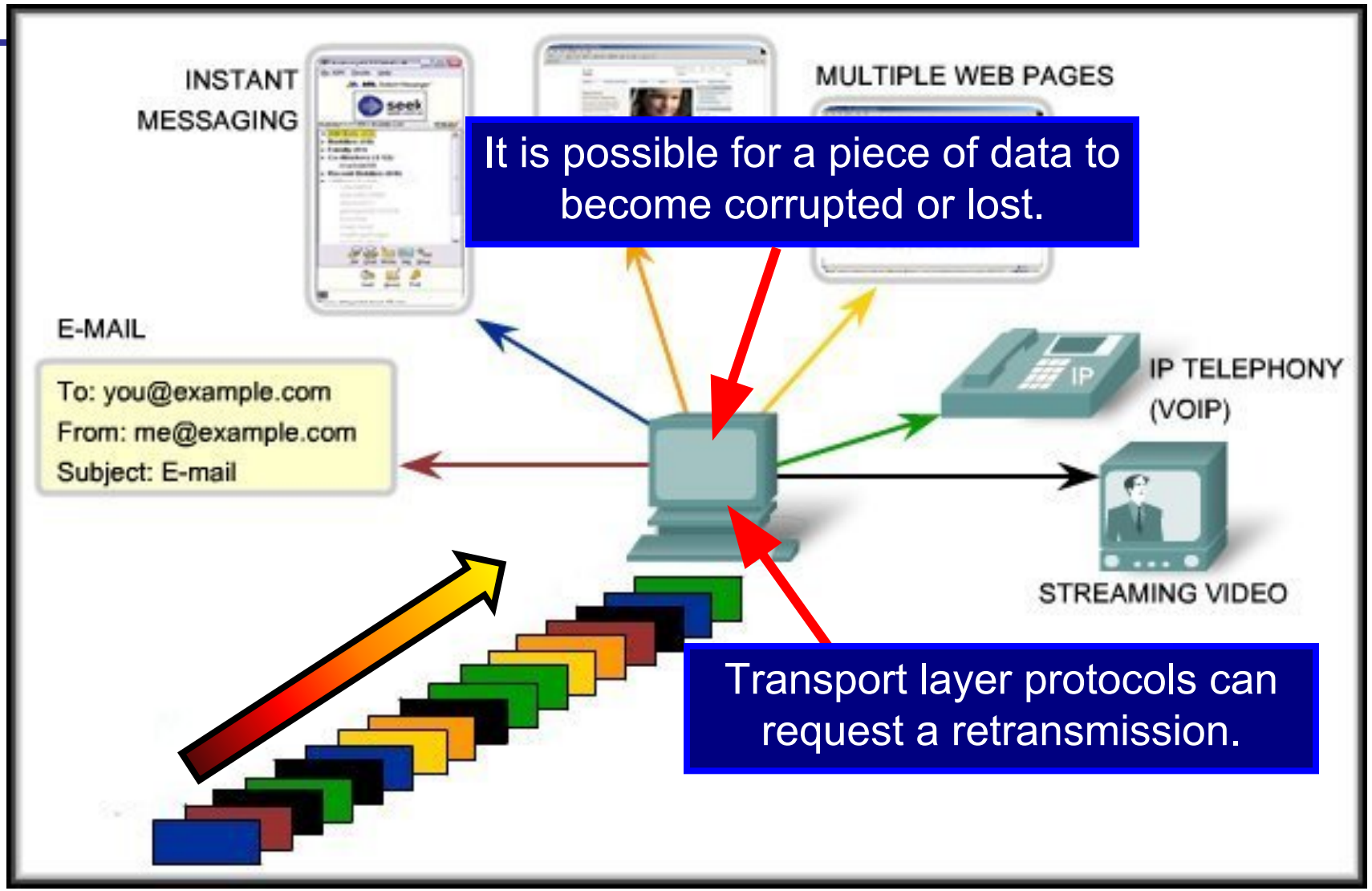
Initiating a Session



Flow Control



Error Control



Purpose of the Transport Layer

- Primary responsibilities:
  - Segmenting the data and reassembling the segments.
  - Multiplexing segments.
  - Identifying and tracking the different applications.
  - Initiating a session between applications at two user ends.
  - Error Contro st and corrupted packets.
  - Flow control – Controlling the data flow between sender and receiver.

Reliability

Reliability

- To support these reliability operations, more control data is added in the Transport Layer header.
- More extra functions are needed such as connection establishment etc.
- These extra information and functions causes delay but ensures reliability.

Different Applications Different Requirements

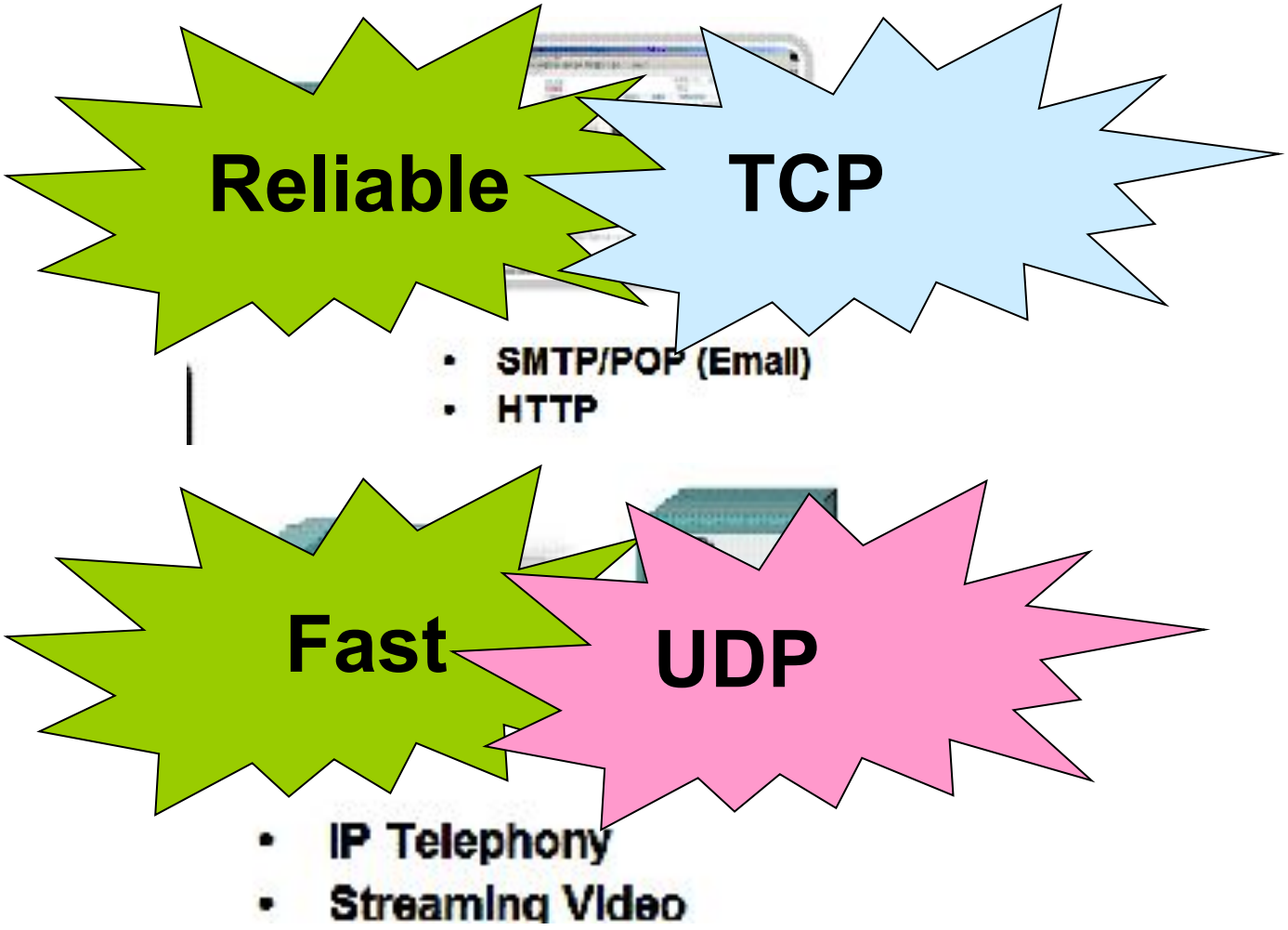
- SMTP/POP (Email)
- HTTP

- IP Telephony
- Streaming Video

- Data needs to be complete with no errors or gaps
- Slight delay is acceptable to ensure this.

- Accept occasional errors or gaps in the data.
- But delay is not acceptable.

Solution : Two transport protocols?

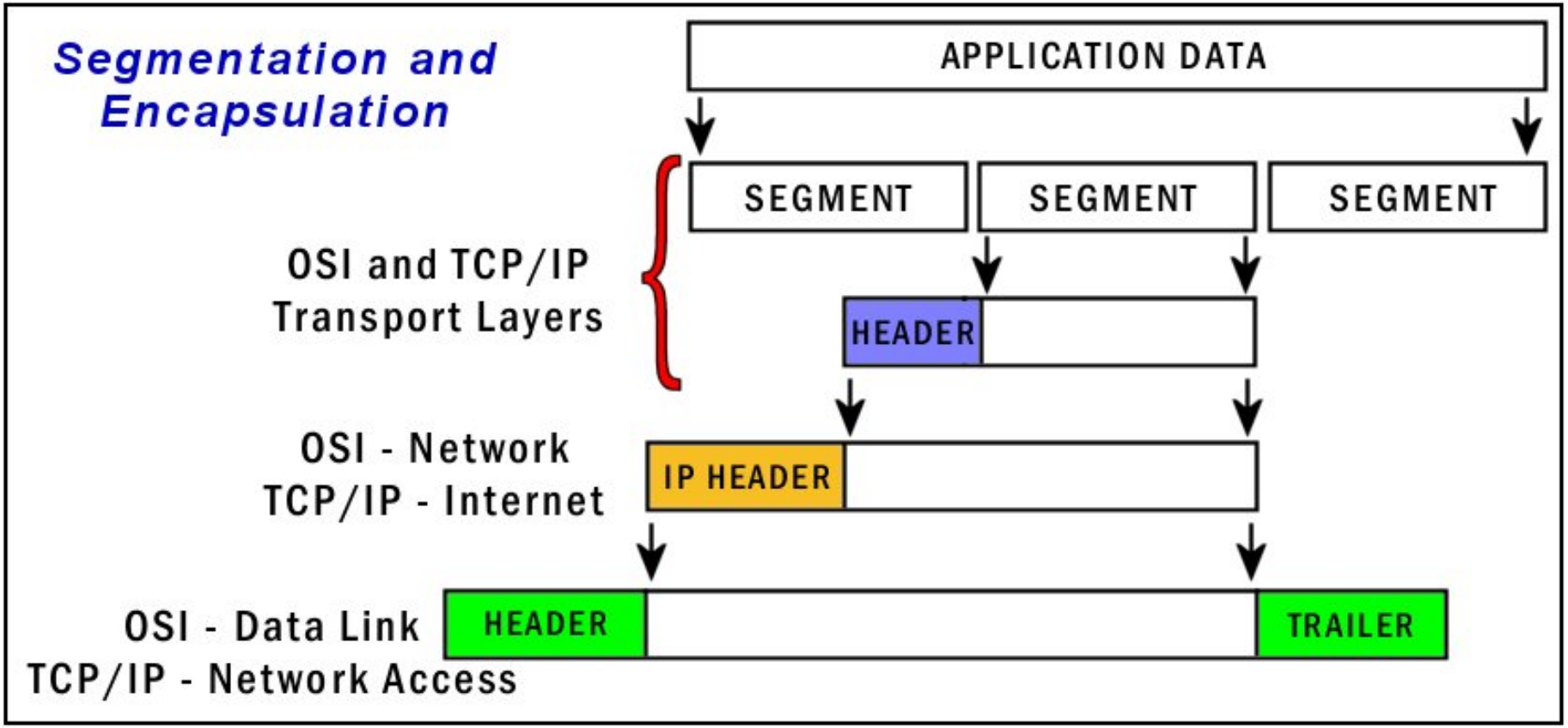


Transport Layer Functions

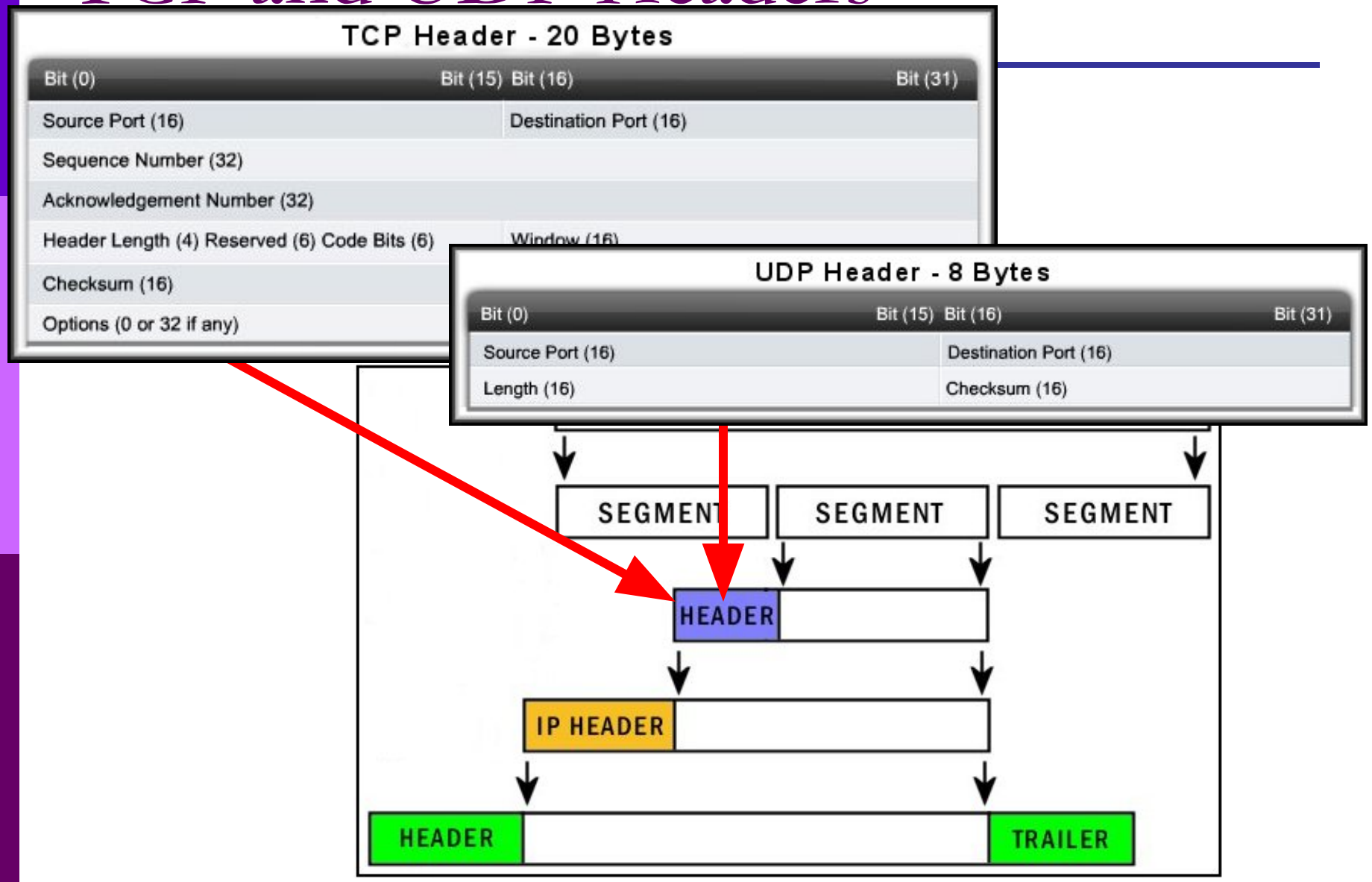
Primary Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

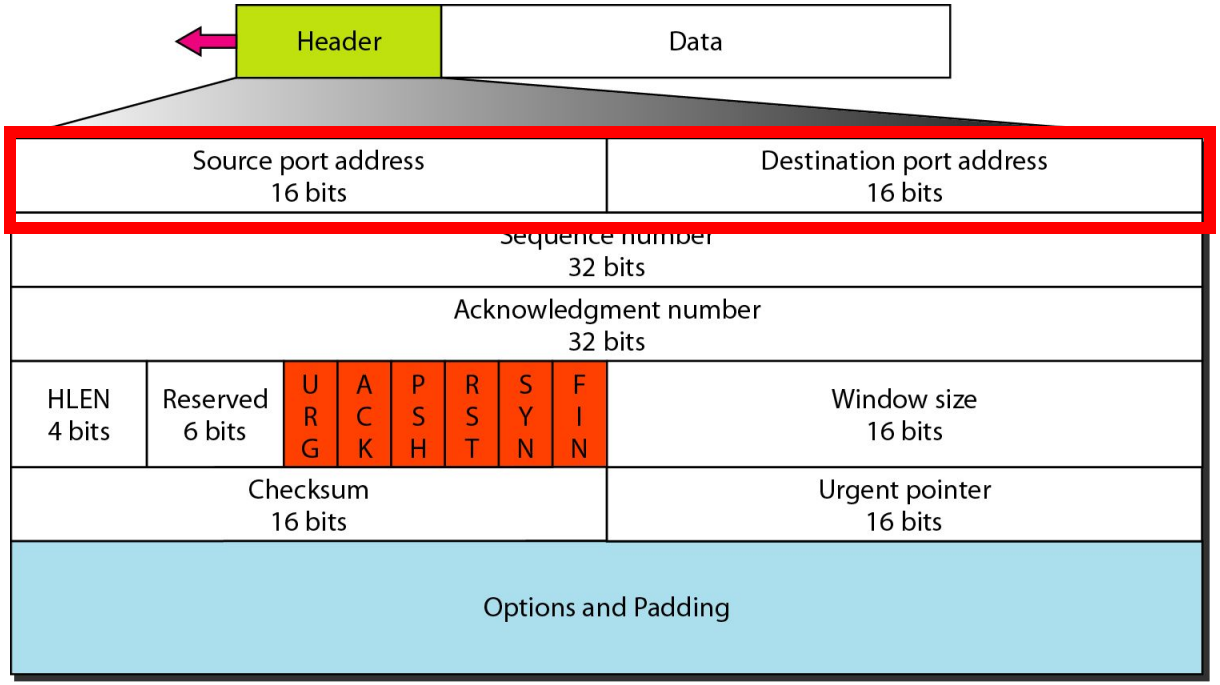
Segmentation and Reassembly



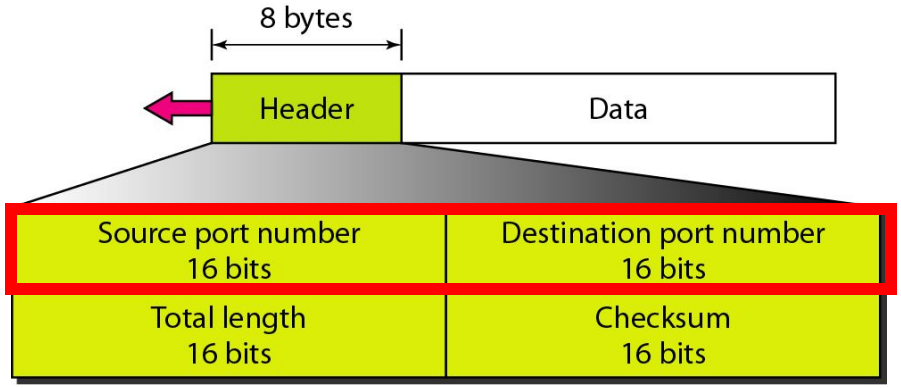
TCP and UDP Headers



TCP and UDP headers



- TCP Header



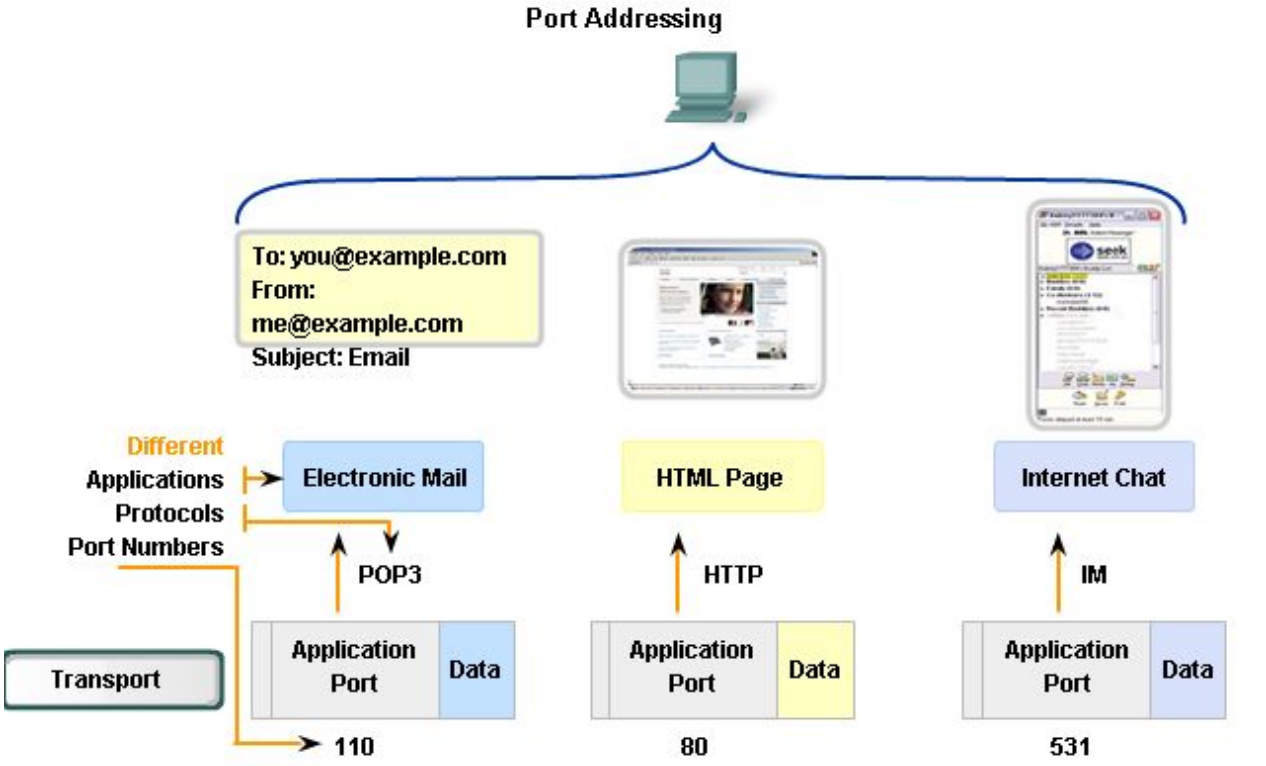
- UDP Header

Primary Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identify **PORT NUMBERS** e segmen cations.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

Port Numbers

- Used by TCP and UDP as a form of addressing.
- Identifies the application and the conversation.



Data for different applications is directed to the correct application because each application has a unique port number.

Port Numbers

- Internet Corporation for Assigned Names and Numbers (ICANN) assigns port numbers.
- Before that The Internet Assigned Numbers Authority (IANA) used to handle port numbers.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Port Addressing Types

- Well-Known Ports:
  - Reserved for common services and applications.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

20 – FTP Data

21 – FTP Control

23 – Telnet

25 – SMTP

110 – POP3

520 – RIP

443 – HTTPS

69 – TFTP

Port Addressing Types

- Registered Ports:
  - Optional user processes and applications.
  - Can be dynamically selected by a client as its source port.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

8008 – Alternate HTTP

1863 – MSN Messenger

5004 – RTP

8080 – Alternate HTTP

5060 – SIP (VoIP)

Port Addressing Types

- Dynamic Ports:
  - Assigned to a user application at connect time.

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

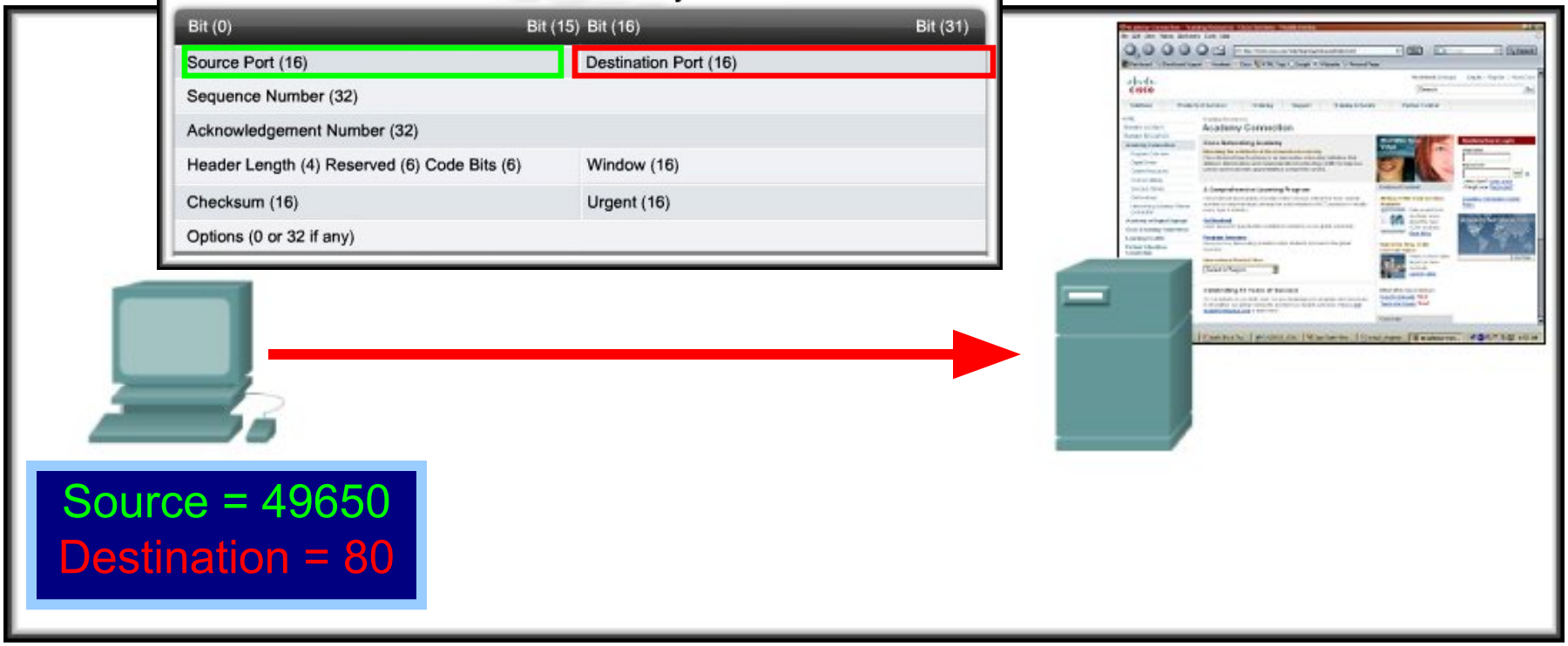
Dynamic port usage will become clearer as we move through the material.

Port Addressing Types and Tools

- Using both TCP and UDP:
  - Some applications may use both TCP and UDP.
  - For example, the low overhead of UDP enables DNS to serve many client requests very quickly.
  - Sometimes, however, sending the requested information may require the reliability of TCP. In this case, the well known port number of 53 is used by both protocols with this service.

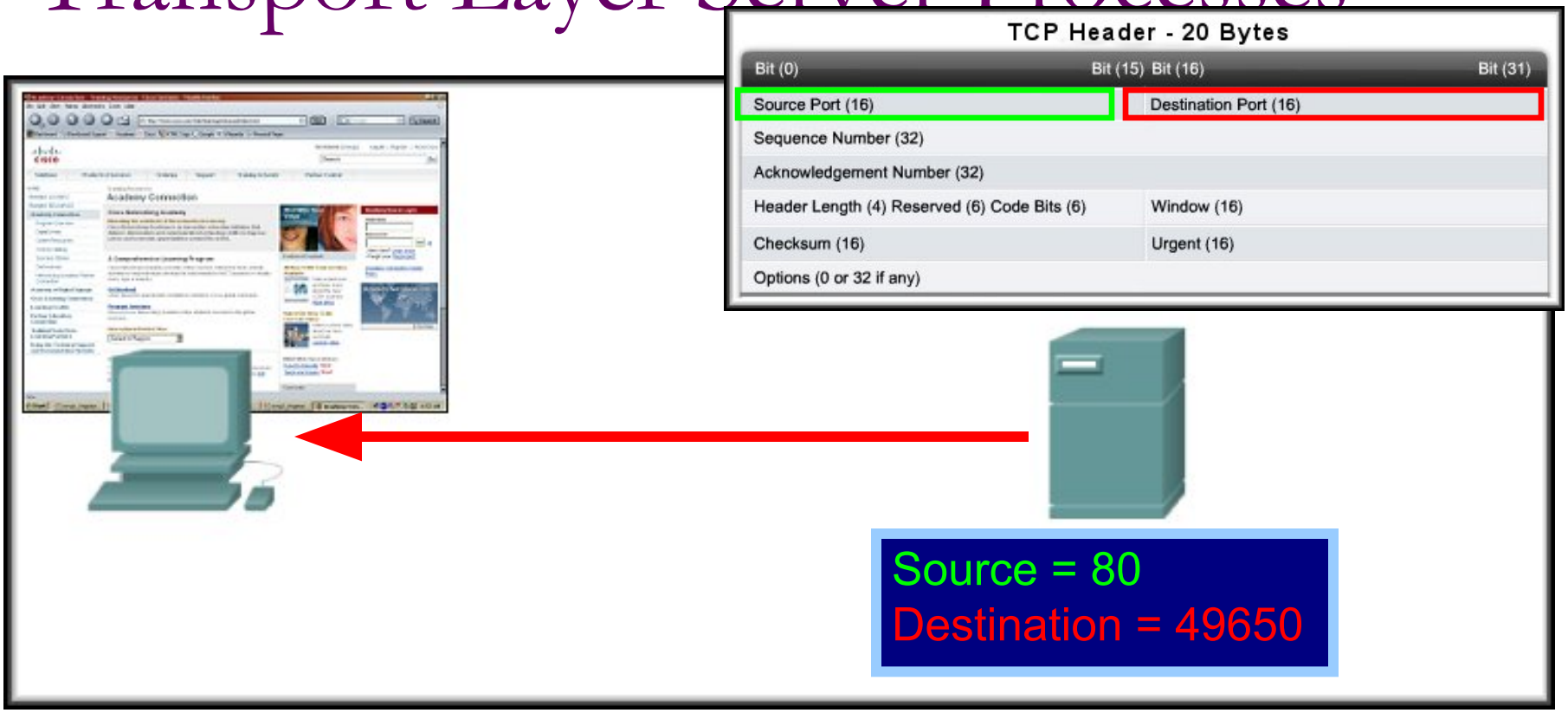
Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Transport Layer Server Processes



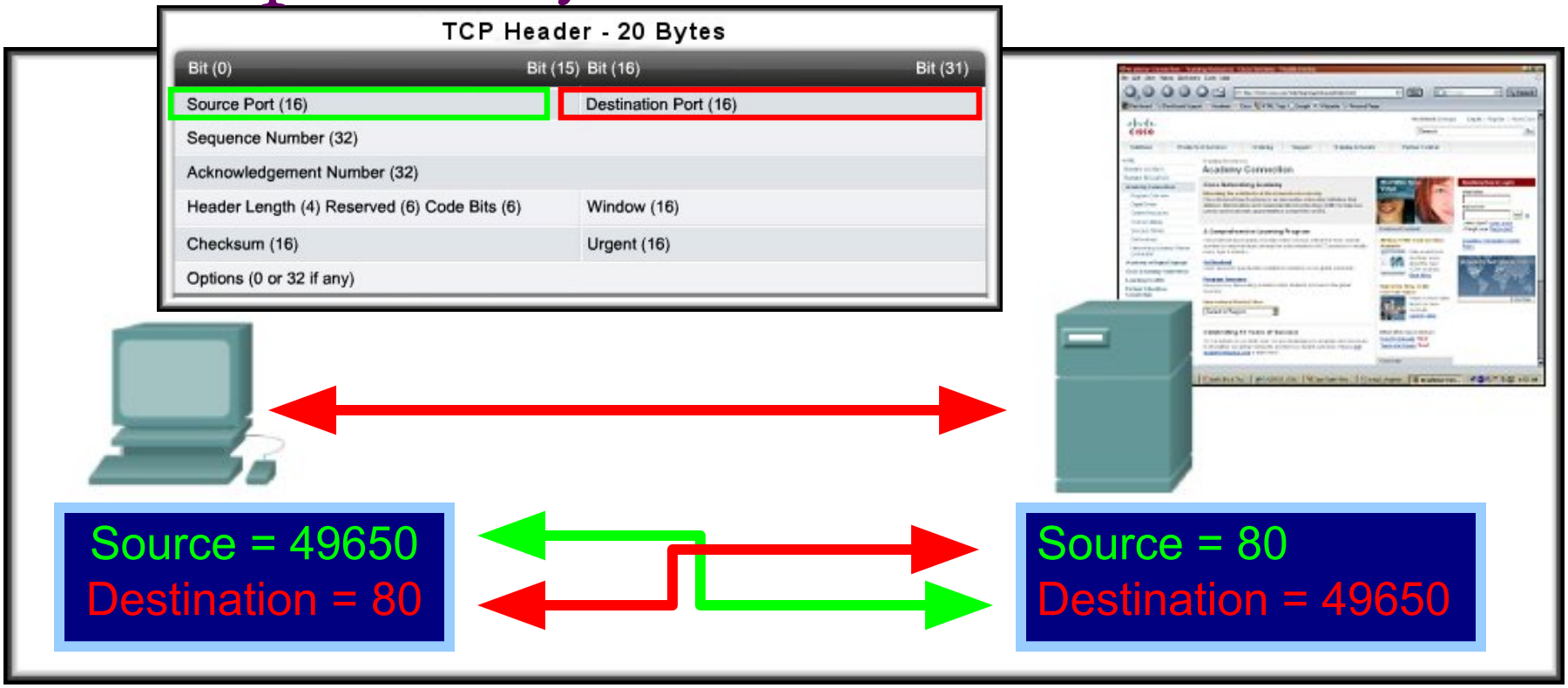
- Client application uses a dynamic port to identify itself. (Dynamically generated)
- All Web Servers runs the server side HTTP on Port 80.
- The client sets the destination port to 80.

Transport Layer Server Processes



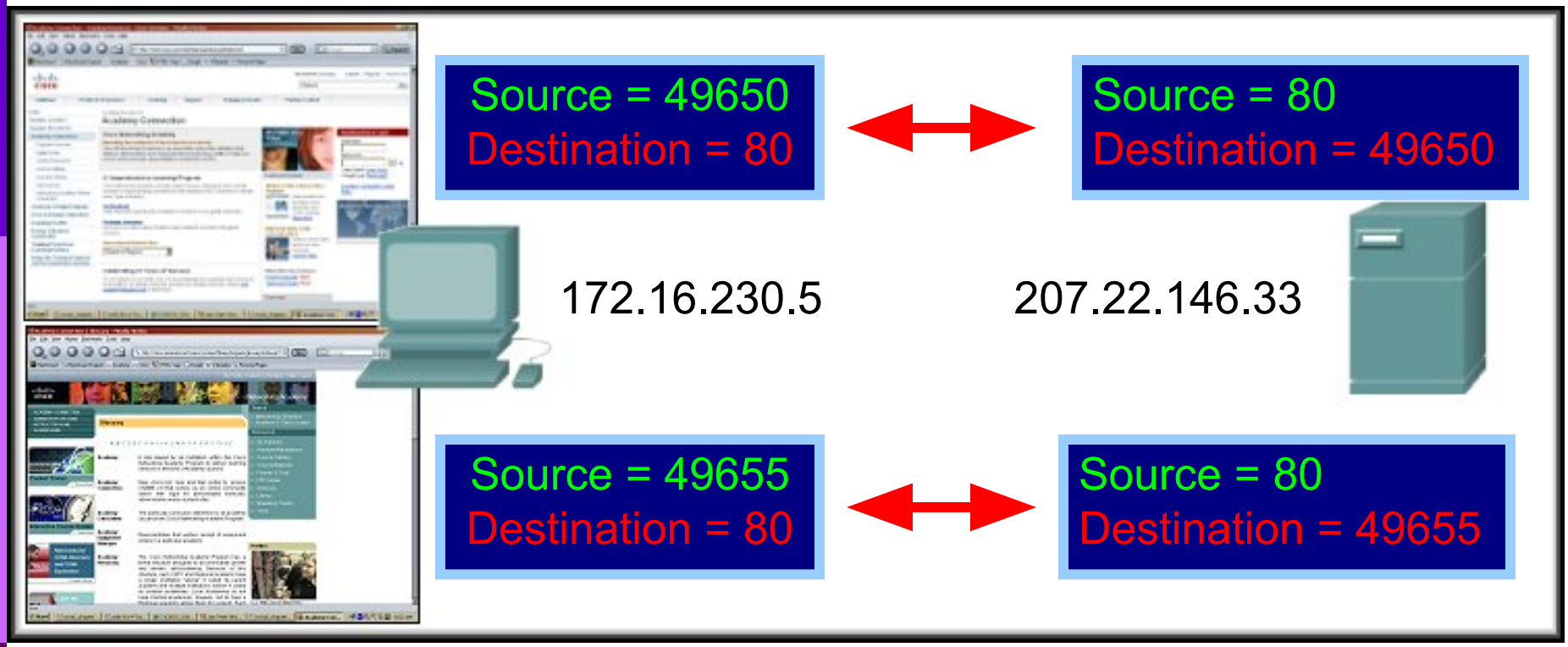
- Server replies with the web page.
  - Sets the source port to 80 and uses the client's source port as the destination.

Transport Layer Server Processes



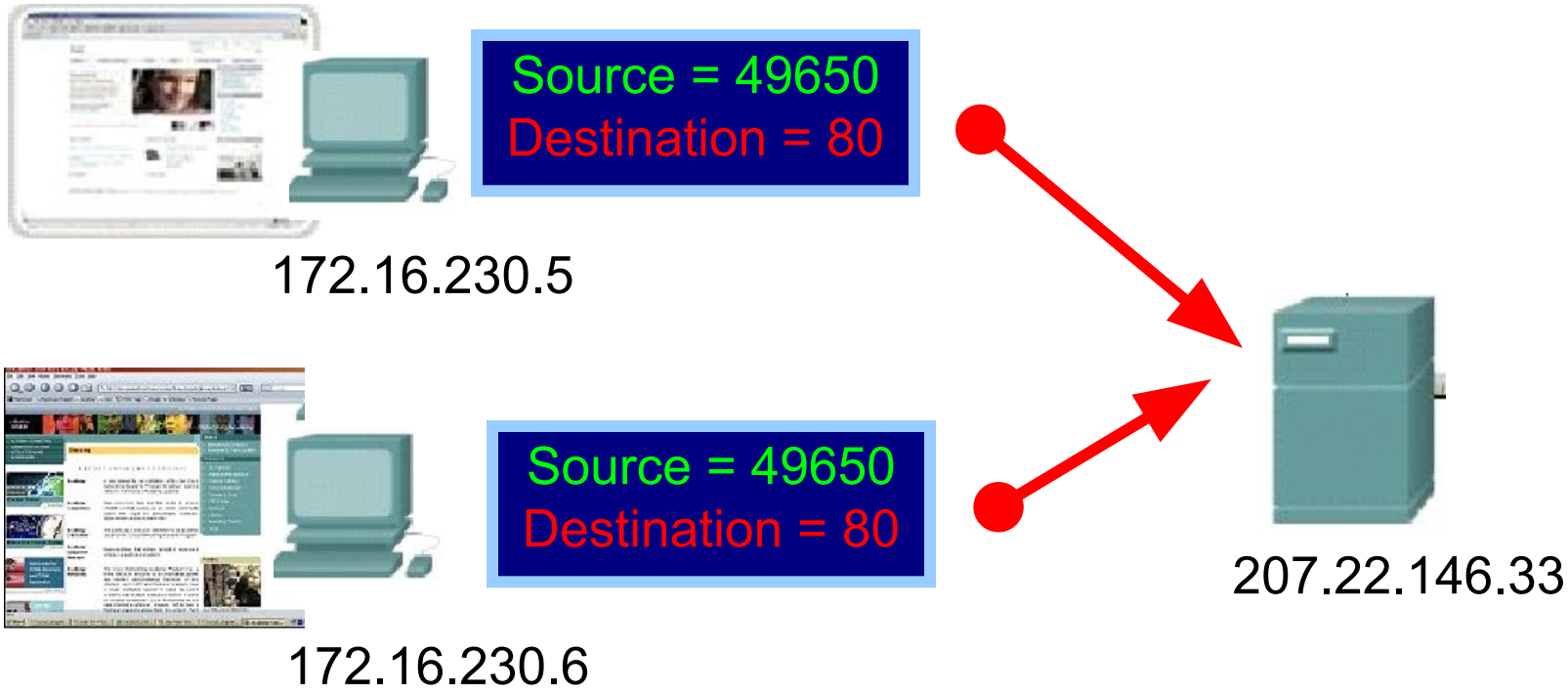
- Notice how the source and destination ports are used.

Transport Layer Server Processes



- What if there are two sessions to the same server?
  - The client uses another dynamic port as its source and the destination is still port 80.
  - Different source ports keep the sessions unique.

Socket Address

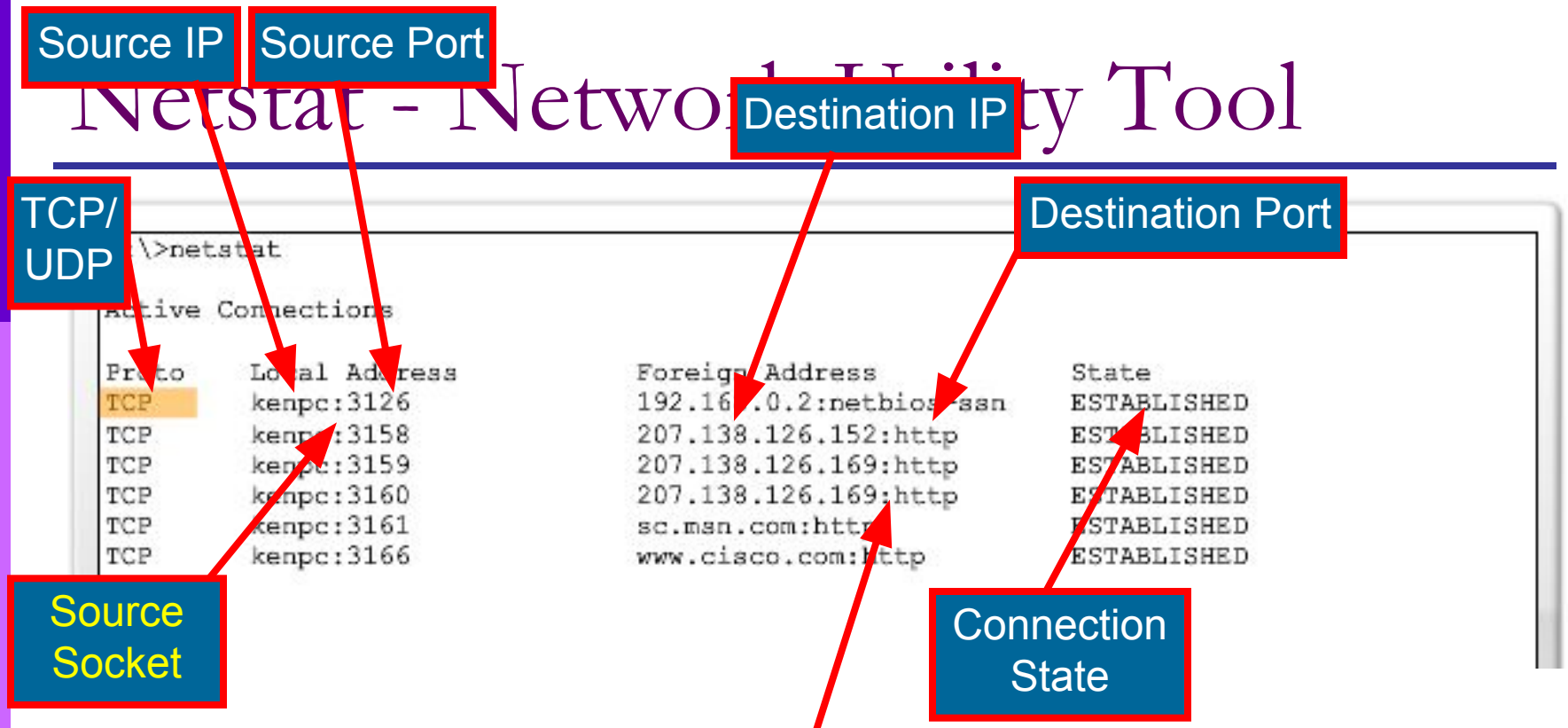


- How does the Server's Transport Layer keep them separate?
  - The socket (IP Address:Port)

172.16.230.5:49650 ⟷ **207.22.146.33:80**

172.16.230.6:49650 ⟷ **207.22.146.33:80**

Netstat - Network Utility Tool



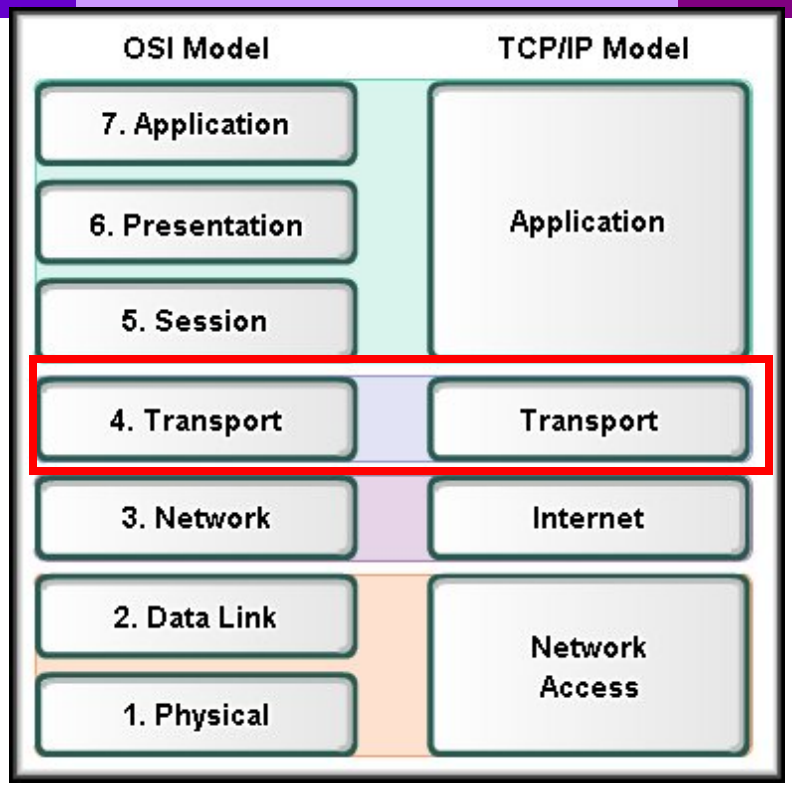
- Shows protoco address and port number, foreig ss and port number.
- Unexpected connections may mean there is a security problem.

Primary Functions:

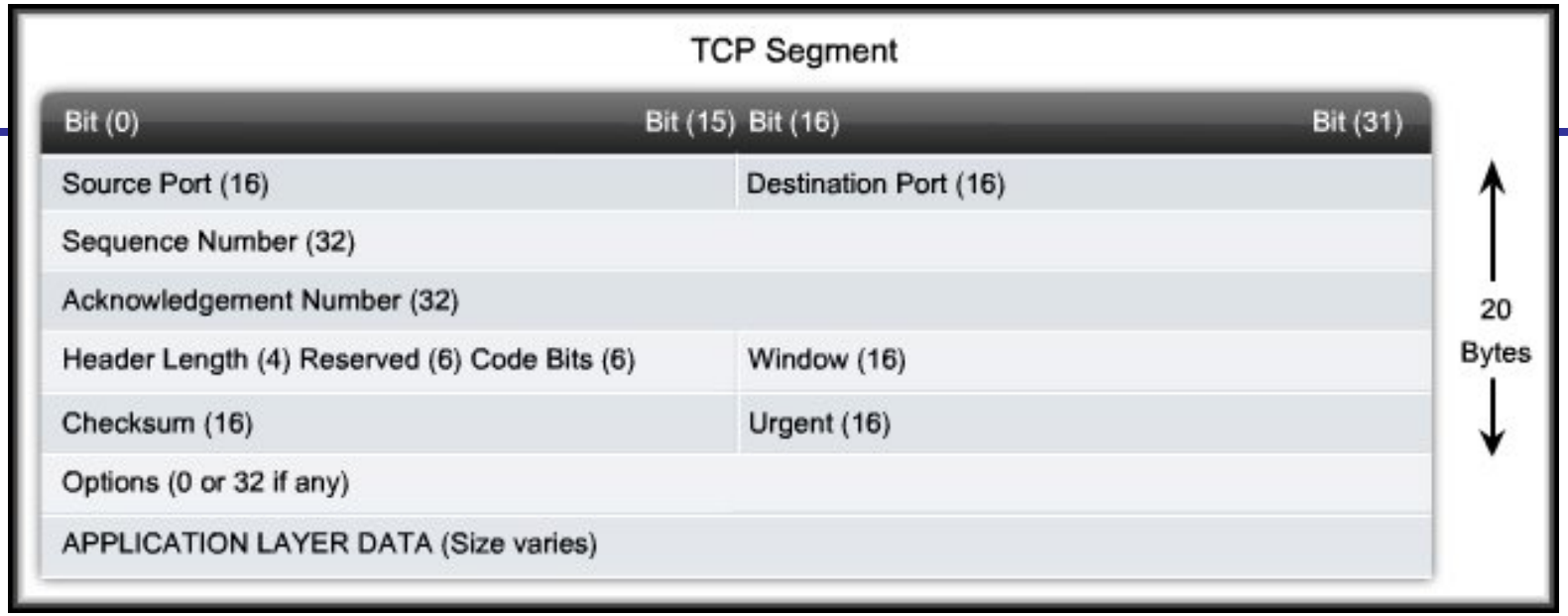
- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data tran
- Erro
- Flow control.

Performed only by TCP

TCP
Communicating with Reliability



Transmission Control Protocol



- Connection-oriented
- Reliable delivery
- Error checking
- Flow control

Example Applications

Hypertext Transfer Protocol (HTTP)

File Transfer Protocol (FTP)

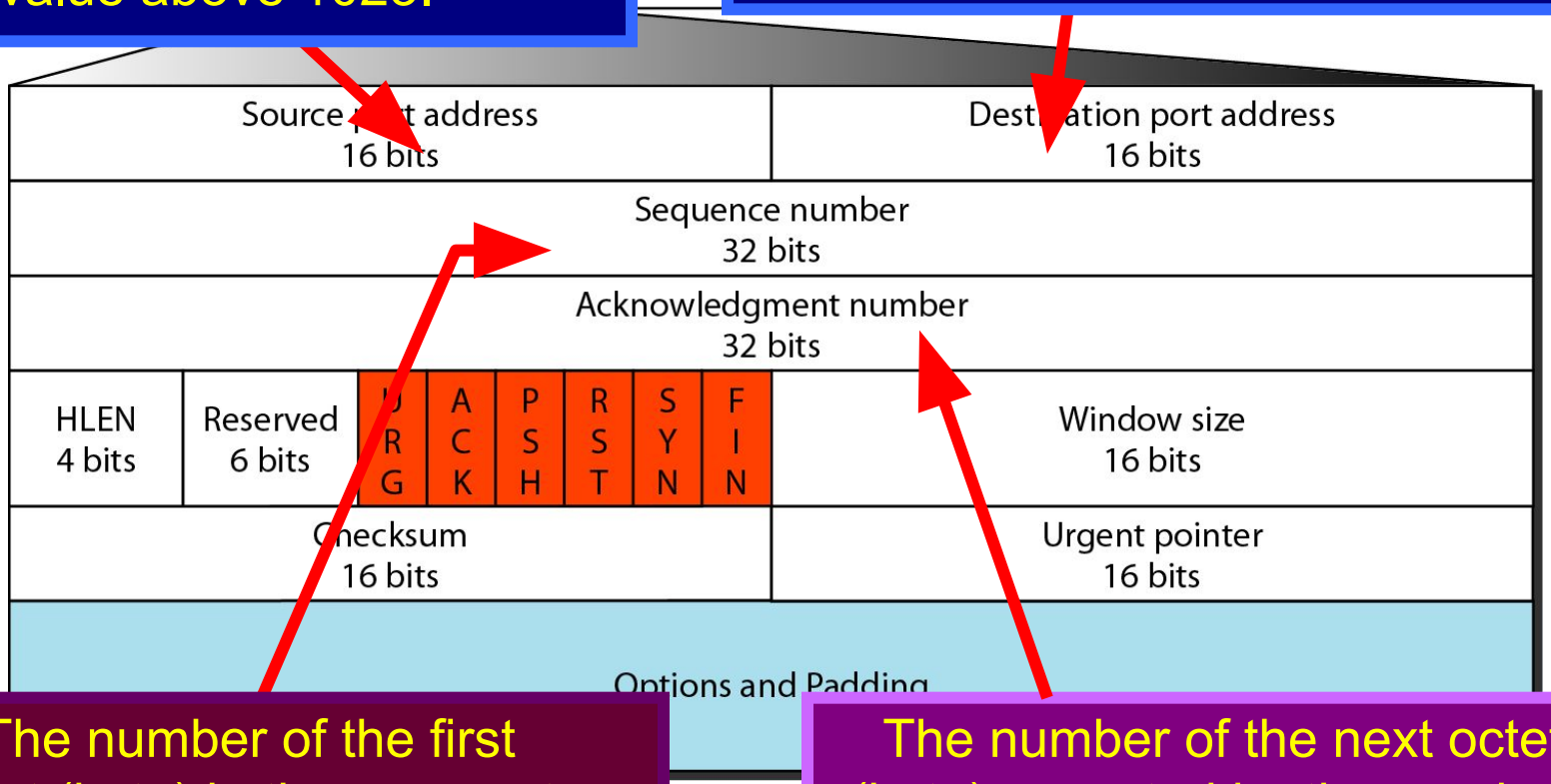
Telnet

Simple Message Transfer Protocol (SMTP)

TCP - Header

TCP session that opened a connection. Usually a random value above 1023.

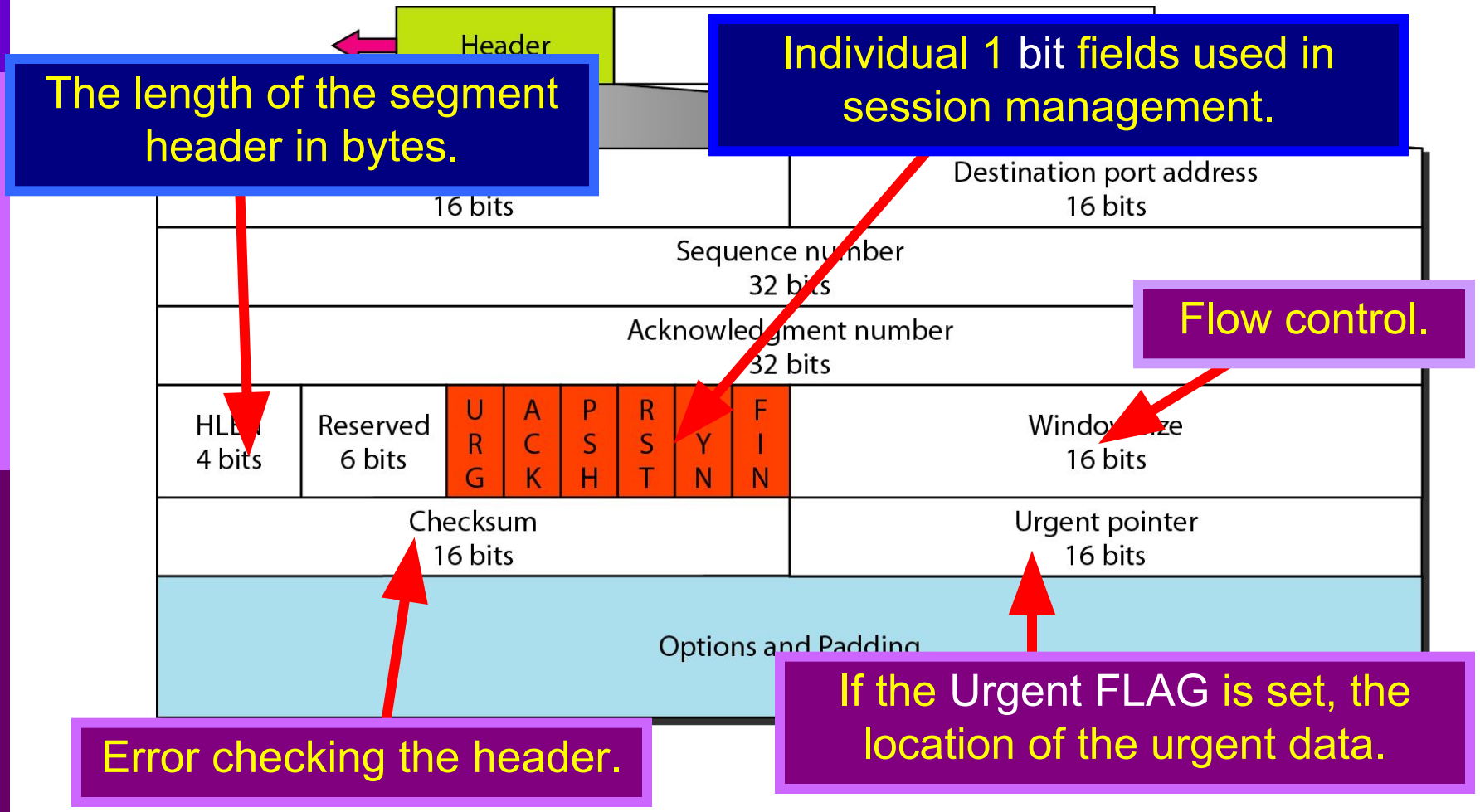
Upper Layer application on the remote site.



The number of the first octet (byte) in the segment.

The number of the next octet (byte) expected by the receiver.

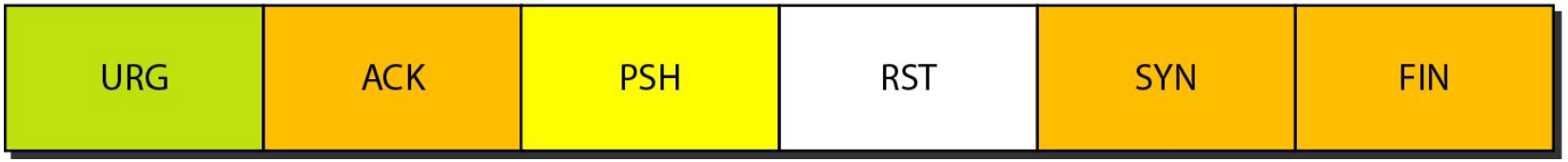
TCP - Header


Header
The length of the segment header in bytes.
Individual 1 bit fields used in session management.
16 bits
Destination port address
16 bits
Sequence number
32 bits
Acknowledgment number
32 bits
HLEN
4 bits
Reserved
6 bits
URG
ACK
PSH
RST
SYN
FIN
Window size
16 bits
Flow control.
Checksum
16 bits
Urgent pointer
16 bits
Options and Padding
Error checking the header.
If the Urgent FLAG is set, the location of the urgent data.

Control/Flag/Code Bits

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

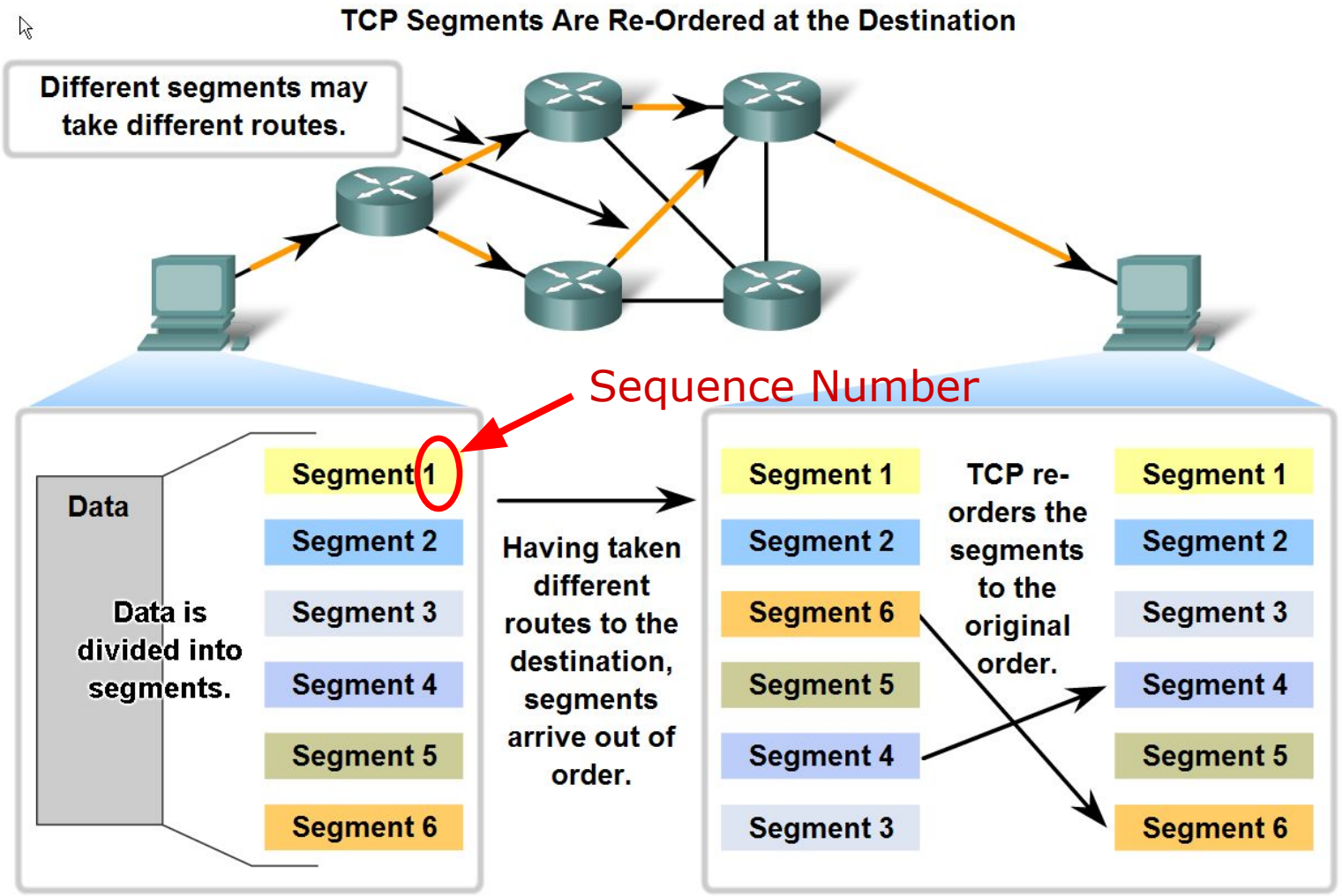


6 Bits
0 = OFF
1 = ON

TCP Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

TCP Segment Reassembly



Sequence Numbers

- The number of the first data byte contained in that 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)**

TCP Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

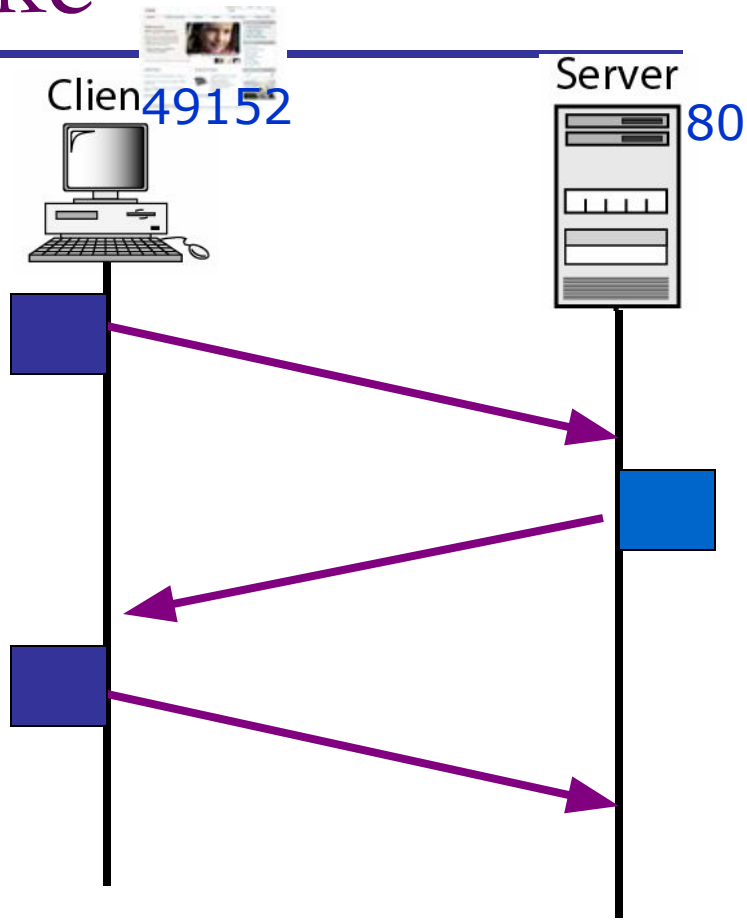
Connection Establishment

- TCP sets up a connection between end hosts before sending data
- This process is the **Three-way handshake.**
- When data transfer is finished, the hosts send signals to end the session.

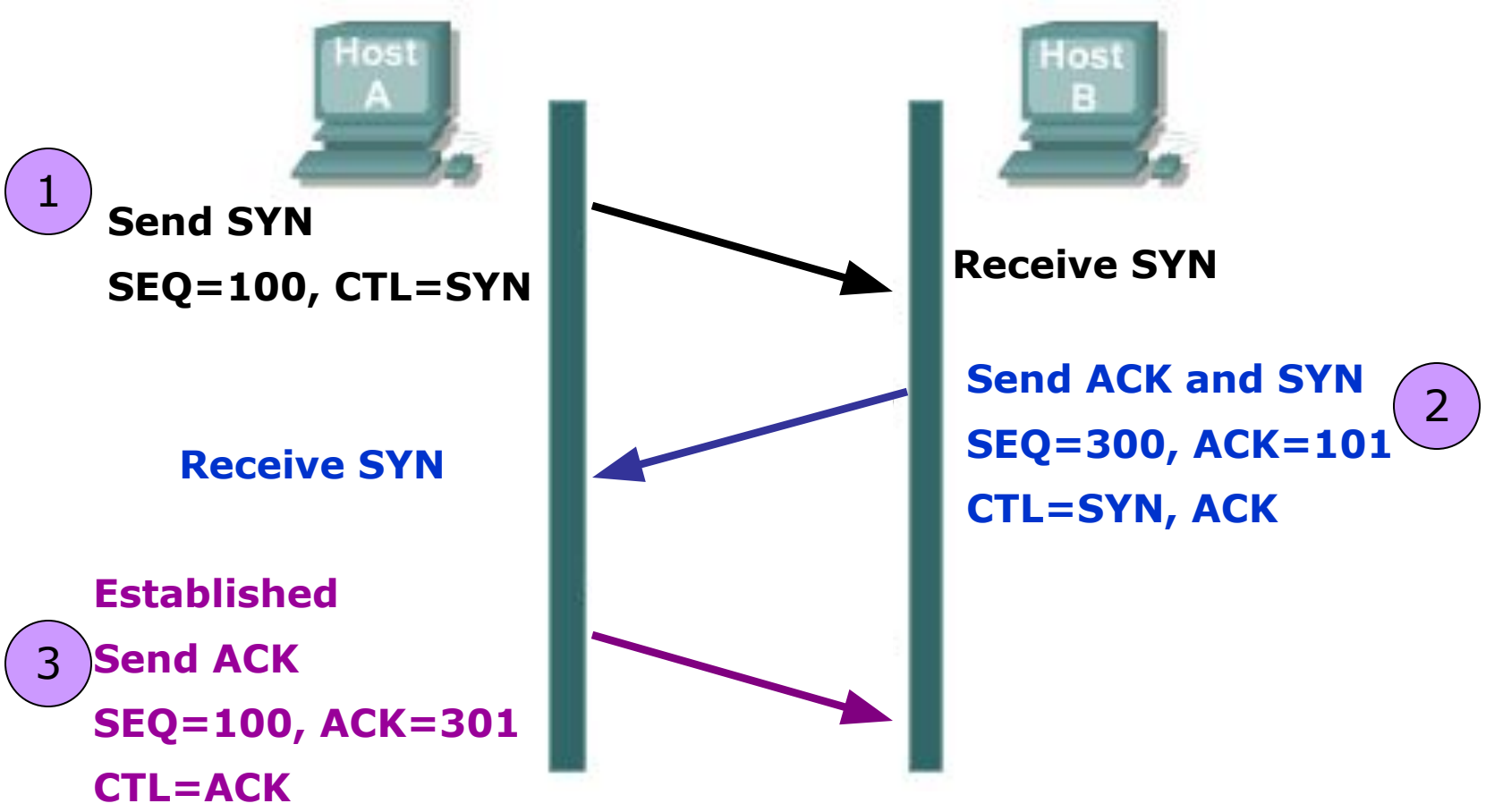
TCP Connection Establishment "Three Way Handshake"

Source Port No.	Destination Port No.
Sequence No.	
Acknowledgement No.	
U A P R S F Window Size	
Others	

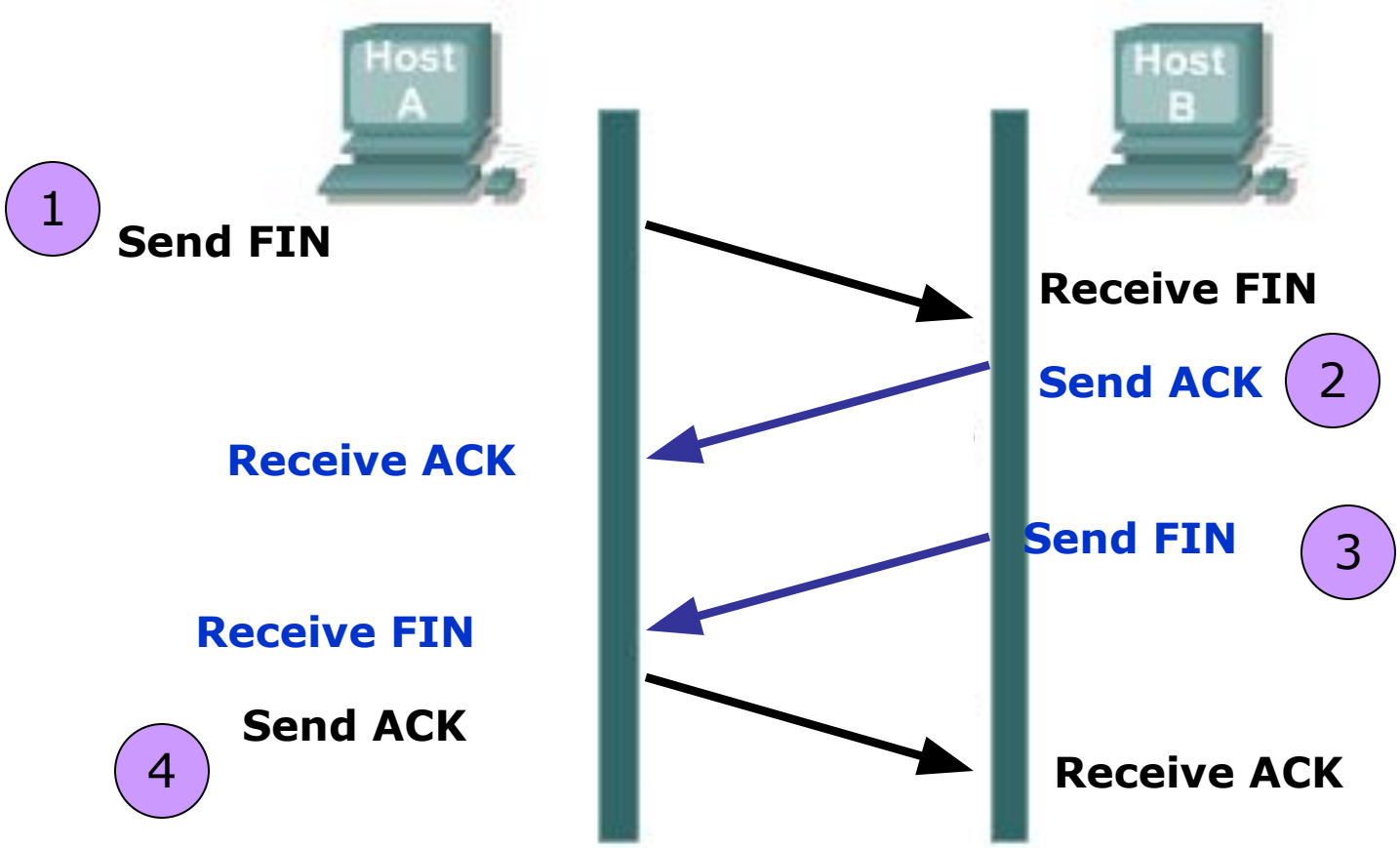
49152	80
0 0 1	
0 1 1	
A S Window Size	
Others	



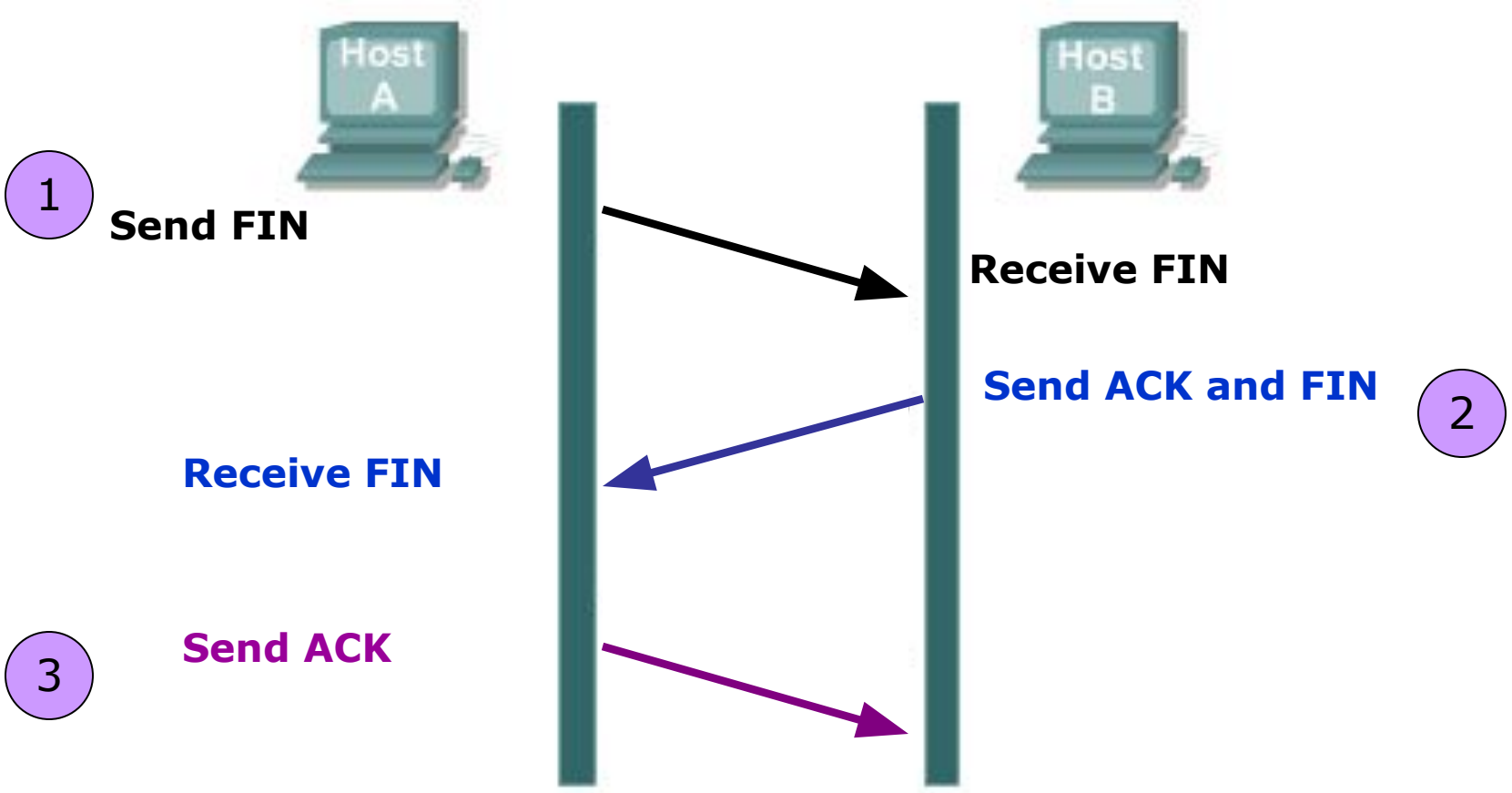
TCP Connection Establishment "Three Way Handshake"



TCP Connection Termination



TCP Connection Termination
"Three Way Termination"

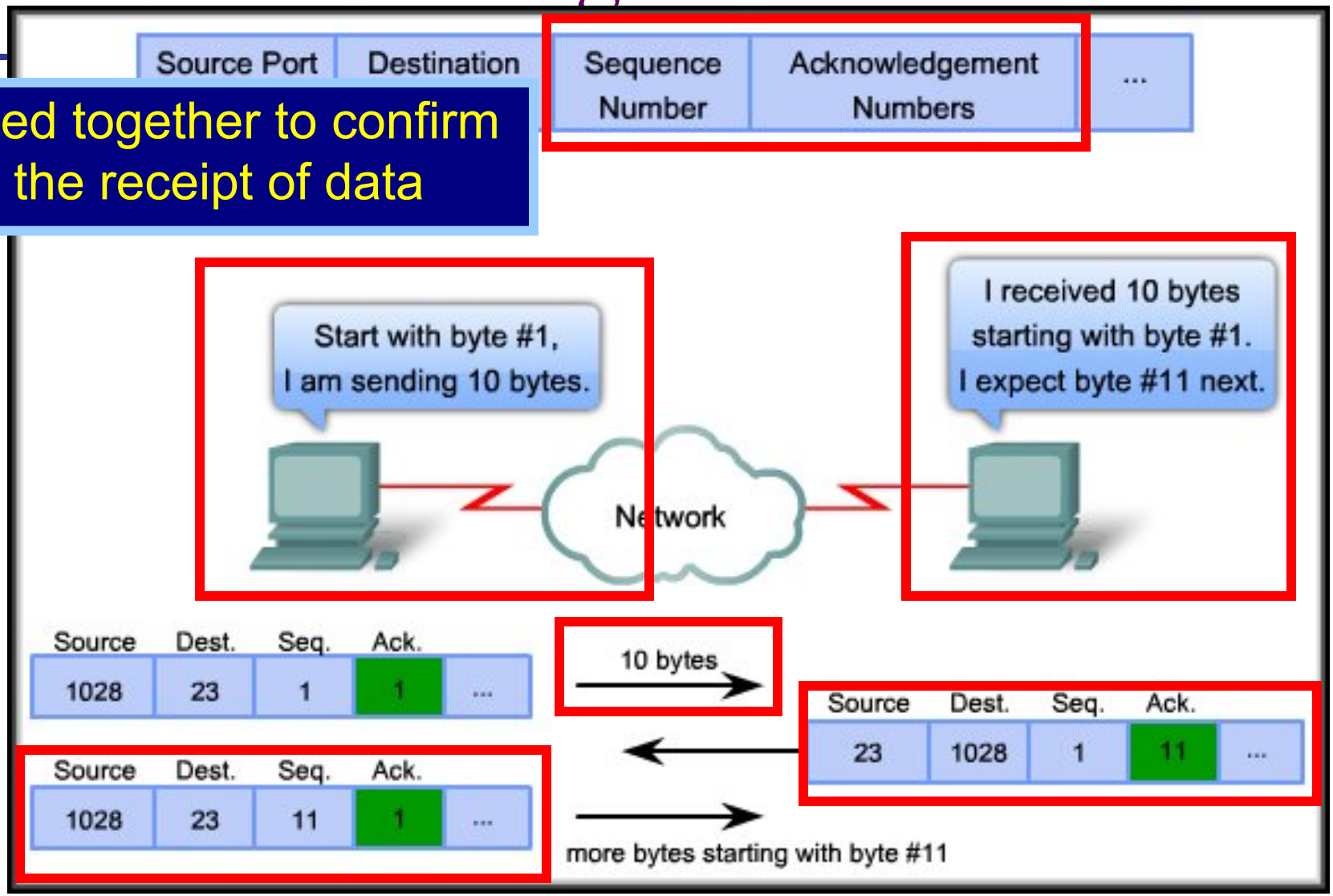


TCP Functions:

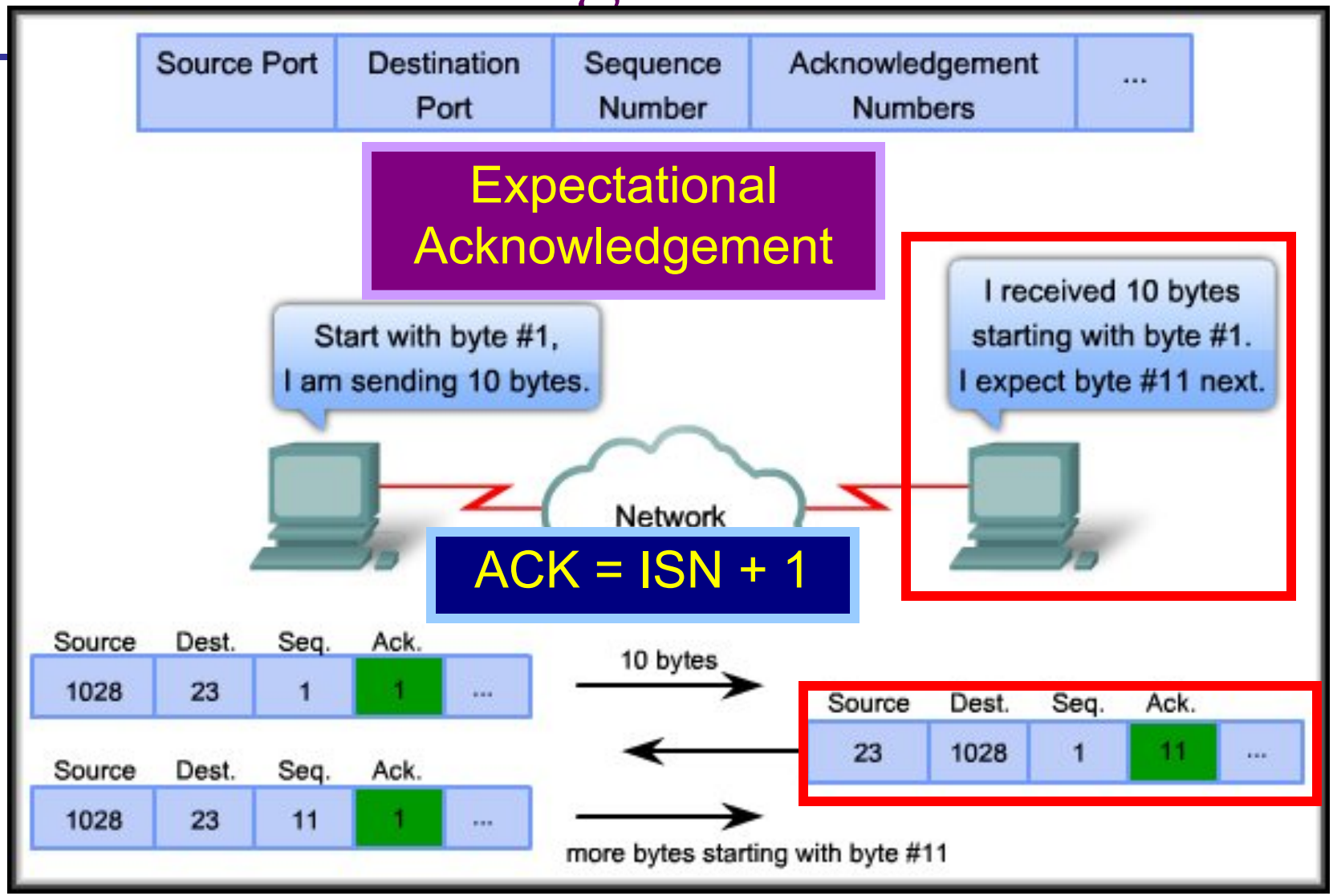
- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

TCP Acknowledgements

Used together to confirm the receipt of data

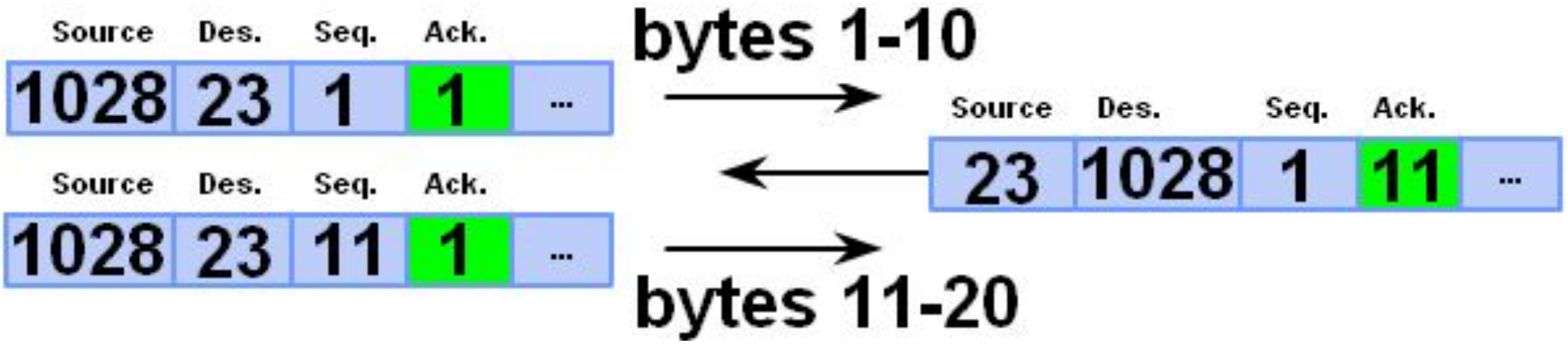


TCP Acknowledgements



Expectational acknowledgement

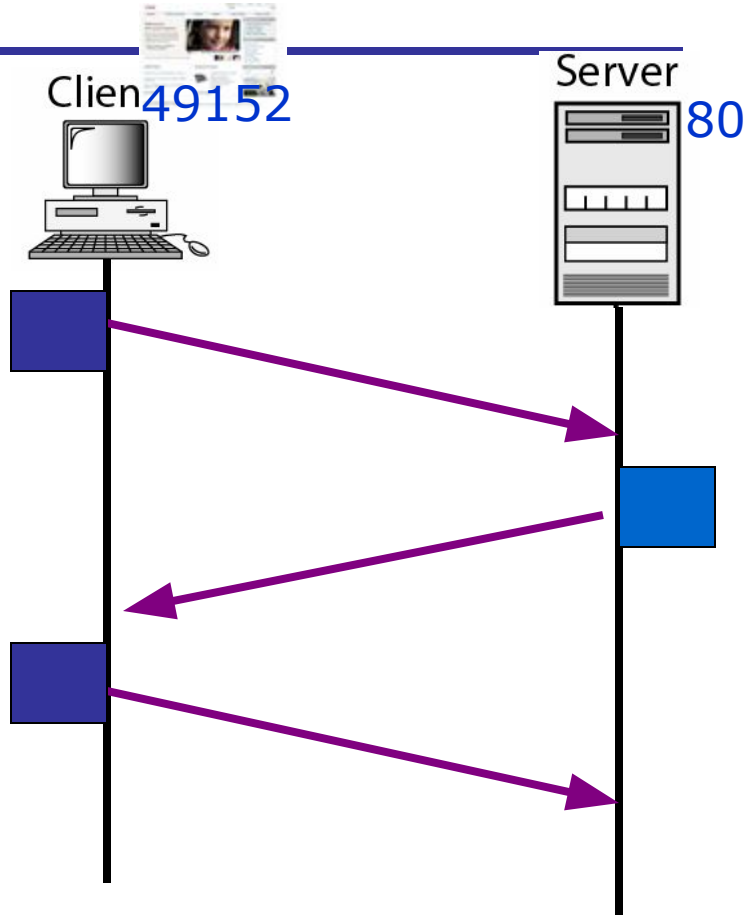
- If receiving host TCP receives uncorrupted data, then
- It sends an acknowledgement giving the sequence number of the byte that it expects next.



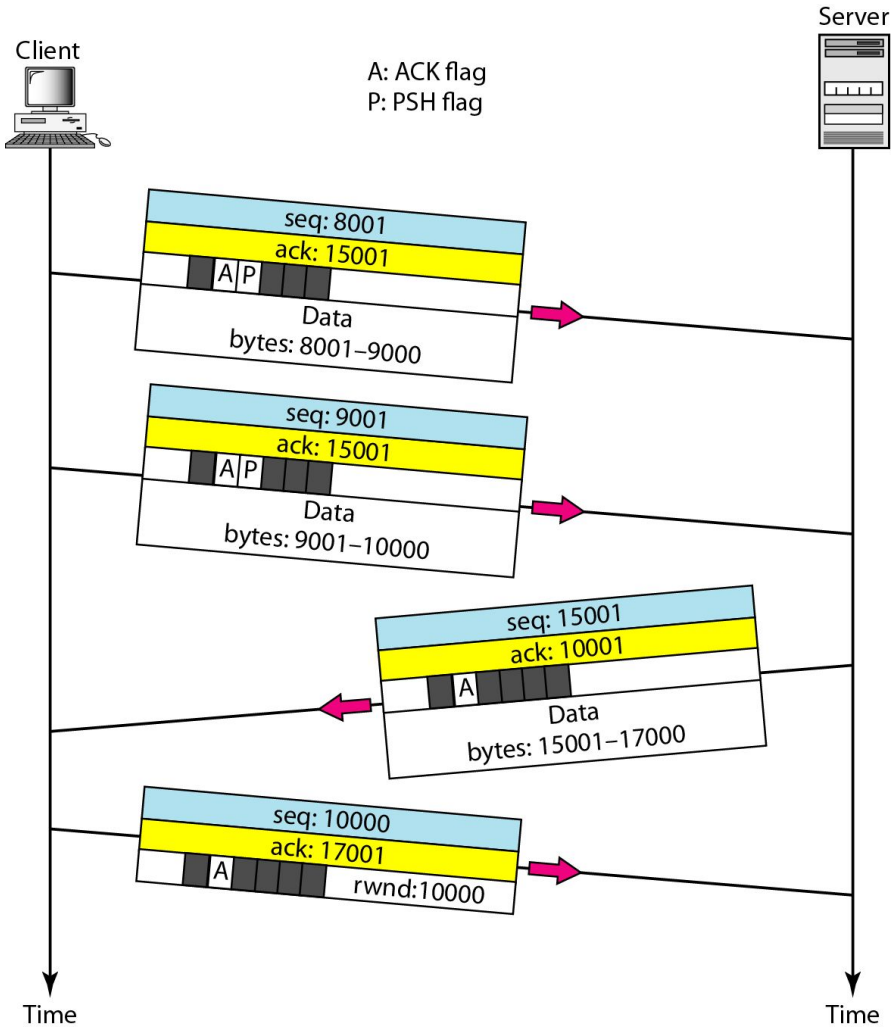
TCP Data Transfer

Source Port No.							Destination Port No.
Sequence No.							
Acknowledgement No.							
	U	A	P	R	S	F	Window Size
Others							

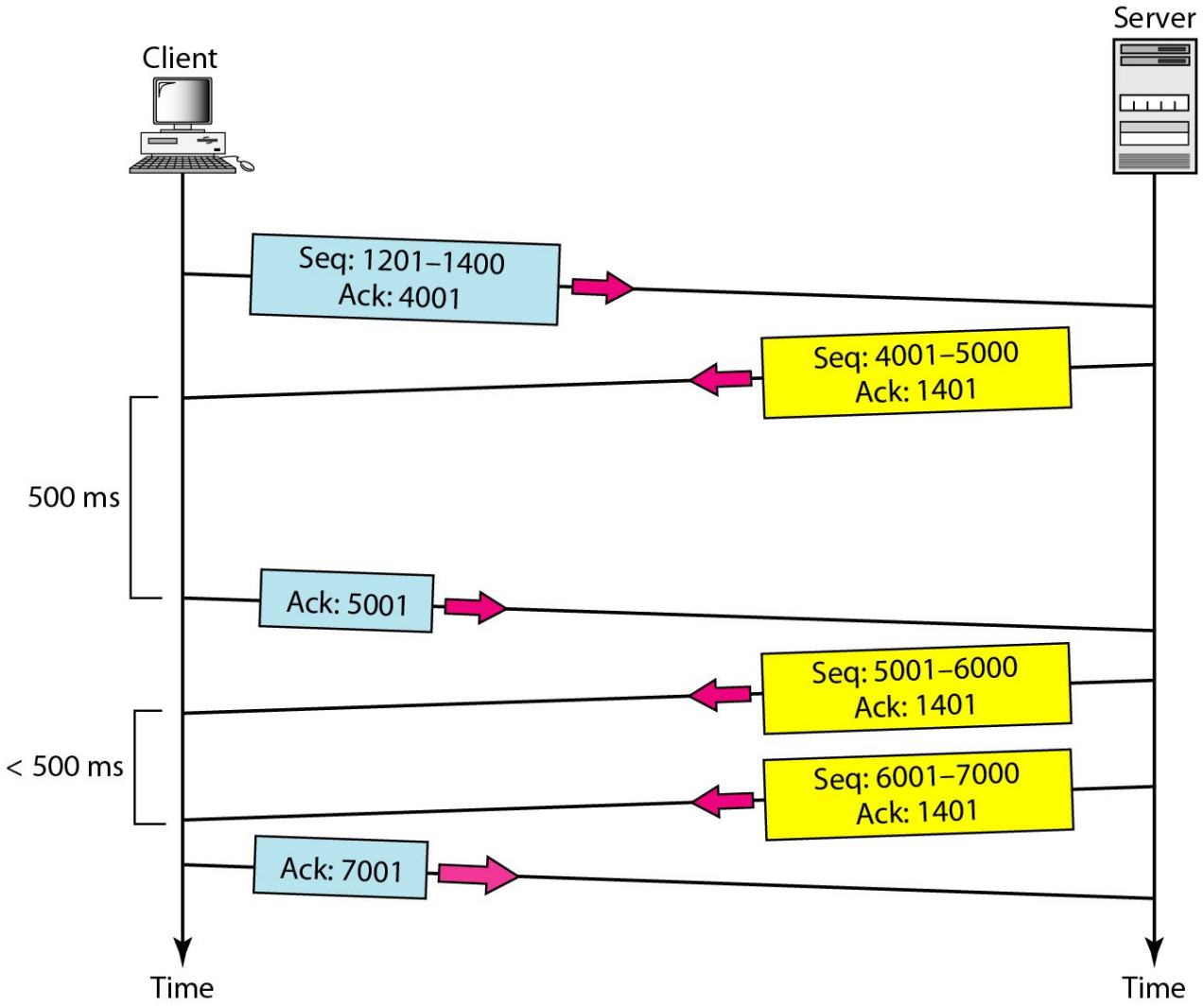
49152	80						80 49152
0 1400							
4 1060 0 1							
		A	P				Window Size
0-3 1960 bytes							



TCP Data Transfer- Multiple Segments



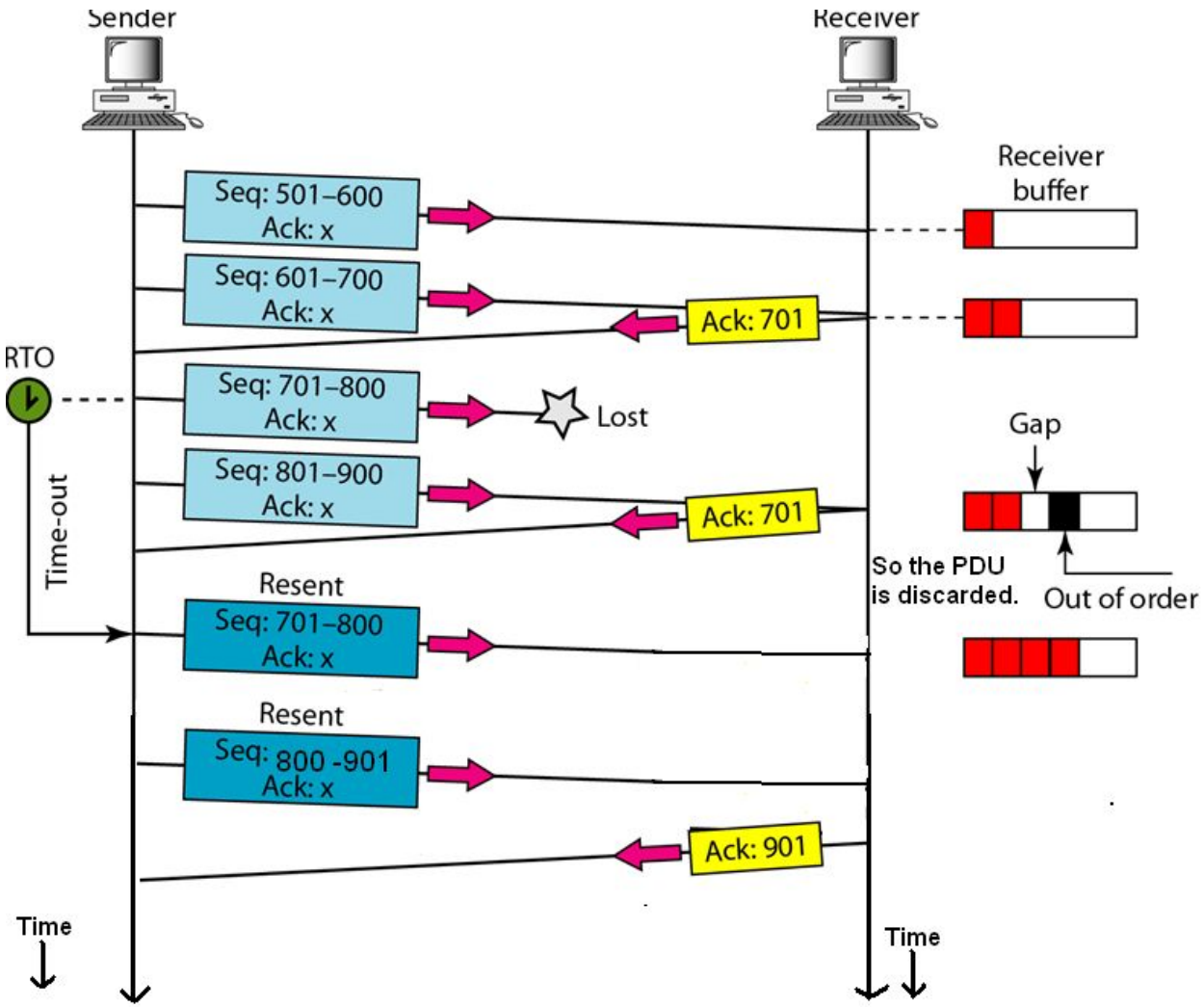
TCP Data Transfer-Reliability



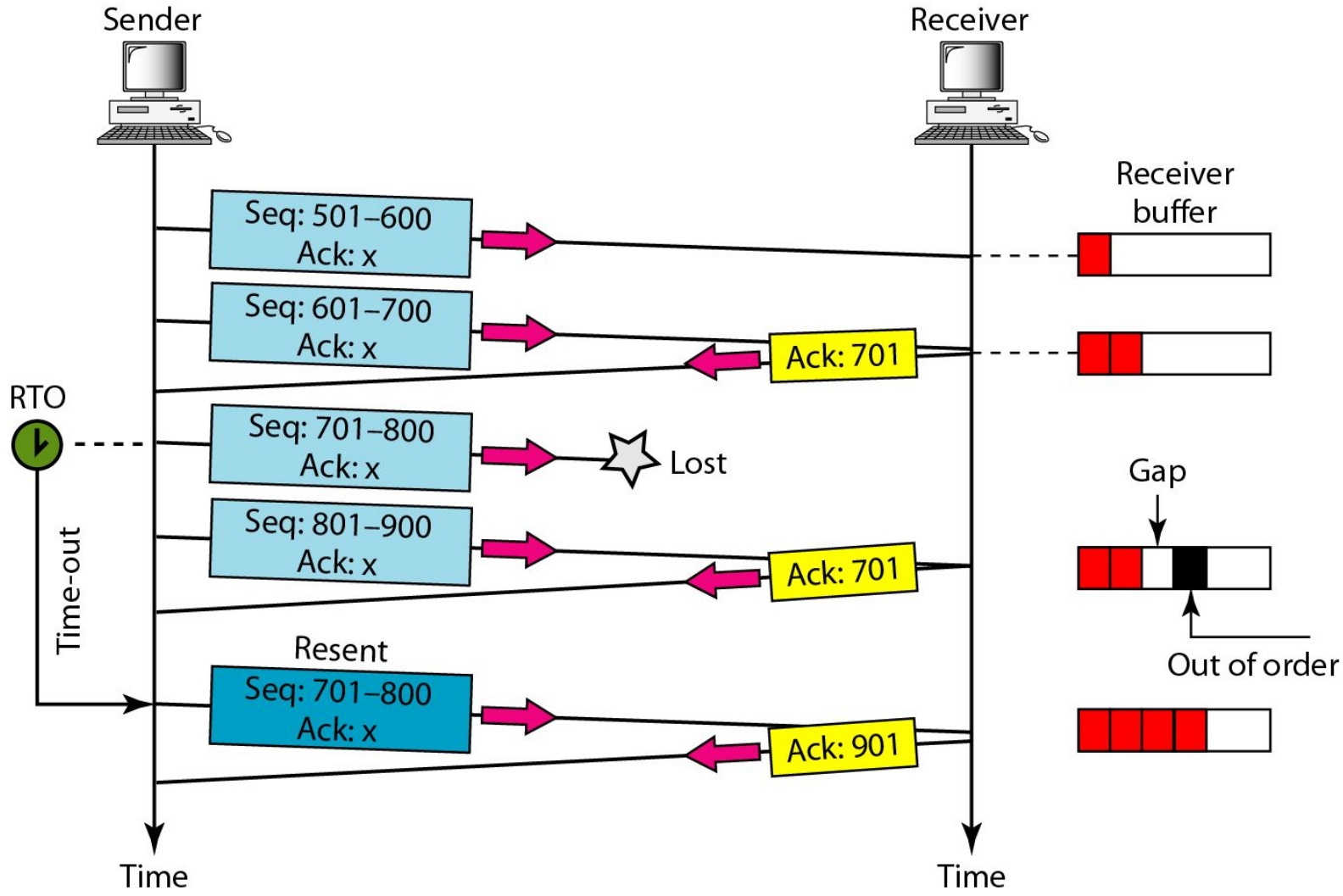
TCP Error Control

- Packet Lost or Corrupted?
- Retransmission of packet ??
- How will the sender know ??
- **Send ARQs** – Automatic Repeat Request.
- What about the receiver, not aware of a packet sent?
- **Automatic Retransmission** after time out.

TCP Error Control- Go-Back –N ARQ



TCP Error Control- Selectively Reject ARQ

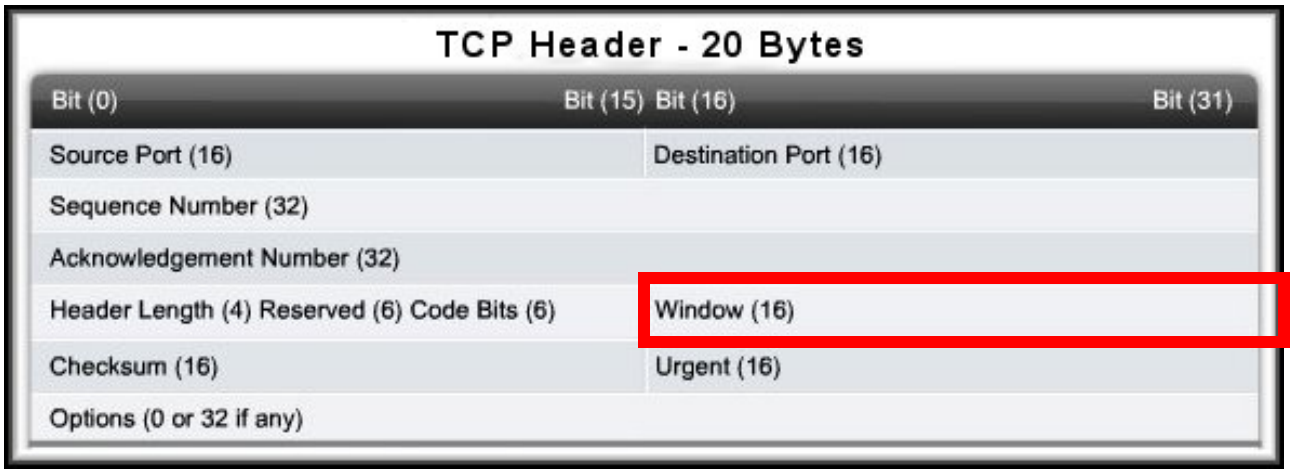


TCP Functions:

- Segmenting and Reassembling segments.
- Multiplexing segments.
- Identifying and tracking the segments of different applications.
- Establishing a connection before data transfer.
- Error Control.
- Flow control.

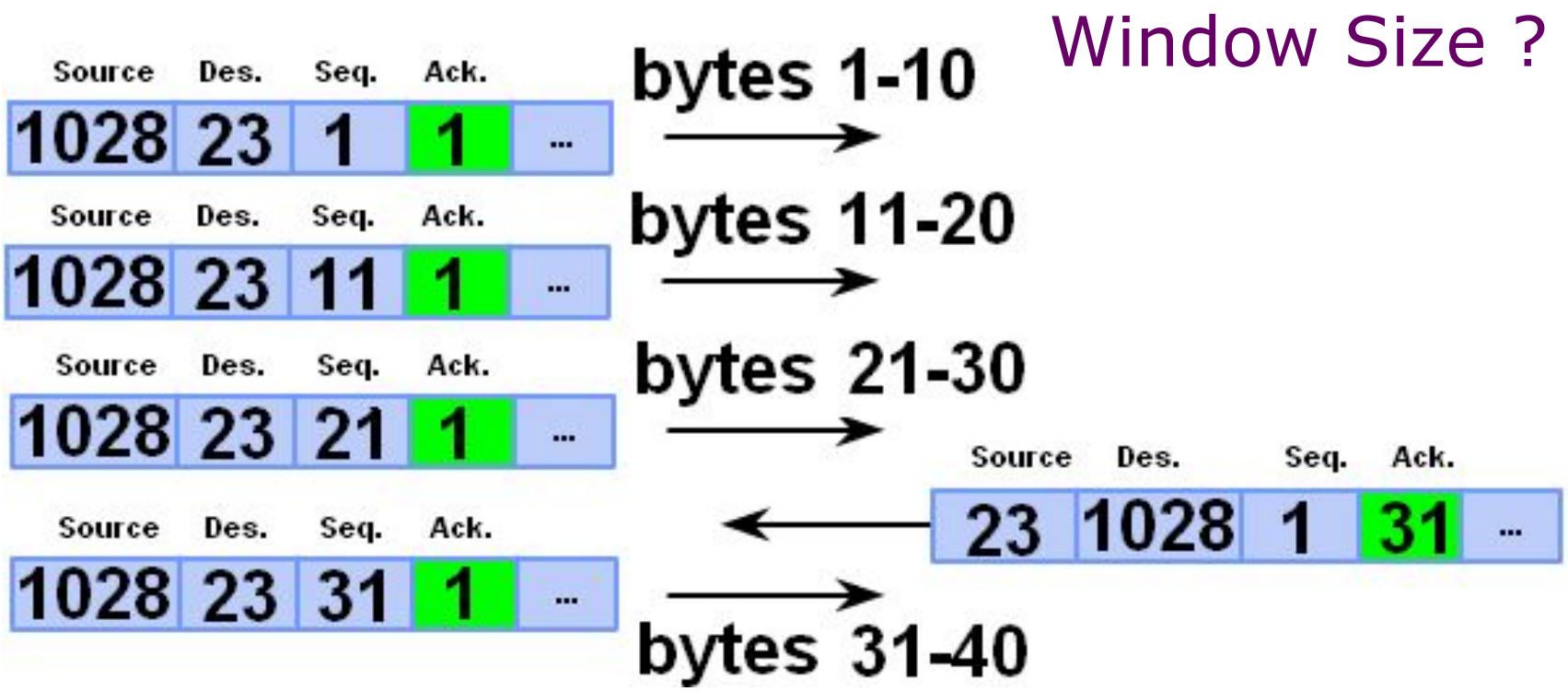
TCP Windowing

- With a window size of 10
  - Each segment carries only ten bytes of data.
  - And must be acknowledged before another segment is transmitted.

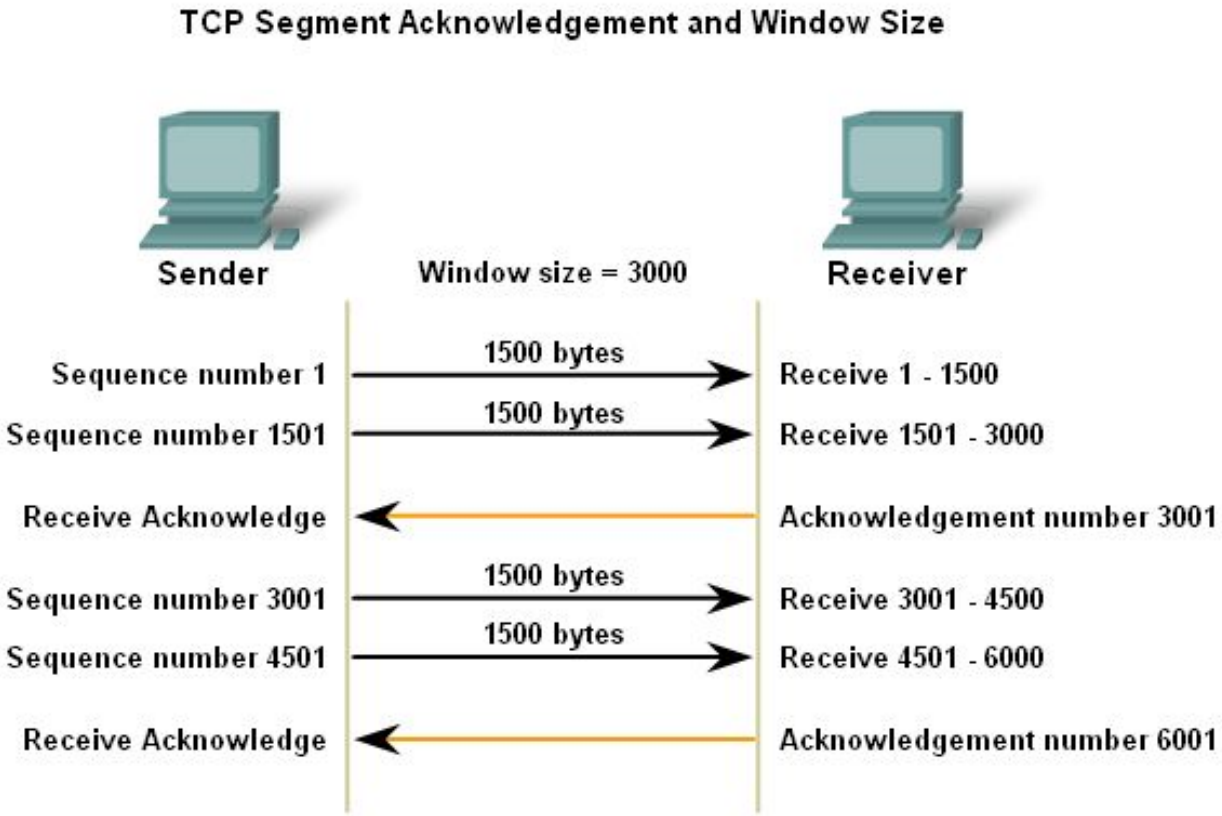


Window size

- Controls how many bytes are sent before an acknowledgement is expected.

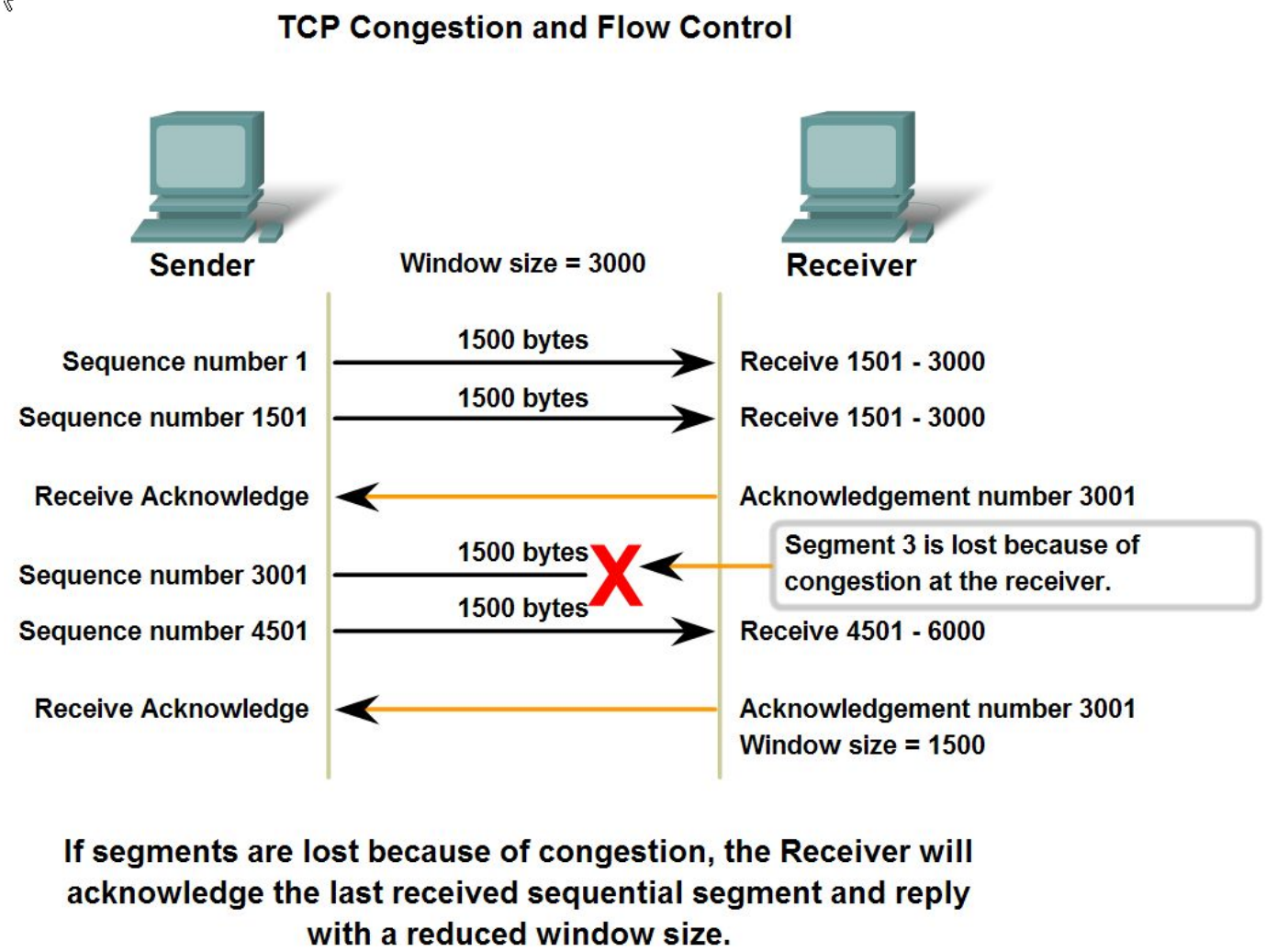


TCP Flow Control



The window size determines the number of bytes sent before an acknowledgment is expected.
The acknowledgement number is the number of the next expected byte.

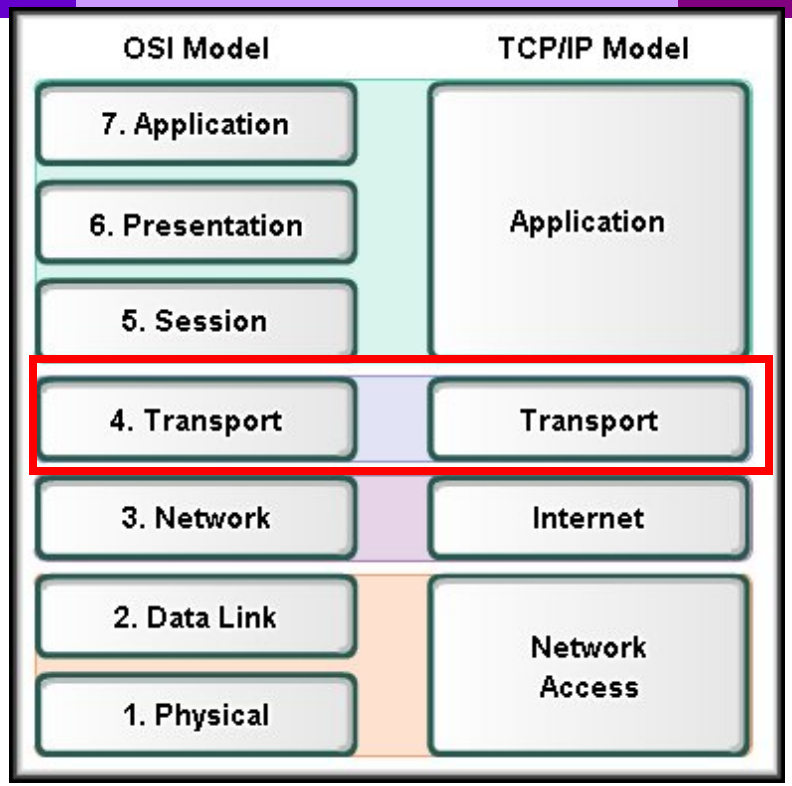
TCP Flow Control



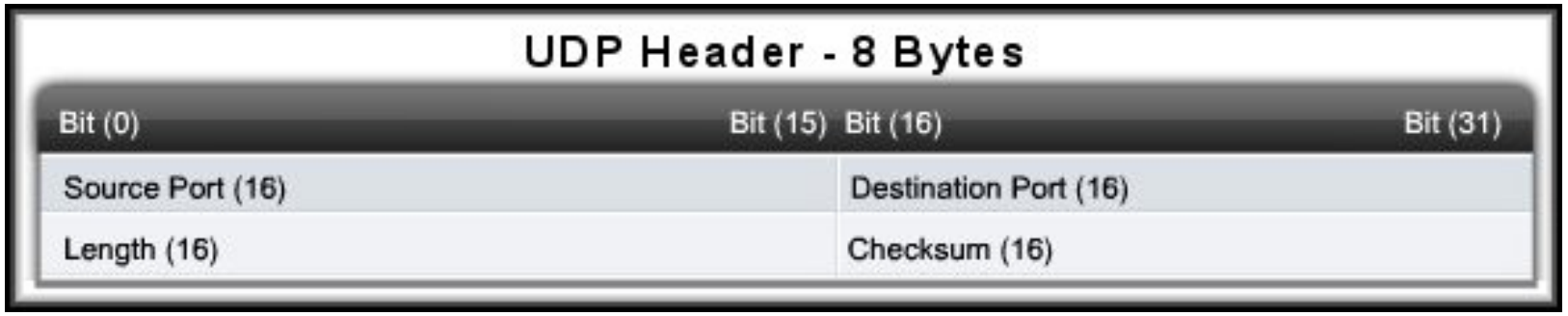
Flow control

- The initial window size is agreed during the three-way handshake.
- If this is too much for the receiver and it loses data (e.g. buffer overflow) then it can decrease the window size.
- If all is well then the receiver will increase the window size.

UDP
Communicating with No Reliability



User Datagram Protocol (UDP)



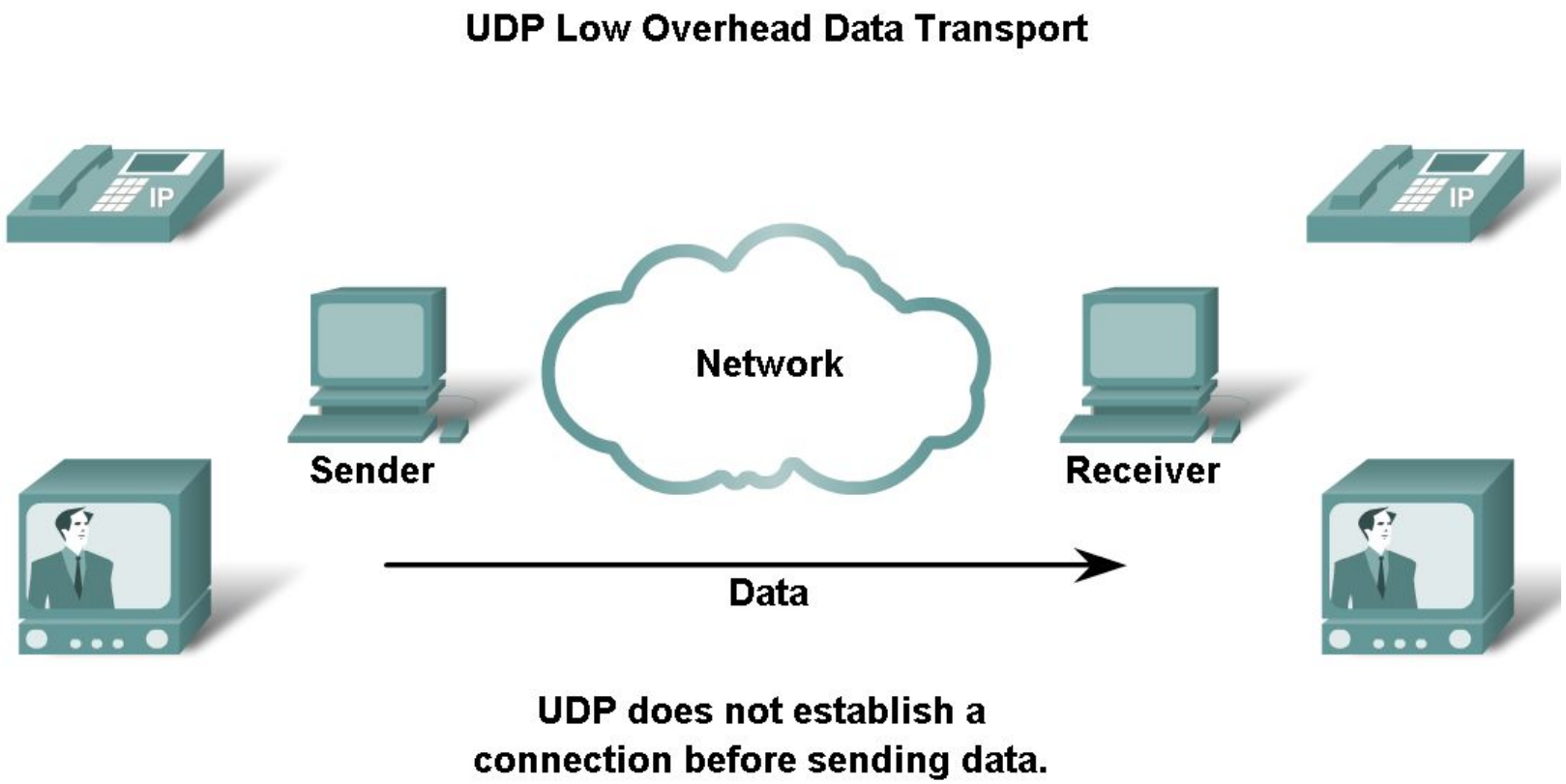
- Connectionless
- No reassembly to order.
- No Error checking
- No Flow control

Example Applications

Domain Name Service (DNS)
Trivia File Transfer Protocol (TFTP)
Dynamic Host Configuration Protocol (DHCP)

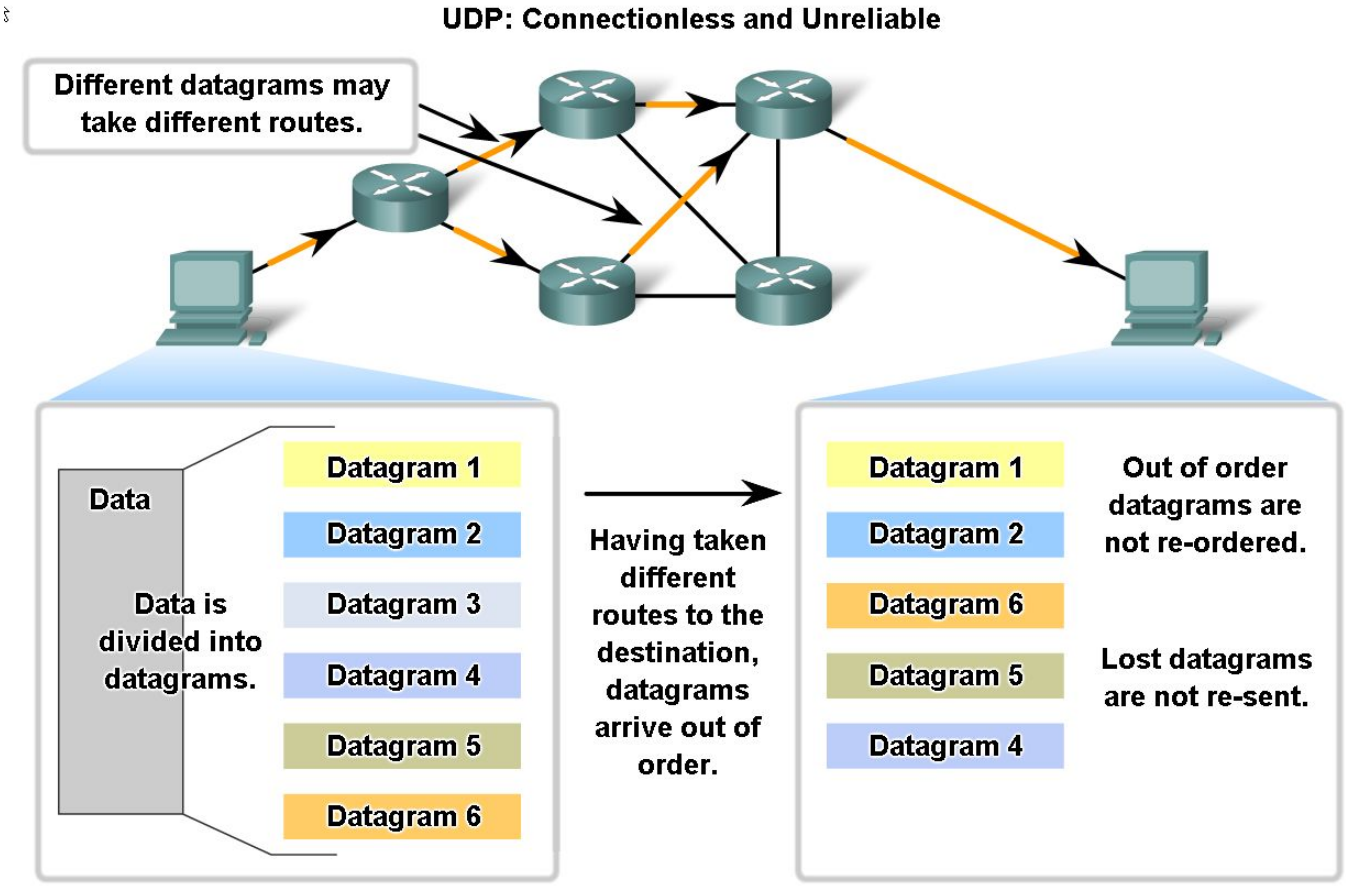
UDP

- Connectionless



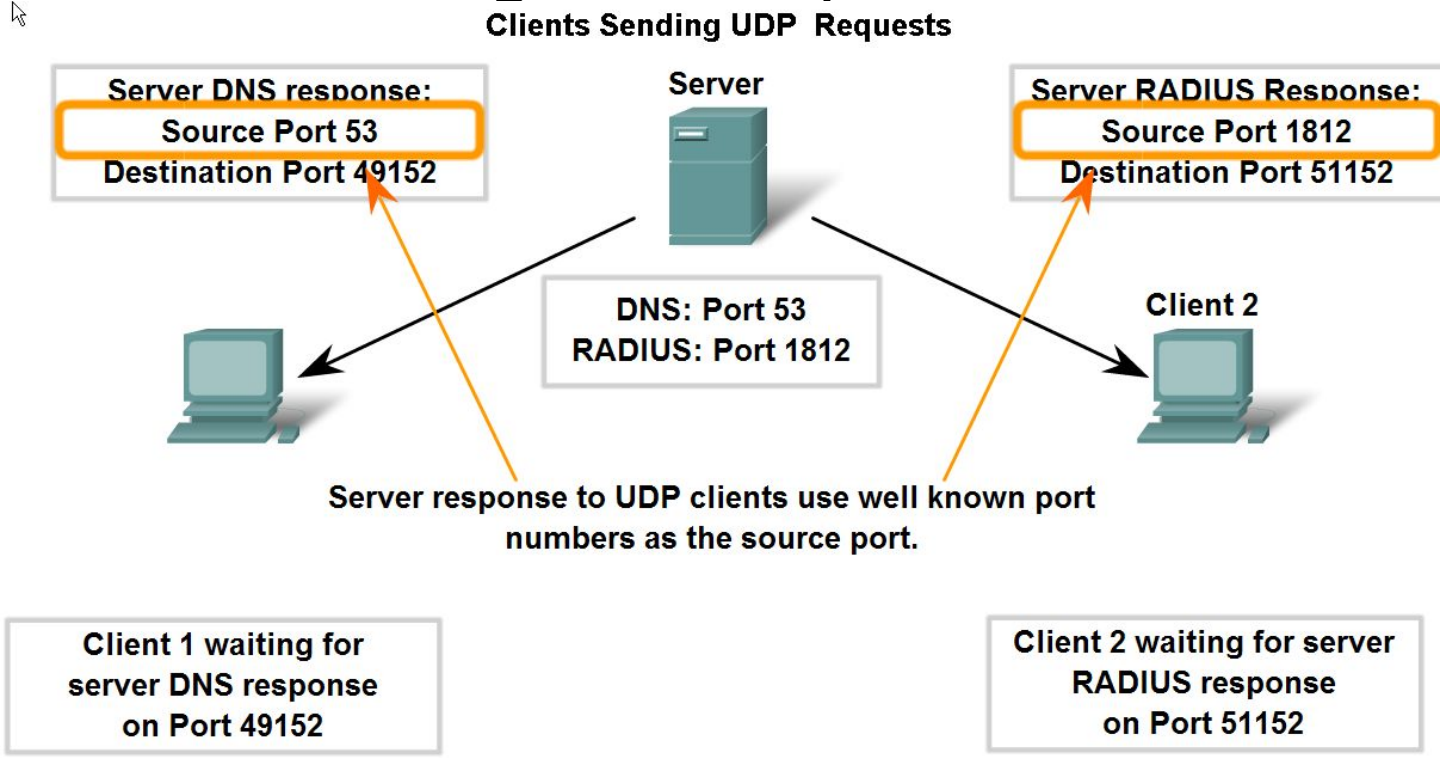
UDP

- No reassembly at receiving end.

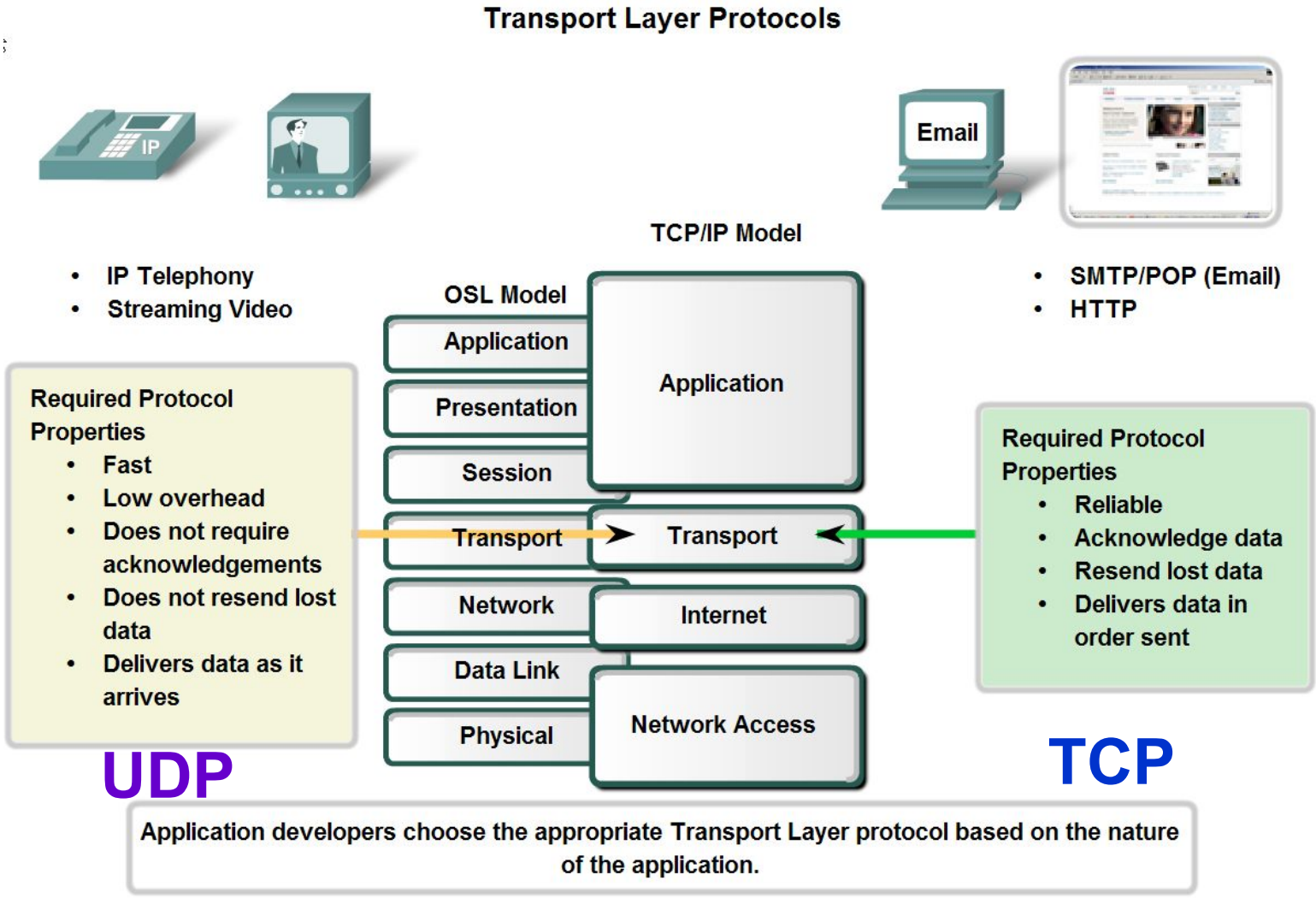


UDP Ports

- Like TCP-based applications, UDP-based server applications are assigned Well Known or Registered port numbers.



UDP versus TCP



TCP UDP

- Sets up a connection with the receiving host before sending data.
- Checks if segments have arrived and resends if they were lost. (Reliability)
- Sorts segments into the right order before reassembling the data.
- Sends at a speed to suit the receiving host. (Flow control)
- High overhead (20 bytes header)
- Used for Emails, Web Browsing etc.

- Connectionless. Does not contact receiving host before sending data.
- Does not check if data arrived and does not re-send. “Best effort”.
- Does not sort into the right order.
- No flow control.
- Low overhead (8 bytes header)
- Used for VoIP, streaming video, DNS, TFTP

The End