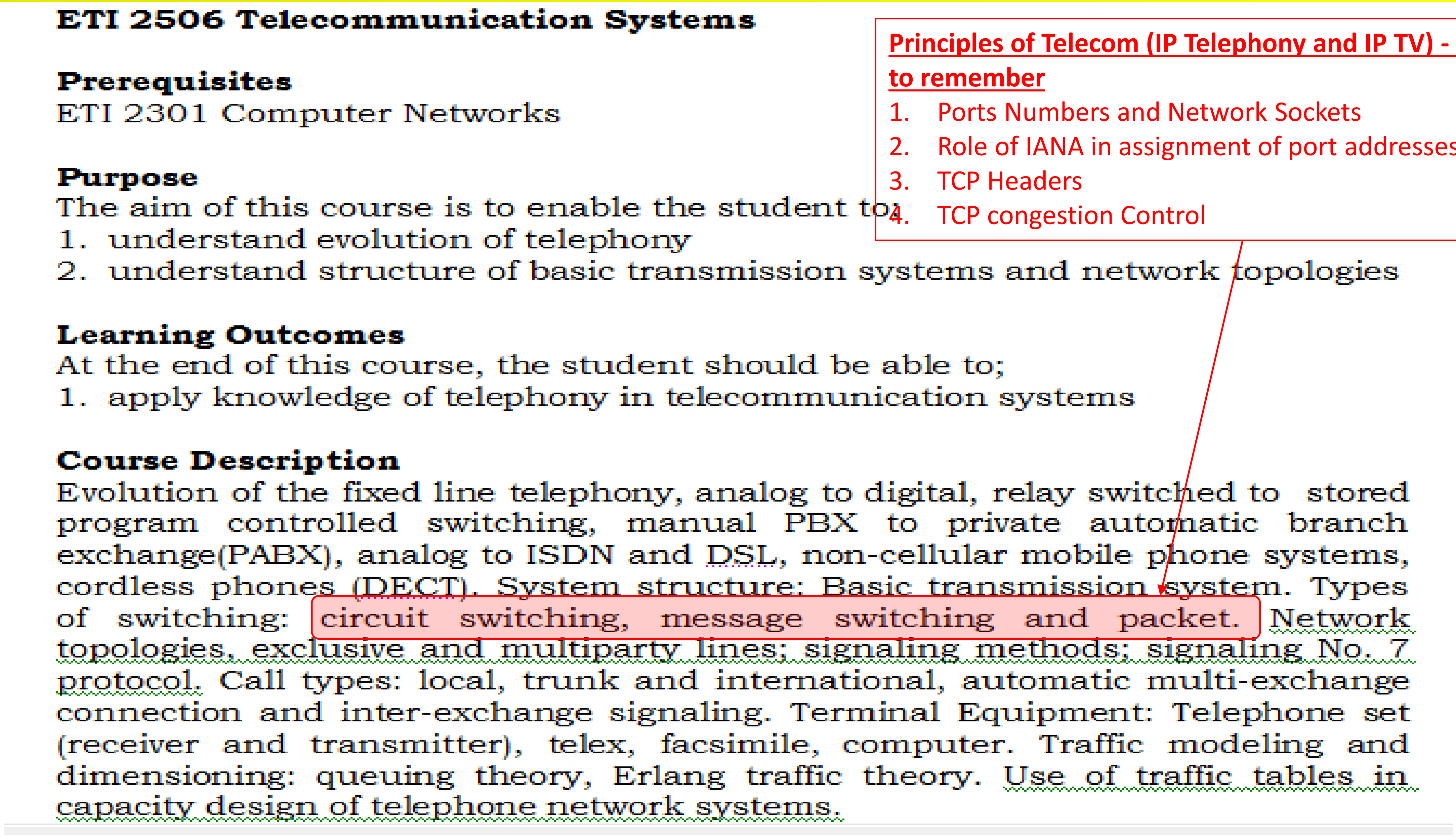
TRANSMISSION CONTROL PROTOCOL

ETI 2506 – TELECOMMUNICATION SYSTEMS

Monday, 7 November 2016

ETI 2506 - TELECOMMUNICATION SYLLABUS



ETI 2506 Telecommunication Systems

Prerequisites
ETI 2301 Computer Networks

Purpose
The aim of this course is to enable the student to;
1. understand evolution of telephony
2. understand structure of basic transmission systems and network topologies

Learning Outcomes
At the end of this course, the student should be able to;
1. apply knowledge of telephony in telecommunication systems

Course Description
Evolution of the fixed line telephony, analog to digital, relay switched to stored program controlled switching, manual PBX to private automatic branch exchange(PABX), analog to ISDN and DSL, non-cellular mobile phone systems, cordless phones (DECT). System structure: Basic transmission system. Types of switching: circuit switching, message switching and packet. Network topologies, exclusive and multiparty lines; signaling methods; signaling No. 7 protocol. Call types: local, trunk and international, automatic multi-exchange connection and inter-exchange signaling. Terminal Equipment: Telephone set (receiver and transmitter), telex, facsimile, computer. Traffic modeling and dimensioning: queuing theory, Erlang traffic theory. Use of traffic tables in capacity design of telephone network systems.

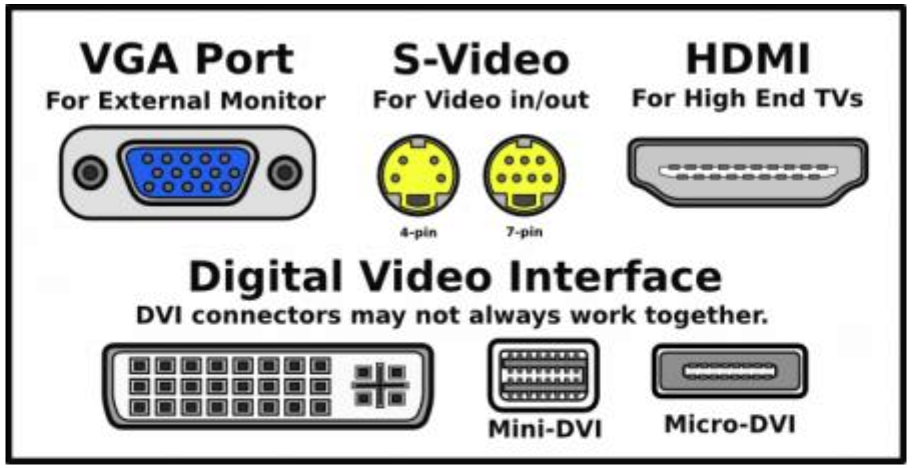
Principles of Telecom (IP Telephony and IP TV) - Key Issues to remember
1. Ports Numbers and Network Sockets
2. Role of IANA in assignment of port addresses
3. TCP Headers
4. TCP congestion Control

WHAT IS A PORT?

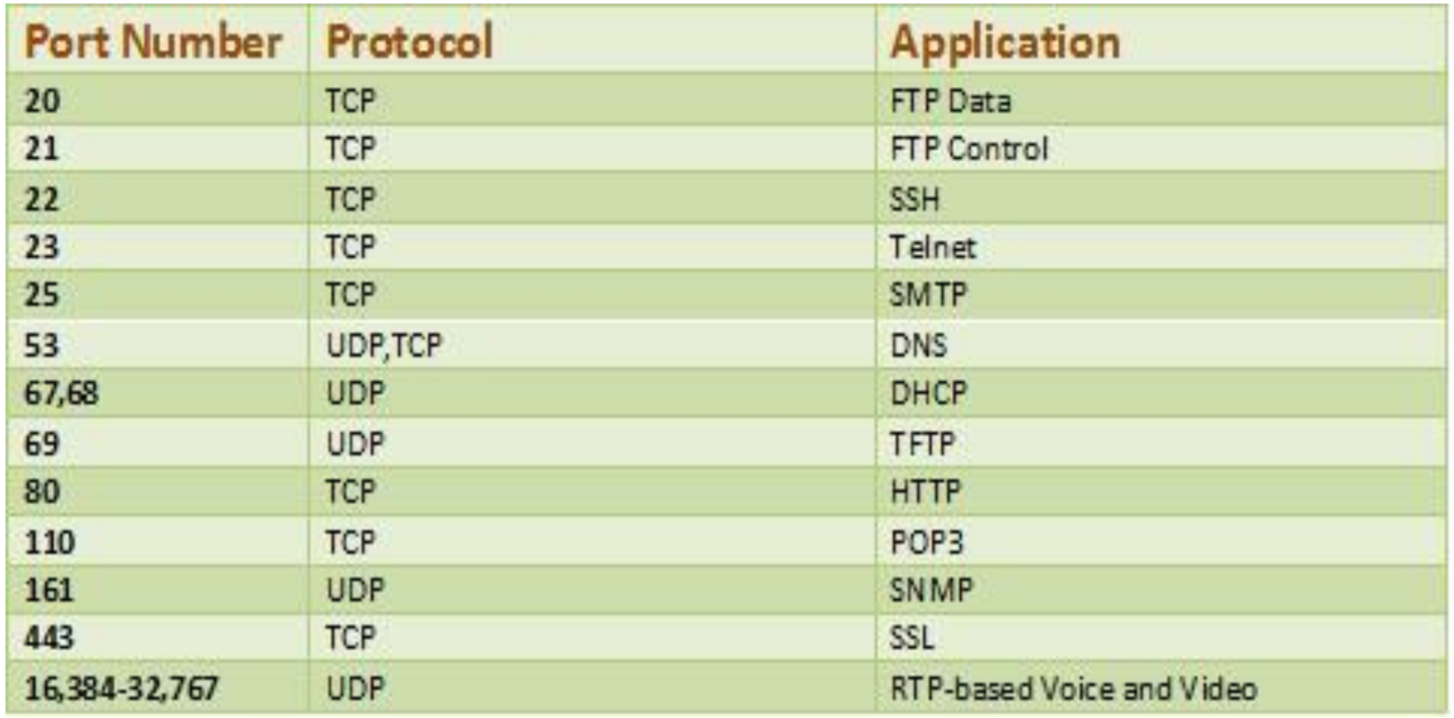
1. **A port is an endpoint of communication in an operating system.** While the term is also used for hardware devices, in software it is a logical construct that identifies a specific process or a type of network service.

2. **A port is always associated with an IP address of a host and the protocol type of the communication**, and thus completes the destination or origination address of a communication session.
3. 1024 well-known port numbers are reserved by convention to identify specific service types on a host.
4. Transport layer protocols, such as the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP) primarily use ports.

WHICH PORT- HARDWARE OR SOFTWARE?



(a) Hardware Ports



Port Number	Protocol	Application
20	TCP	FTP Data
21	TCP	FTP Control
22	TCP	SSH
23	TCP	Telnet
25	TCP	SMTP
53	UDP,TCP	DNS
67,68	UDP	DHCP
69	UDP	TFTP
80	TCP	HTTP
110	TCP	POP3
161	UDP	SNMP
443	TCP	SSL
16,384-32,767	UDP	RTP-based Voice and Video

(a) Software Ports

PORT NUMBERS

1. The port numbers are divided into three categories:
   a) Well-known ports
   b) Registered ports
   c) Dynamic or Private ports.
2. **Well-known ports** (also known as system ports) are those from 0 through 1023.
3. **Internet Assigned Numbers Authority (IANA)** is responsible for:
   a. Registration of port numbers.
   b. Global coordination of the DNS Root
   c. IP address Assignment
   d. Internet protocol resources

SOME WELL KNOWN PORTS

21: File Transfer Protocol (FTP)

22: Secure Shell (SSH)

23: Telnet remote login service

25: Simple Mail Transfer Protocol (SMTP)

53: Domain Name System (DNS) service

80: Hypertext Transfer Protocol (HTTP) used in the World Wide Web

110: Post Office Protocol (POP3)

119: Network News Transfer Protocol (NNTP)

123: Network Time Protocol (NTP)

143: Internet Message Access Protocol (IMAP)

161: Simple Network Management Protocol (SNMP)

194: Internet Relay Chat (IRC)

443: HTTP Secure (HTTPS)

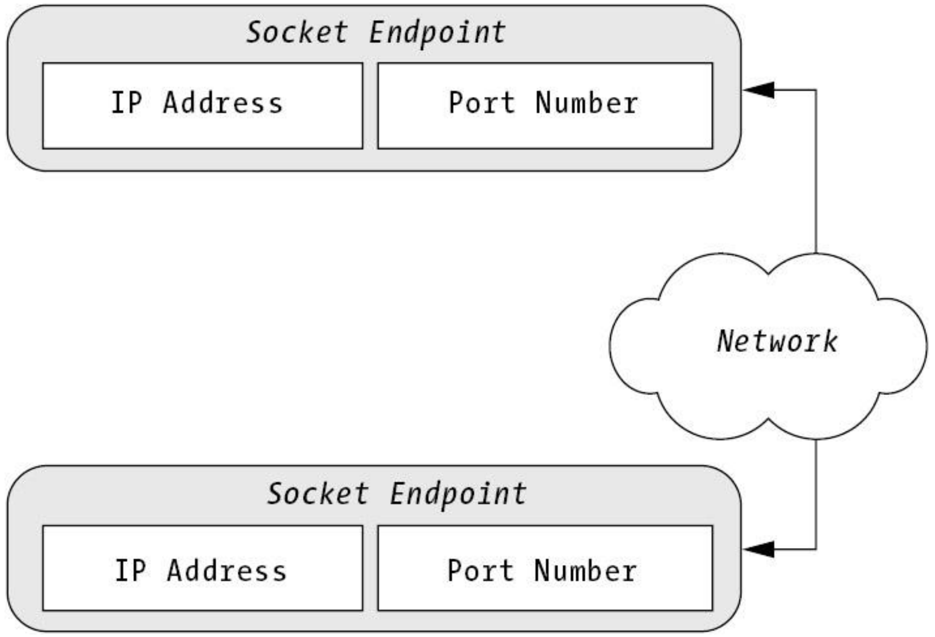
NETWORK SOCKET

1. **A network socket is one endpoint in a communication flow between two programs** usually over a network.

2. **Sockets are created and used with a set of programming requests or "function calls"** sometimes called the sockets application programming interface (API).

3. **. Sockets can also be used for communication between processes within the same computer**

4. **An Internet socket address** is the combination of an IP address and a port number.



SOCKET REQUESTS

SERVER SIDE

- socket()
  |
  bind()
  |
  recvfrom()
  |
  (wait for a sendto request from some client)
  |
  (process the sendto request)
  |
  sendto (in reply to the request from the client...for example, send an HTML file)

CLIENT SIDE

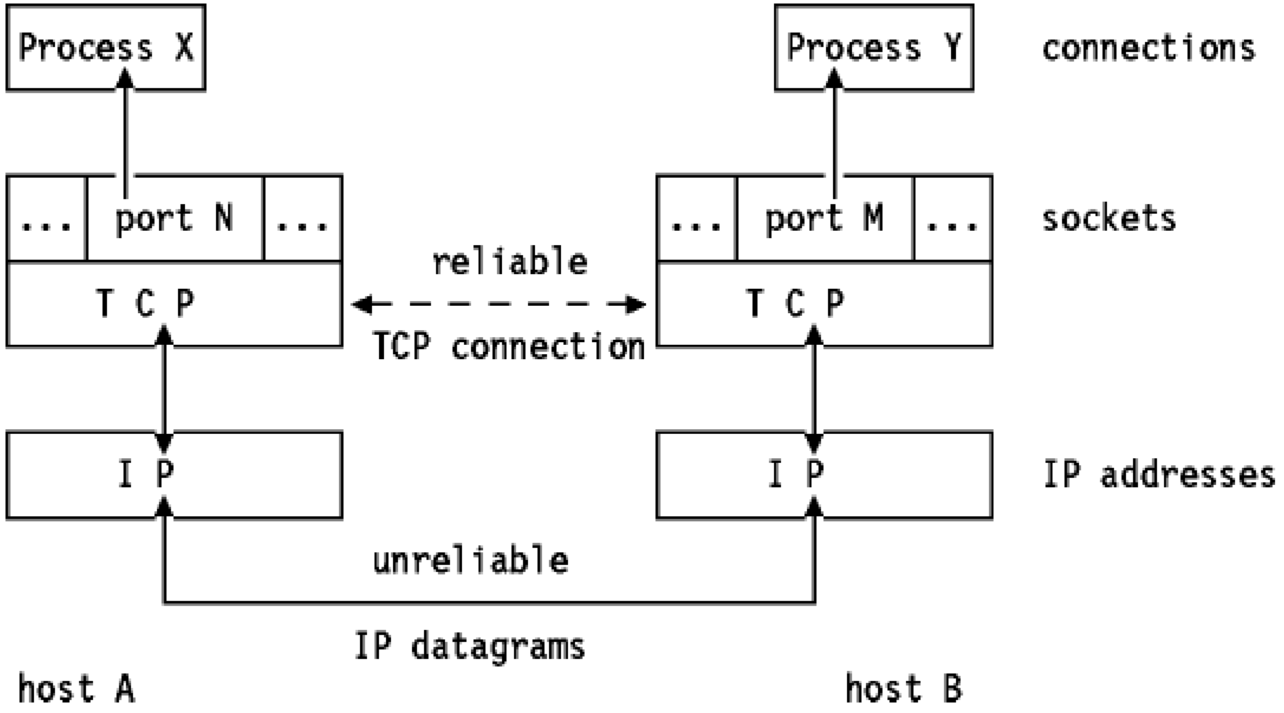
- socket()
  |
  bind()
  |
  sendto()
  |
  recvfrom()

WHAT IS TRANSMISSION CONTROL PROTOCOL (TCP)?

1. **TCP provides a connection oriented, reliable, byte stream service.**

2. The term connection-oriented means the two applications using TCP must establish a TCP connection with each other before they can exchange data.
3. **It is a full duplex protocol**, meaning that each TCP connection supports a pair of byte streams, one flowing in each direction.

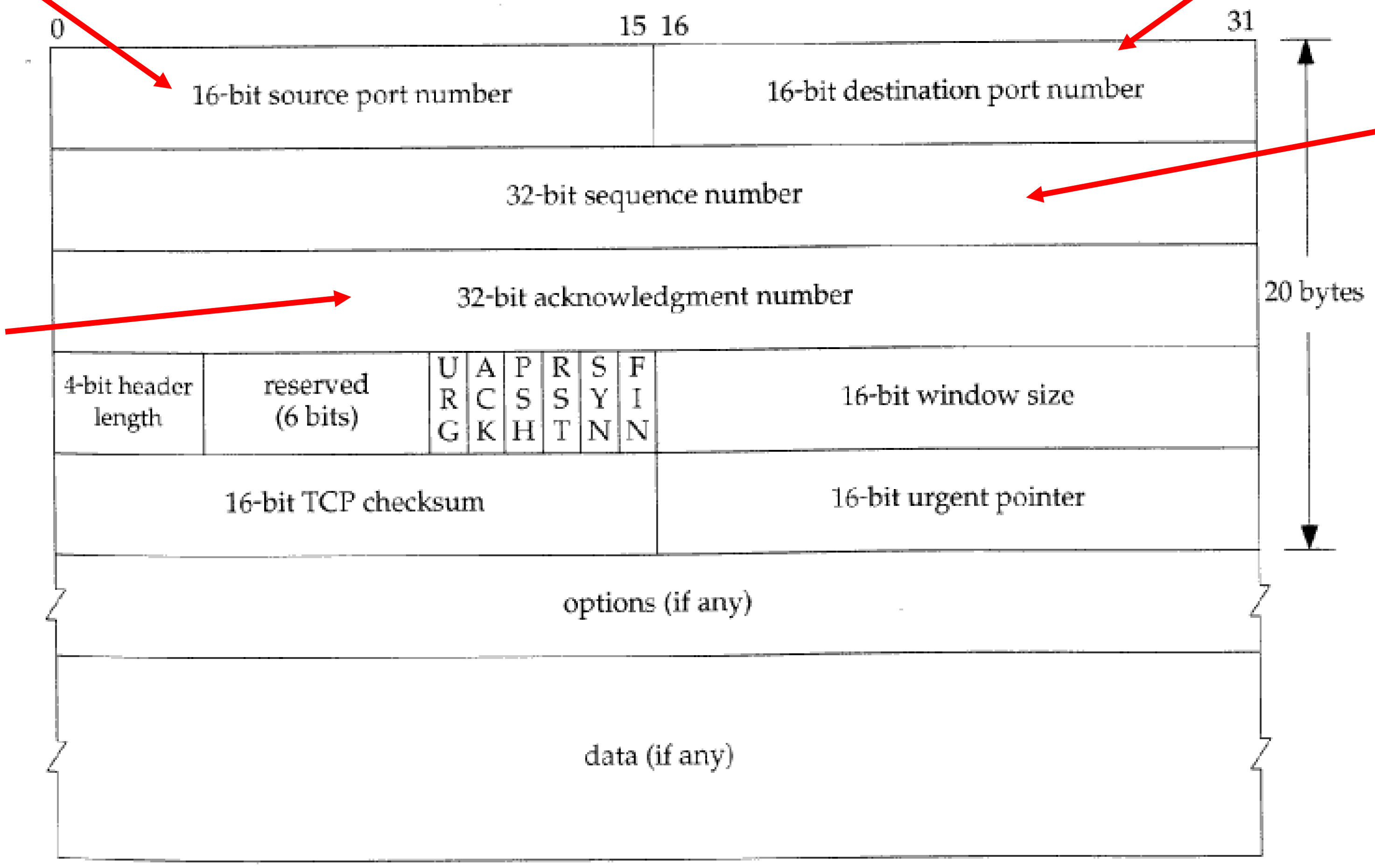
4. **TCP includes a flow-control mechanism** for each of these byte streams that allows the receiver to limit how much data the sender can transmit.



TCP HEADER

Source Port
Identifies the source

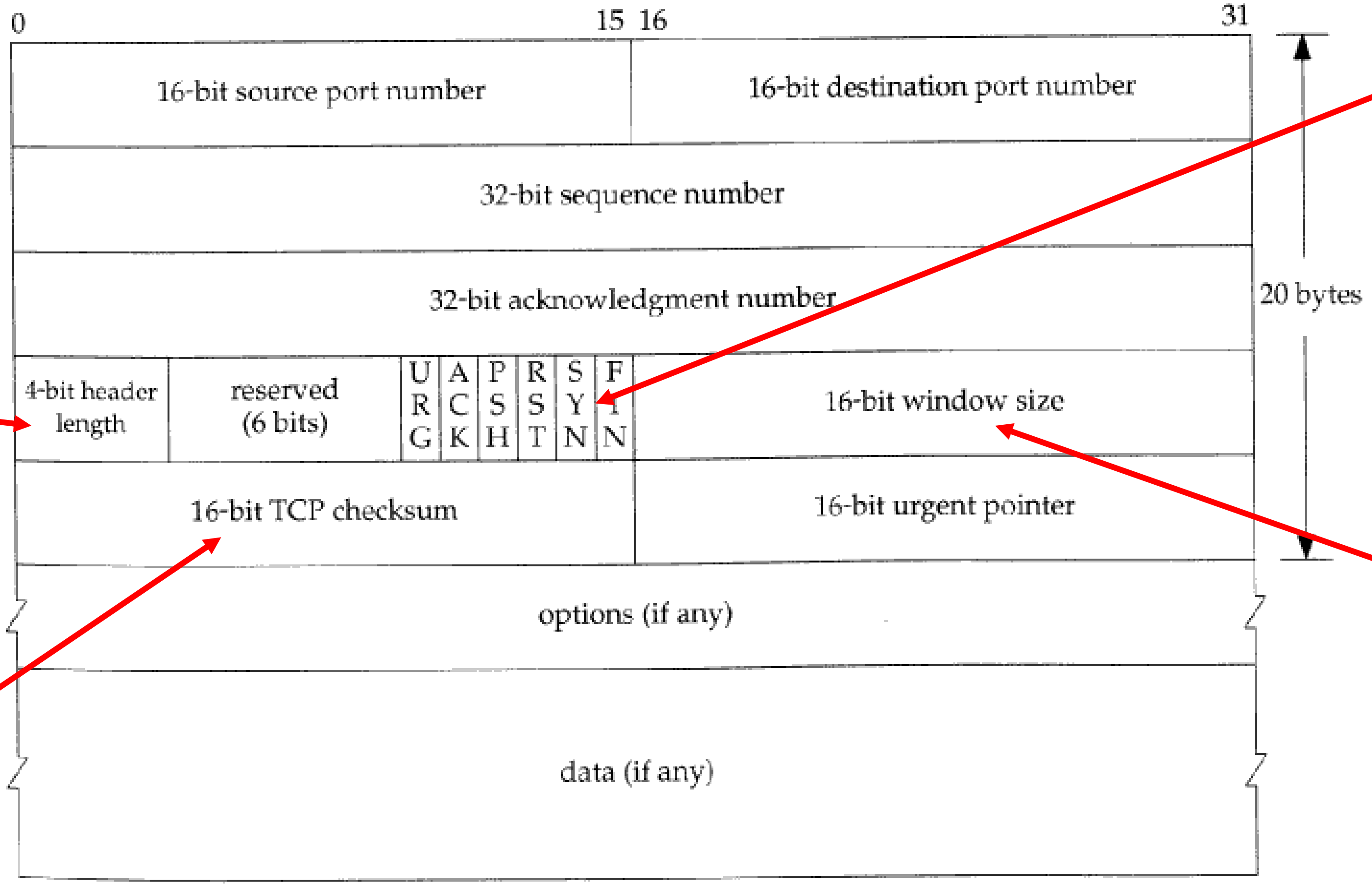
Destination Port
Identifies the destination



Sequence Number
Identifies the byte in the stream of data from the sending TCP to the receiving TCP that the first byte of data in this segment represents.

Acknowledgement number
Contains the next sequence number that the sender of the acknowledgement expects to receive.

TCP HEADER (2)



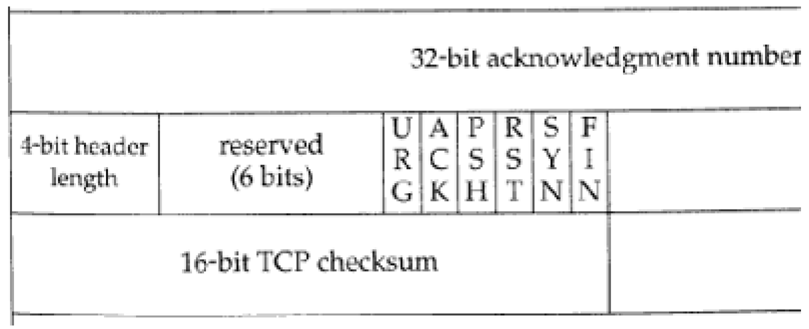
Flags field
Used to relay control information between TCP peers. The possible flags include SYN, FIN, RESET, PUSH, URG, and ACK.

Header length
The length of the header. Required because the length of the options field is variable

Window size
Used to advertises a sliding window size to the sender

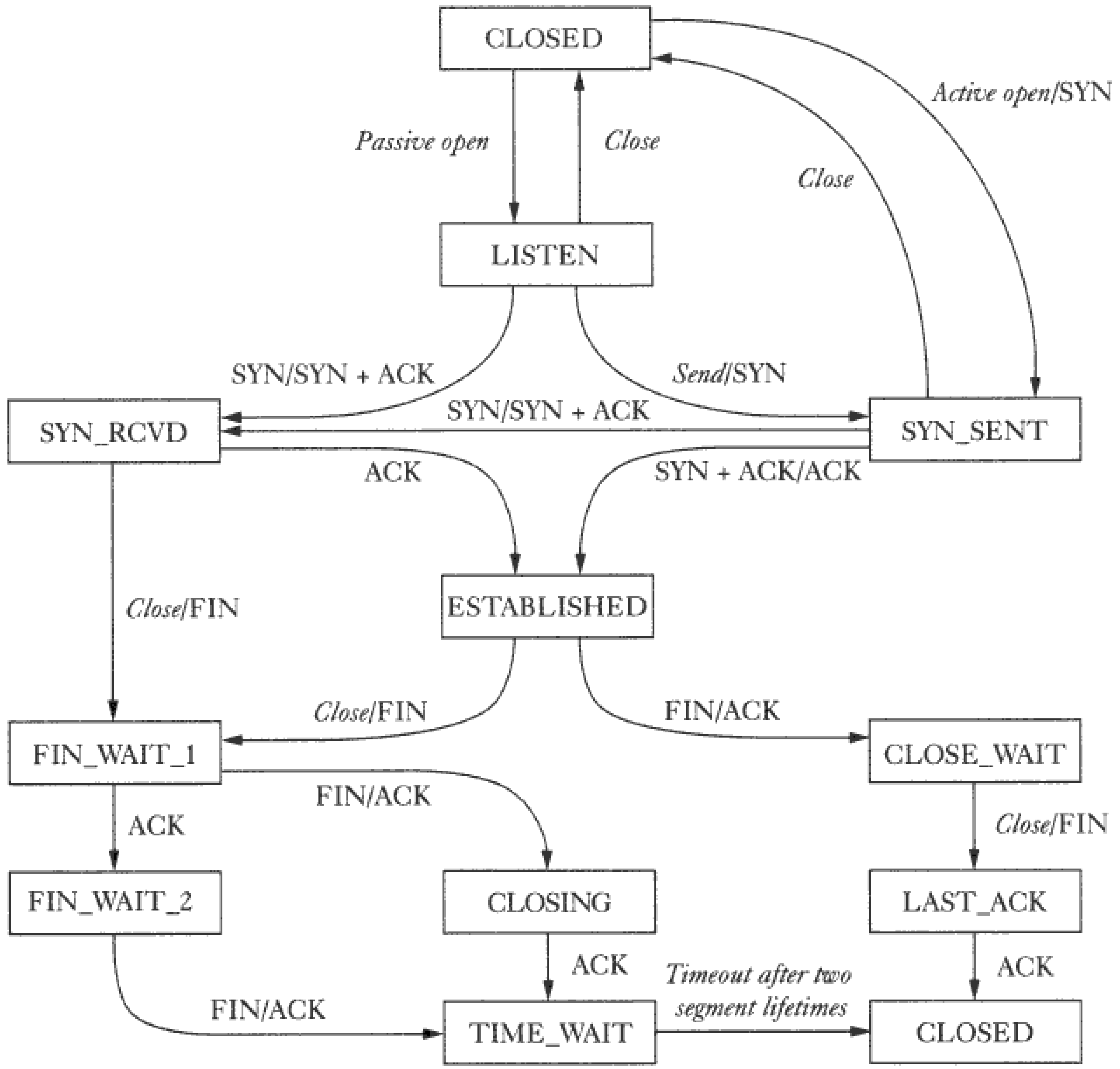
TCP Checksum
Computed on TCP segment, TCP header and the TCP data

DETAILS OF THE FLAG FIELD

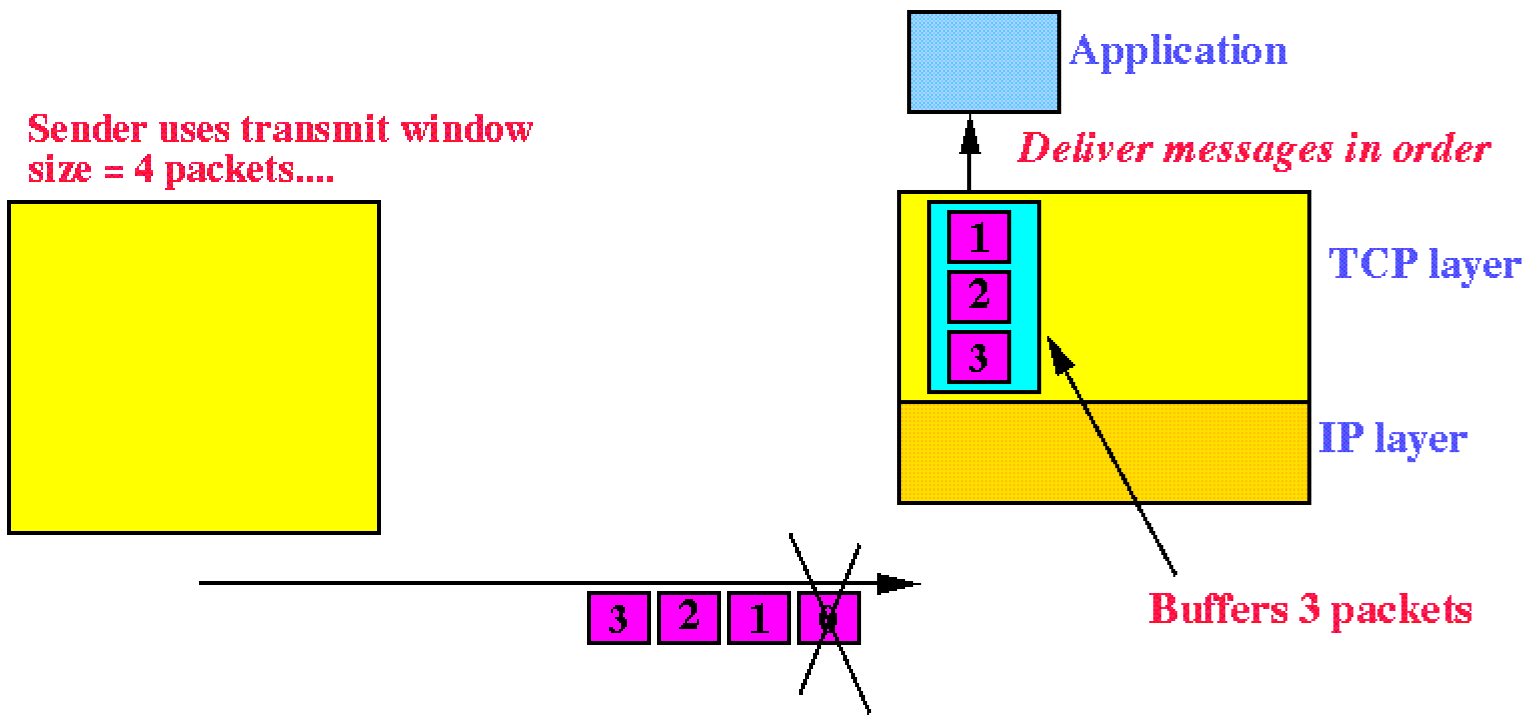


1. **SYN and FIN** used when establishing and terminating a TCP connection, respectively.
2. **ACK** set any time the Acknowledgement field is valid, implying that the receiver should pay attention to it.
3. **URG** signifies that this segment contains urgent data.
4. **PUSH** signifies that the sender invoked the push operation.
5. **RESET** signifies that the receiver has become confused and so wants to abort the connection

TCP STATE TRANSITION DIAGRAM



PRINCIPLE OF CONGESTION CONTROL AT TCP LAYER

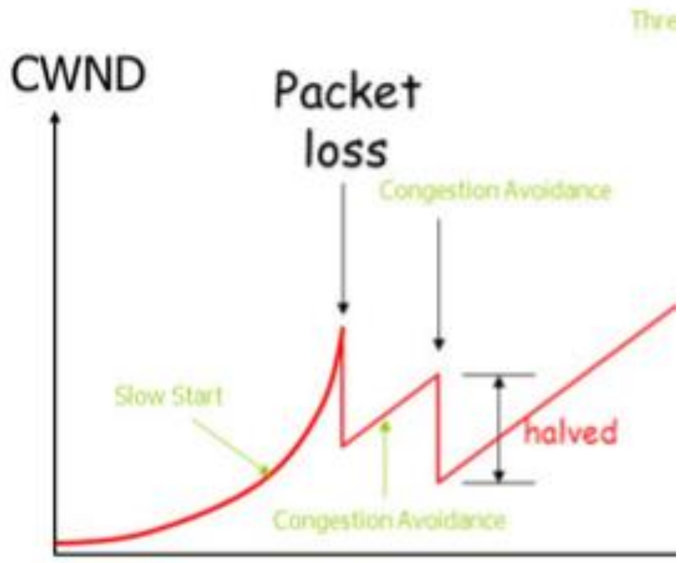


Sender uses transmit window size = 4 packets....
Application
Deliver messages in order
1
2
3
TCP layer
IP layer
3
2
1
0
Buffers 3 packets

Receiver's buffer is now FULL
It will DROP any NEW packets from the sender....
Receiver WILL NEVER receive packet 0 !!!
DEAD LOCK !!!

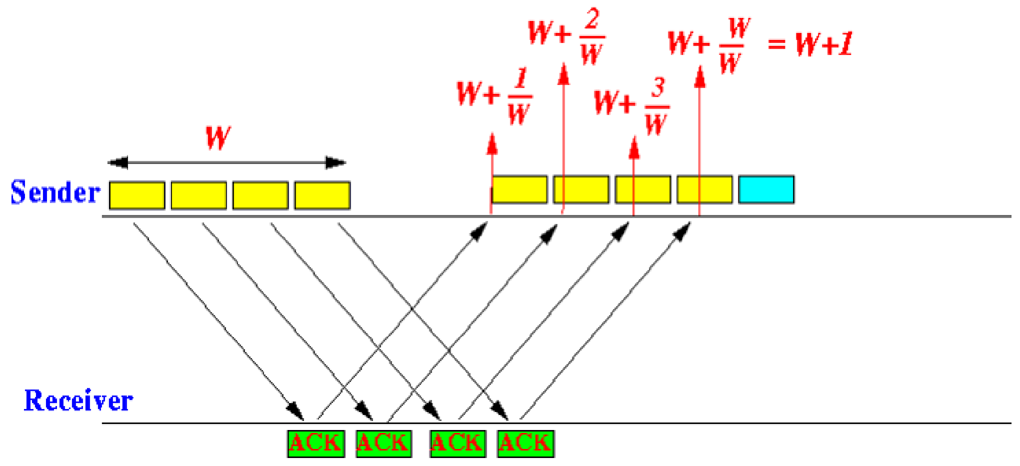
CONGESTION CONTROL – SLOW START (1)

1. **Slow start adds a congestion window**, called "cwnd".
2. **When a new connection is established the congestion window is initialized** to one segment (i.e., the segment size announced by the other end, or the default, typically 536 or 512).
3. **Each time an ACK is received, the congestion window is increased by one segment.** The sender can transmit up to the minimum of the congestion window and the advertised window.
4. **Congestion window is flow control imposed by the sender** based on it's assessment of perceived network congestion.
5. **Advertised window is flow control imposed by the receiver based** on amount of available buffer space.



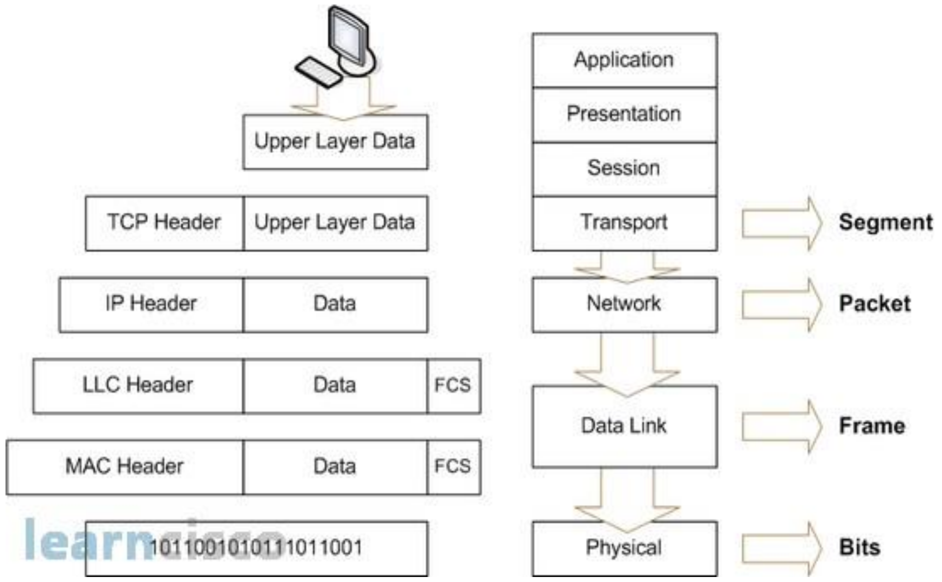
CONGESTION CONTROL – SLOW START (2)

5. **The sender starts by transmitting one segment and waiting for its ACK**. When that ACK is received, the congestion window is incremented from one to two, and two segments can be sent.
6. **When each of those two segments is acknowledged, the congestion window is increased**. This provides an exponential growth, although it is not exactly exponential because the receiver may delay its ACKs, typically sending one ACK for every two segments that it receives.
7. **At some point the capacity of the internet will be reached**, and an intermediate router will start discarding packets.

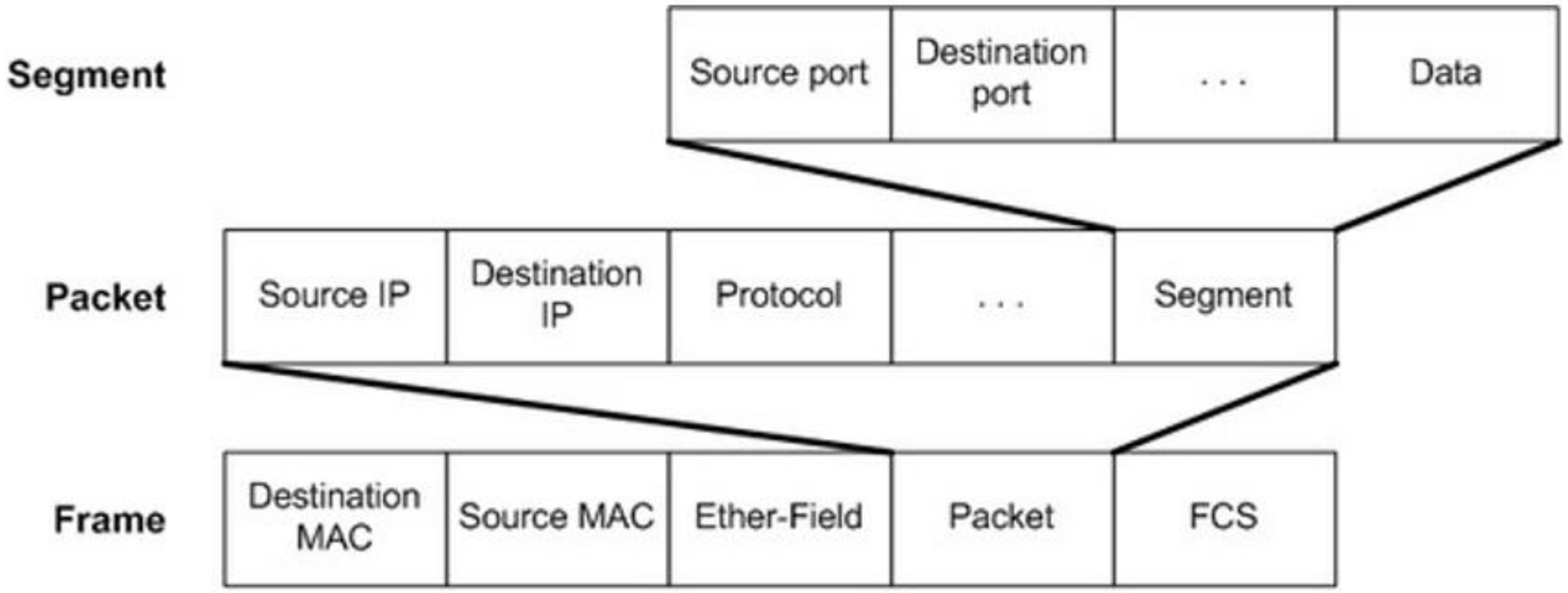


DATA ENCAPSULATION (1)

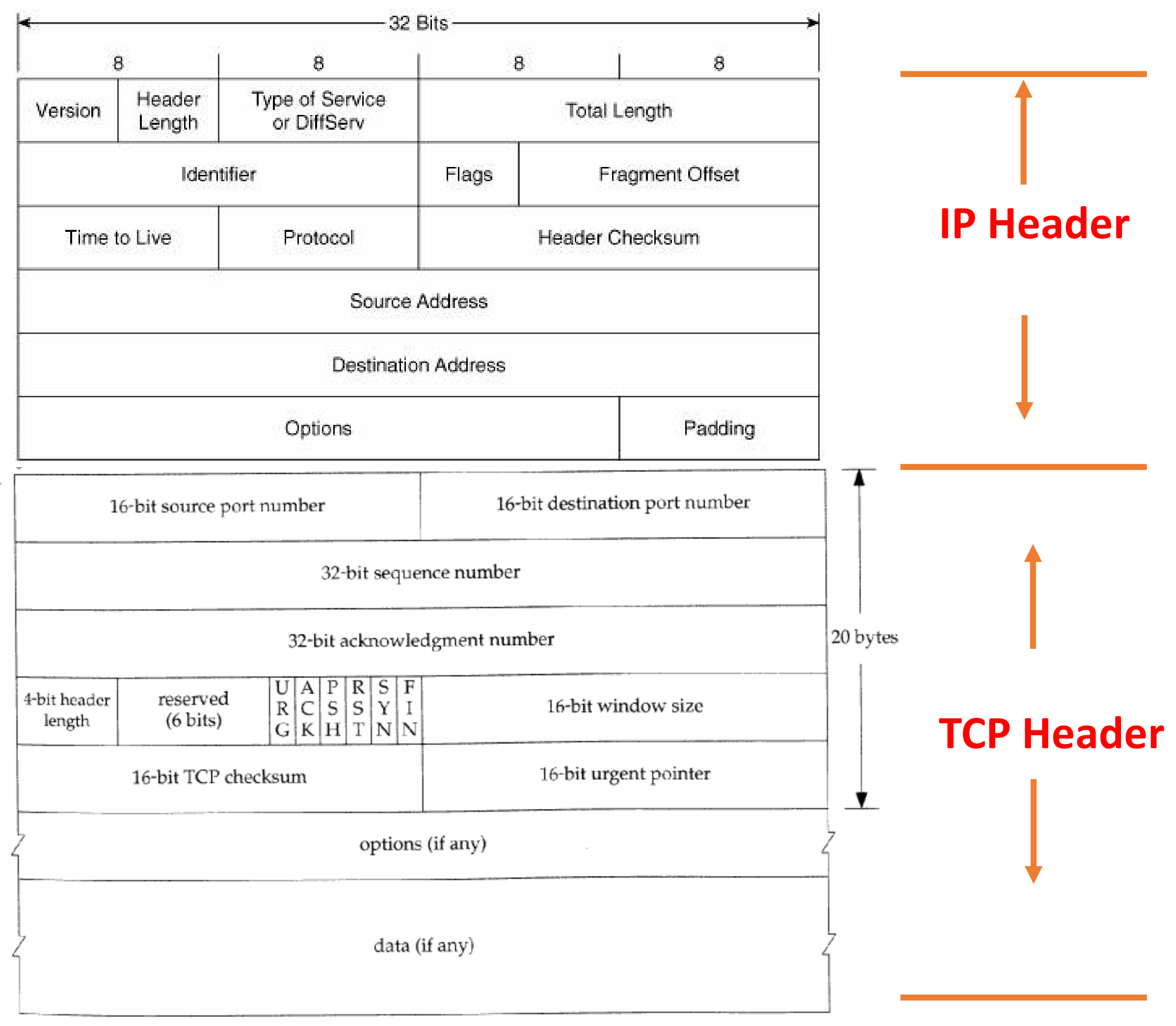
1. When a host transmits data across a network to another device, the data goes through encapsulation as follows:
   a) **Data is wrapped with protocol information at each layer of the OSI model.**
   b) Each layer communicates only with its peer layer on the receiving device.
   c) **To communicate and exchange information, each layer uses Protocol Data Units (PDUs).**
   d) PDUs hold the control information attached to the data at each layer of the model.
   e) **PDUs are usually attached to the header in front of the data field but can also be in the trailer, or end, of it.**
   f) PDU information is read only by the peer layer on the receiving device.



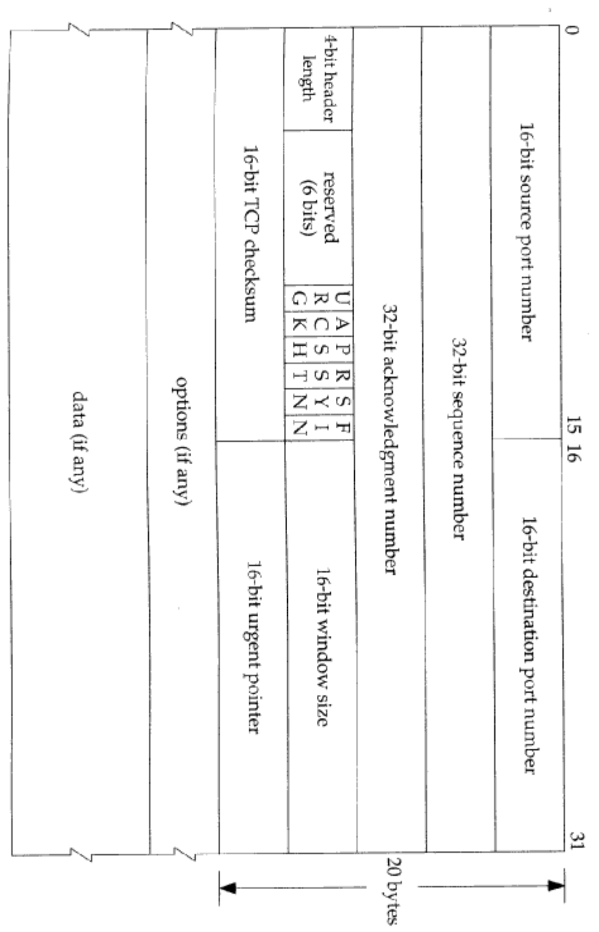
DATA ENCAPSULATION (2)



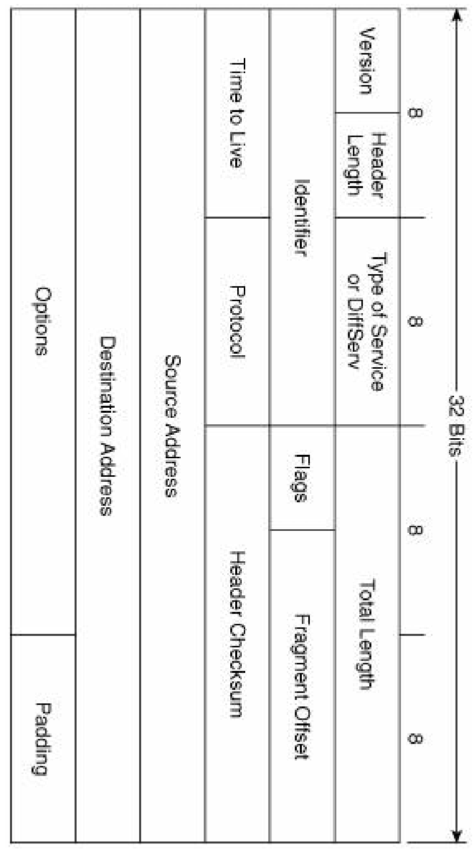
DATA ENCAPSULATION – TCP/IP



TCP Payload



IP Header



http://www.learncisco.net/cisco-practice-tests.html