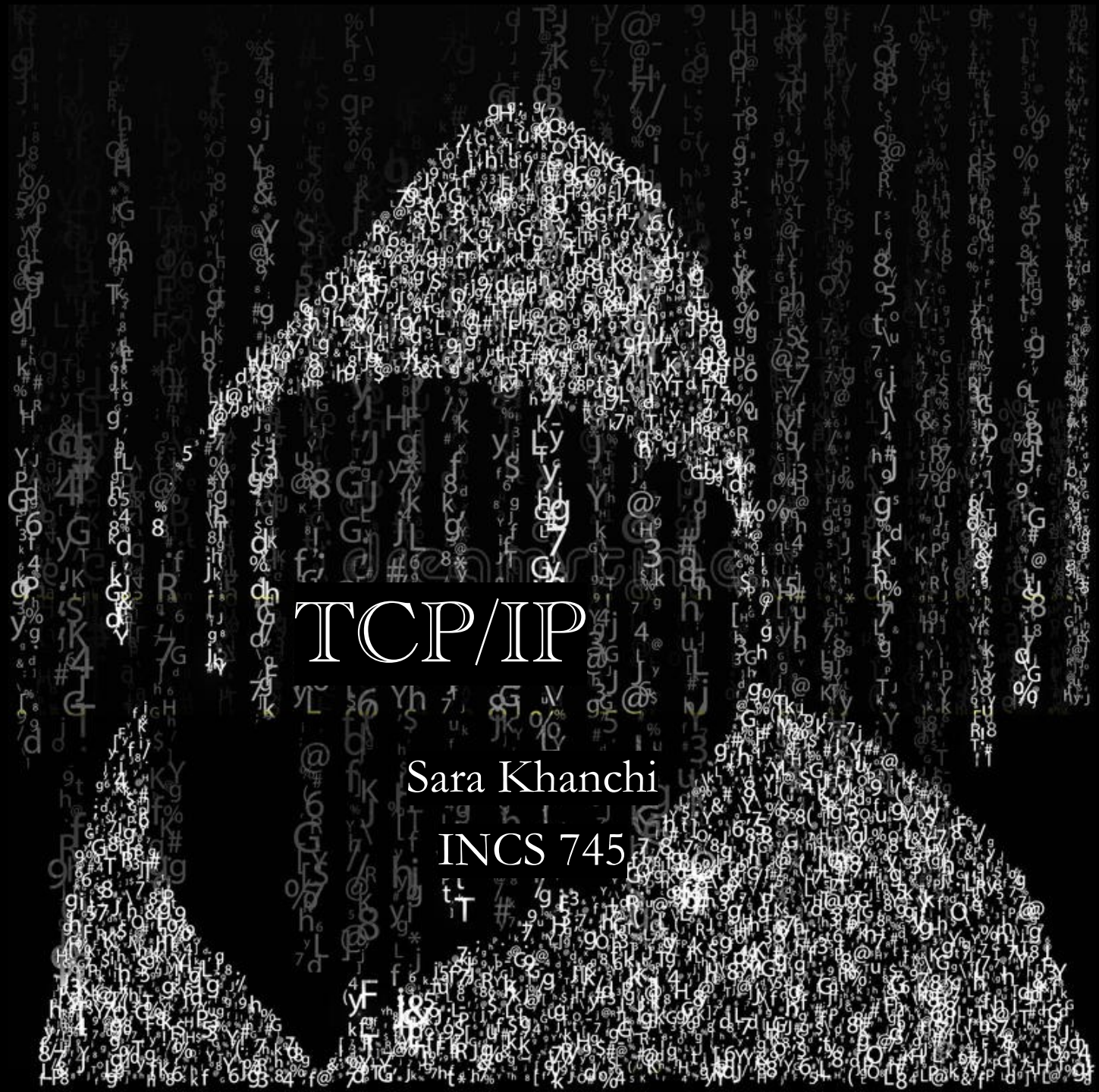


TCP/IP

Sara Khanchi

INCS 745

Outline

- TCP/IP model
- Data Encapsulation
- Internet Protocol (IP)
- Transmission Control Protocol (TCP)

TCP/IP

- Transmission Control Protocol/Internet Protocol (TCP/IP)
  - Suite of protocols that underlie the Internet
  - Comprises many protocols and applications
  - Common language of networked computers
  - Makes transferring information fast and efficient
- IP has tools to correctly rout packets
- TCP is responsible for safe and reliable data transfer between host computers

TCP/IP

OSI Model Layers	TCP/IP Layers	Core Protocols
1 Physical layer	Link layer	ARP, Ethernet, DSL, ISDN, FDDI, L2TP, NDP, OSPF, PPP, RARP
2 Data Link layer		
3 Network layer	Internet layer	ECN, ICMP, ICMPv6, IGMP, IPSec, IPv4, IPv6
4 Transport layer	Transport layer	DCCP, RSVP, SCTP, TCP, UDP
5 Session layer	Application layer	BGP, DHCP, DHCPv6, DNS, FTP, HTTP, IMAP, IRC, LDAP, MGCP, NNTP, NTP, POP, RIP, RPC, RTP, RTSP, SIP, SMTP, SNMP, SOCKS, SSH, Telnet, TLS/SSL, XMPP
6 Application layer		
7 Presentation layer		

Figure 5-1 TCP/IP

Data Encapsulation

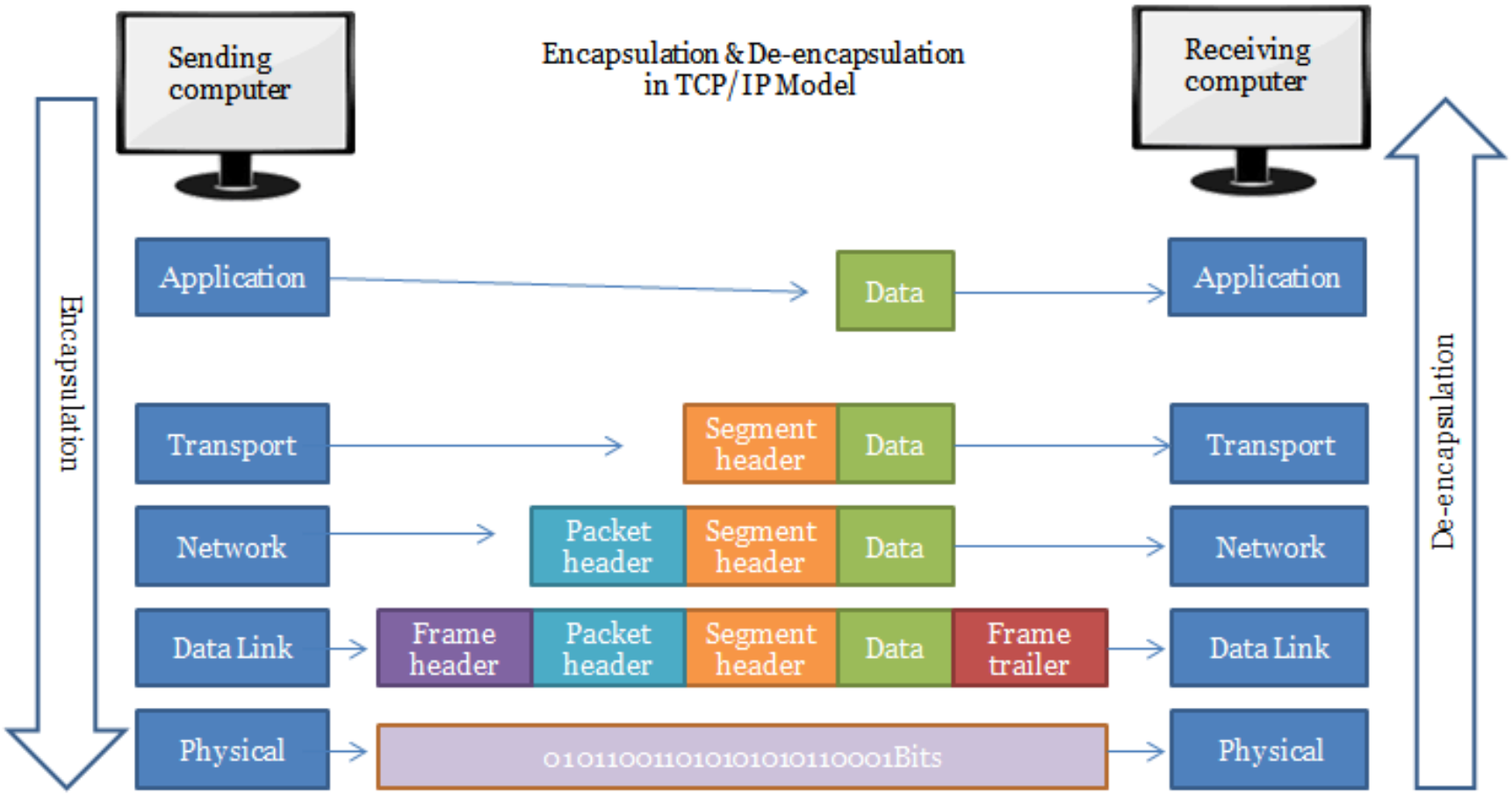
- Data encapsulation
  - Enclosing higher-level protocol information in lower-level protocol information
  - Also called data hiding
  - Implementation details of a class are hidden from user

Data Encapsulation

↑	Application layer	↓
↑	Transport layer	↓
↑	Internet layer	↓
←	Link layer	←

Figure 5-2 TCP/IP data encapsulation example

Data Encapsulation



https://embeddedgeeks.com/encapsulation/

IP (Internet Protocol)

- Internet Protocol (IP)
  - Transmits data from source to final destination
  - Provides unique addressing to find systems regardless of their underlying network specifications
  - Network protocol operating at layer 3 of the OSI Model
  - And layer 2 or 3 of the TCP/IP Model
- IP is connectionless
  - No guarantee of delivery of packets to the destination

IP (Internet Protocol)

- IP protocols
  - IPv4 (32-bit address)
    - Usually written as a dotted-decimal, e.g., 192.168.100
  - IPv6 (128-bit address)
    - Usually written as eight groups of four hex digits, e.g., 2001:0db8:85a3:08d3:1319:8a2e:0370:7334

- IPv4 limitations leads to IPv6
  - IPv6 has more address space

IP (Internet Protocol)

- IP packets often arrive out of sequence
  - Vulnerability that attackers can exploit
- IP fragmentation
  - When the IP packet size is more than the frame size the Data Link can provide
  - IP packet is divided into smaller size chunk of data called fragment

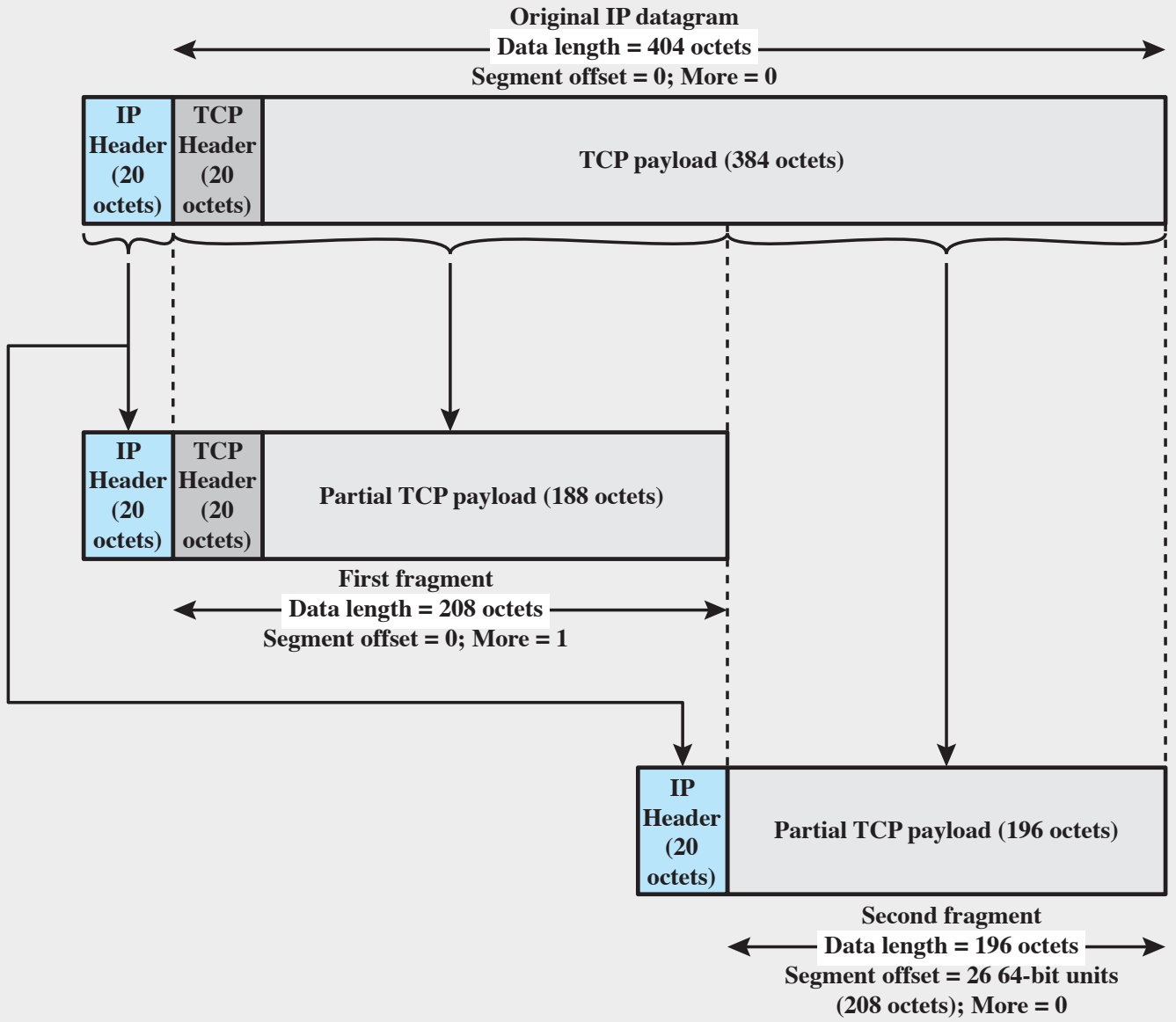
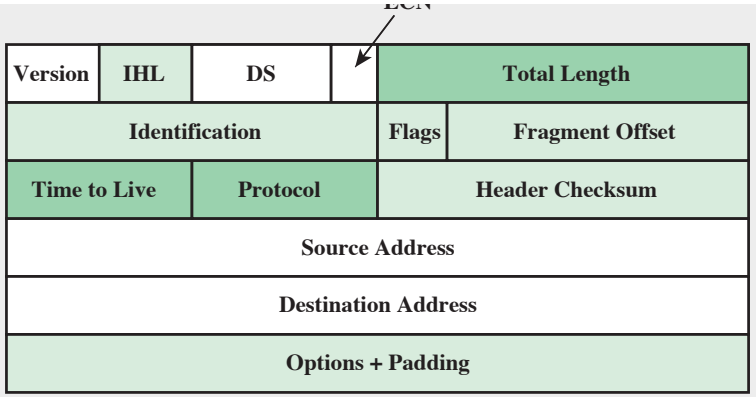
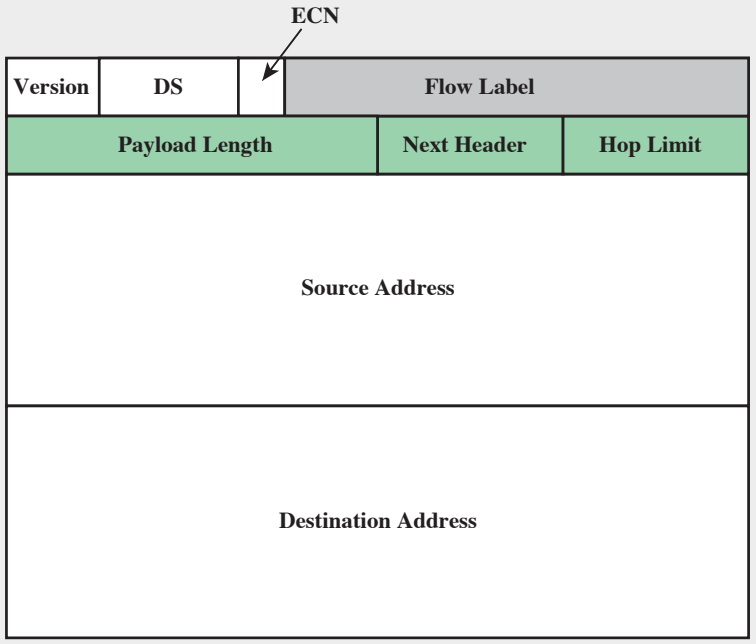


Figure 14.4 Fragmentation Example



(a) IPv4 header



(b) IPv6 header

Figure 14.5 IPv4 and IPv6 Headers

IP Header

Table 5-2 Components of the IP header

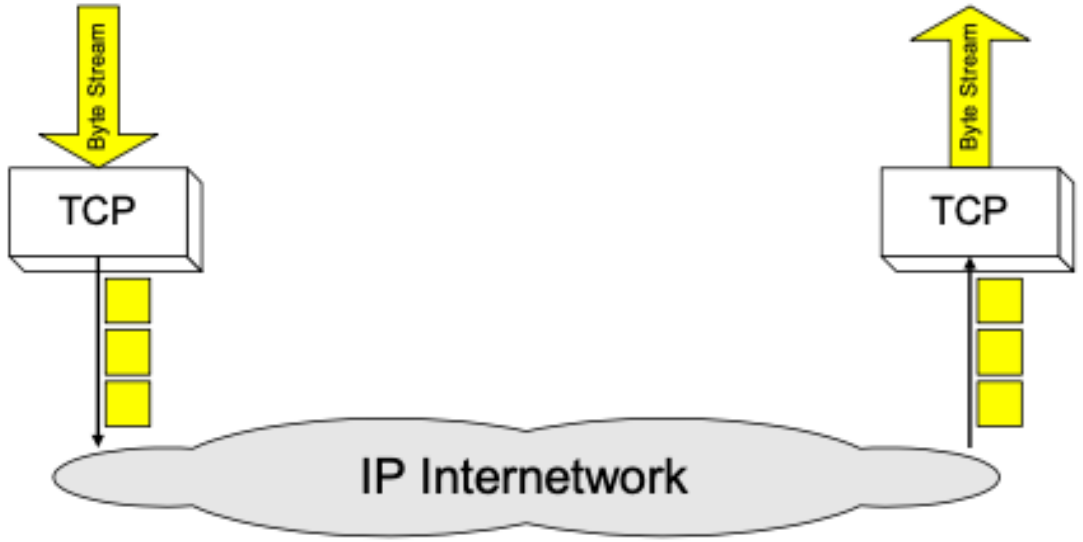
Fields within an IP Header	Size in Bits	Description
Version	4	Set to 4 to specify IPv4 packets or to 6 to specify IPv6 packets.
IHL	4	Internet Header Length (IHL) is the length of the Internet header; normal value=5.
Type of Service	8	Points out the quality of the requested service; the quality of service specifies trade-offs between low delay, high reliability, and high throughput, and specifies the precedence level of a packet.

Table 5-2 Components of the IP header (continued)

Fields within an IP Header	Size in Bits	Description
Total Length	16	Specifies the total length of the datagram packet that is being relayed; because this is a 16-bit field, the maximum size of the IP datagram is 65,535 bytes.
Identification	16	Specifies an identity assigned to all of the packets in a datagram so that if the datagram is fragmented, it can be reassembled at the destination.
Flags	3	Three flags are used in the fragmentation of the datagram. The first is reserves. The second is Don't Fragment (DF), which is used for testing purposes, and is ignored by most higher-level protocols. Setting DF to 1 means the packet should not be fragmented. The third flag is More Fragments (MF). If MF is set to 0, it is the last piece. If set to 1, it will be followed by others.
Fragment Offset	13	Specifies the sequence of the fragment in the datagram.
Time to Live	8	Specifies the maximum time the datagram can remain on the network, stated in terms of how many router hops it can make.
Protocol	8	Specifies the next-level protocol to which the datagram belongs.
Header Checksum	16	Specifies the checksum on only the datagram header. It is a simple 16-bit checksum calculated by dividing the header bytes into words (a word is two bytes) and then adding them together. It is needed because the header changes when the value in the Time to Live field changes. At that point, it is recalculated. Each router performs this checksum, and if the calculated figure and the contents of the Header Checksum field are found to be dissimilar, then that router discards the packet as corrupted.
Source Address	32	Specifies the source IP address of the datagram; this is always the original sender, even if the intervening sender was a router or bridge.
Destination Address	32	Specifies the destination IP address of the datagram; this is always the final destination of the packet, even if the intervening destination is a router or a bridge.
Options	Varies	May or may not be present in a datagram; these options are related to routing packets over the network.
Padding	Varies	Adds extra "0" characters to pad out the header to a multiple of 32 bits.

TCP

- **TCP:** Transmission Control Protocol
- Connection-oriented protocol
- Provides a reliable unicast end-to-end byte stream over an unreliable internetwork.



TCP Connection

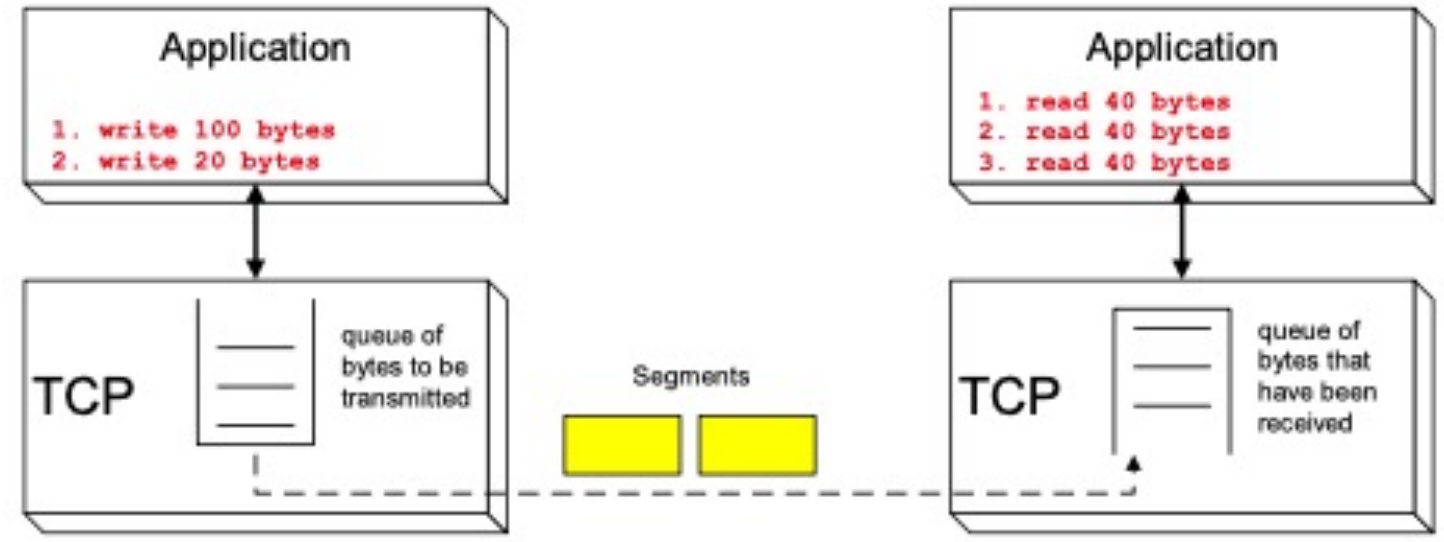
- Needs to create a logical connection before any data transfer
  - Mostly work in Server-Client mode
    - Server is listening for connection request (passive)
    - Client is the one who initiates the connection (active)
- Three-way handshaking to create the connection
- Connection is full duplex
- There similar process is done for closing the connection

Reliable Communication

- Protocol Data Unit in TCP is called Segment
- For successful delivery of each segment, an acknowledgment (ACK) is sent back
- TCP maintains a timer. If an ACK is not received in time, the segment is retransmitted
- TCP assign sequence number to each segment
  - Sequence numbers are increased based on the size of the segment

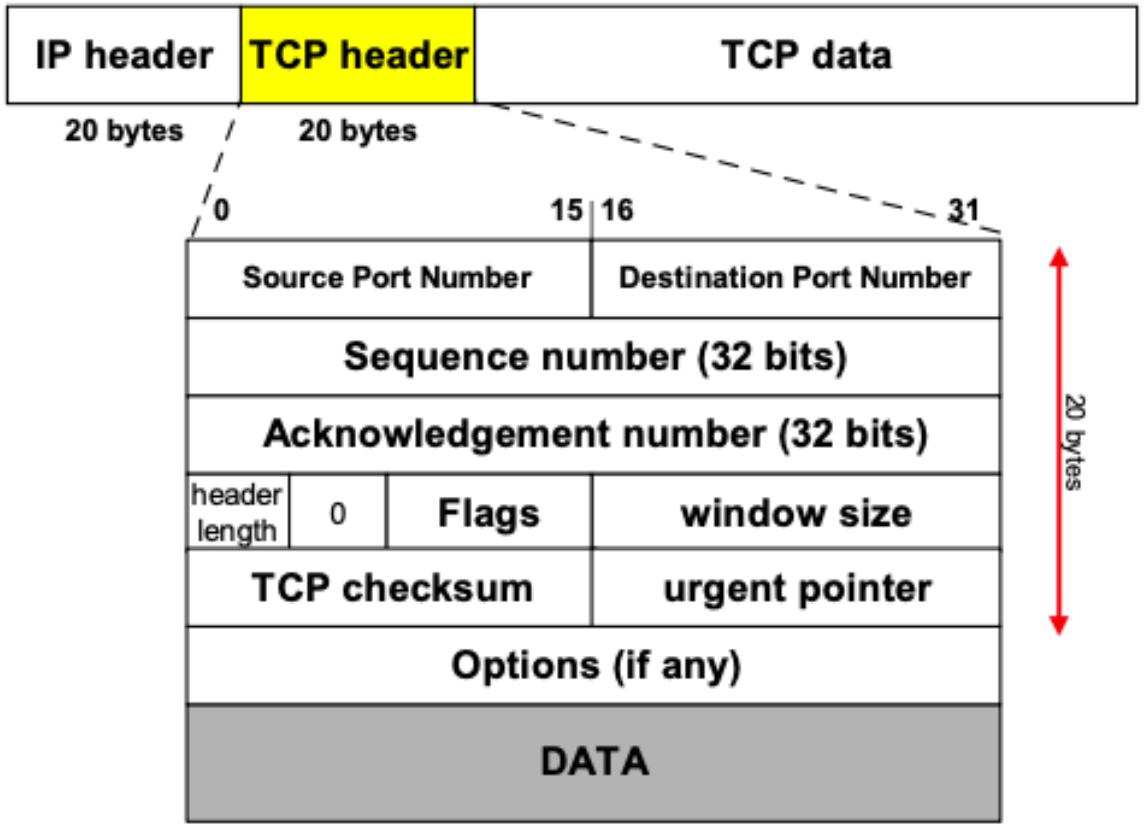
Byte Stream Service

- To the lower layers, TCP handles data in blocks, the segments
- To the higher layers TCP handles data as a sequence of bytes and does not identify boundaries between bytes
- So: Higher layers do not know about the beginning and end of segments!



TCP Segment

- TCP segments have a 20 byte header with >= 0 bytes of data.



TCP Segment

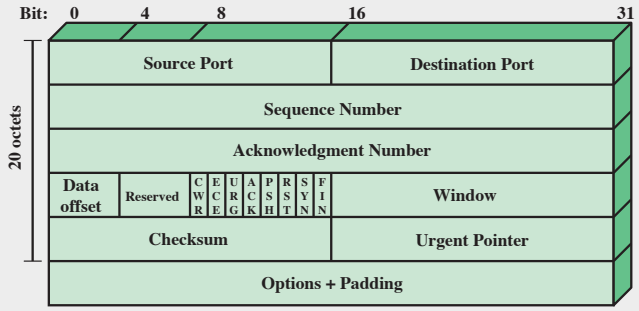


Figure 15.10 TCP Header

TCP Segment

Table 5-3 Components of the TCP header

Fields in a TCP Header	Size in Bits	Description
Source port number	16	Can be any port number from 1 to 65,535; there is no reason for a source port for a Web-page request to be port 80.
Destination port number	16	The port number of the destination; can be any port number from 1 to 65,535.
Sequence number (ISN)	32	Sequence number of the first data octet in this segment.
Acknowledgment number	32	If the ACK bit is set, this field contains the value of the next sequence number that is expected by the sender.
Header	4	Offset where data begins in the packet.
Reserved	6	Reserved for future use.
URG	1	Urgent pointer field.
ACK	1	Acknowledgment field.
PSH	1	Push Function.
RST	1	Reset the connection.

TCP Segment

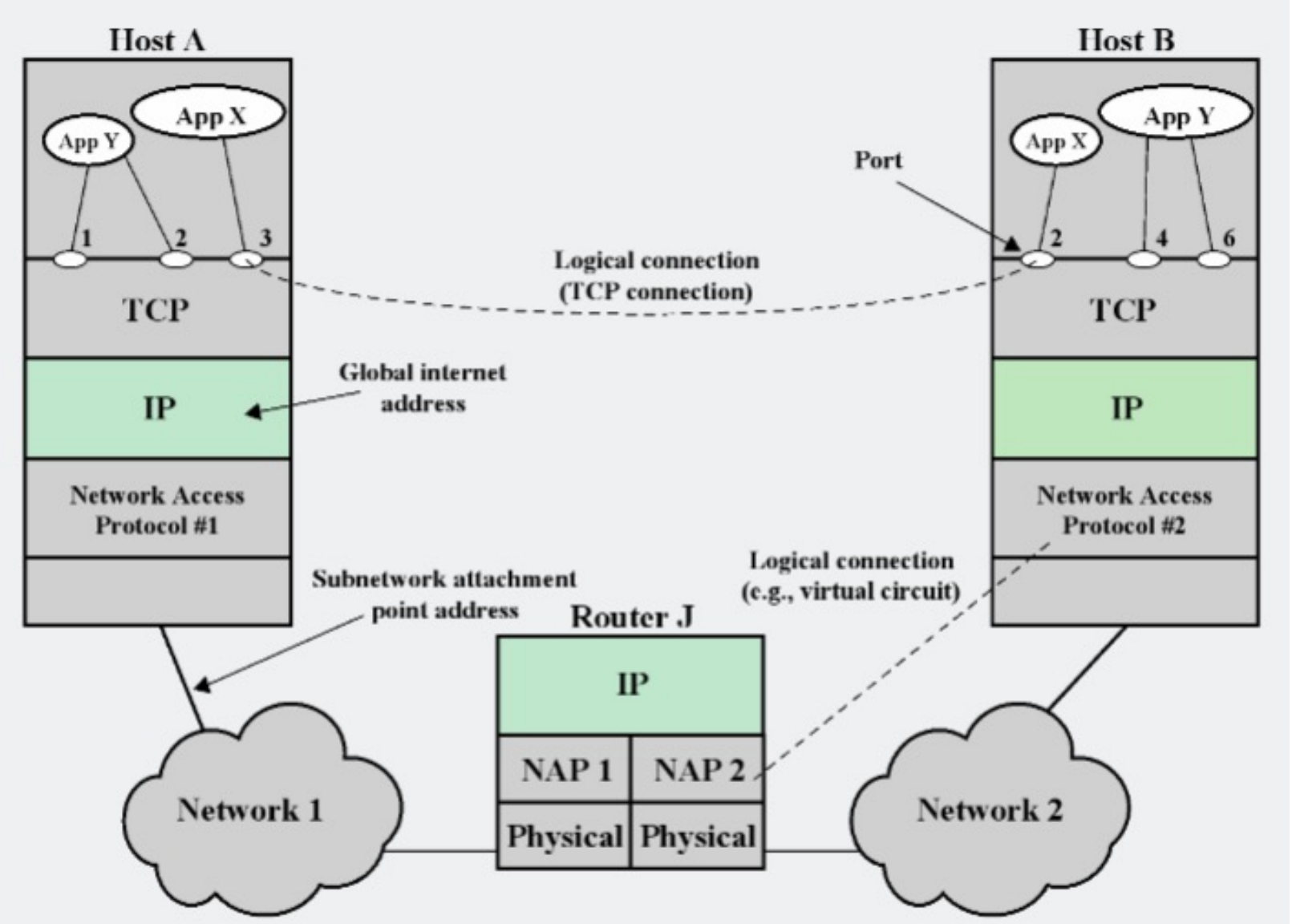
Table 5-3 Components of the TCP header (continued)

Fields in a TCP Header	Size in Bits	Description
SYN	1	Synchronize sequence numbers.
FIN	1	No more data from the sender.
Window size	16	Number of data octets starting from one.
TCP checksum	16	Checksum for error-checking of the header and data in the packet.
Urgent pointer	16	When the URG bit is set, this field is interpreted; it contains the value of the urgent pointer, which is higher than the sequence number in this segment.
Options (if any)	Varies	Routing options for the packet.
Padding	Varies	Pads the header to achieve a multiple of 32 bits.
Data (if any)	Varies	Data to be transmitted.

TCP header

- Port Number:
  - Ports are the communication point with upper layer (application layer)
  - Socket is pair of <IP address, Port number>
    - Uniquely identifies the endpoint address
  - For a full TCP communication two pairs of sockets are needed
    - One for server
    - One for client

TCP header


Host A
App Y
App X
1
2
3
TCP
IP
Network Access Protocol #1
Global internet address
Subnetwork attachment point address
Logical connection (TCP connection)
Port
Host B
App X
App Y
2
4
6
TCP
IP
Network Access Protocol #2
Logical connection (e.g., virtual circuit)
Router J
IP
NAP 1
NAP 2
Physical
Physical
Network 1
Network 2

TCP header

- Sequence Number (SeqNo):
  - 32 bits
  - No restart
  - Sequence numbers are relative to the size of TCP segments
    - Sequence number is incremented byte-by-bye
  - Initial Sequence Number (ISN) of a connection is set during connection establishment

TCP header

- Acknowledgement Number (AckNo):
  - Can be piggybacked
    - The response packet can include the ACK number as well
  - The AckNo identifies the next sequence number it is expecting
    - Means that all segments before that are received successfully
  - TCP uses the sliding window flow protocol to regulate the flow of traffic from sender to receiver
  - If it hasn't received a segment but received segments after that
    - It can only acknowledge the last segment received in order

TCP header

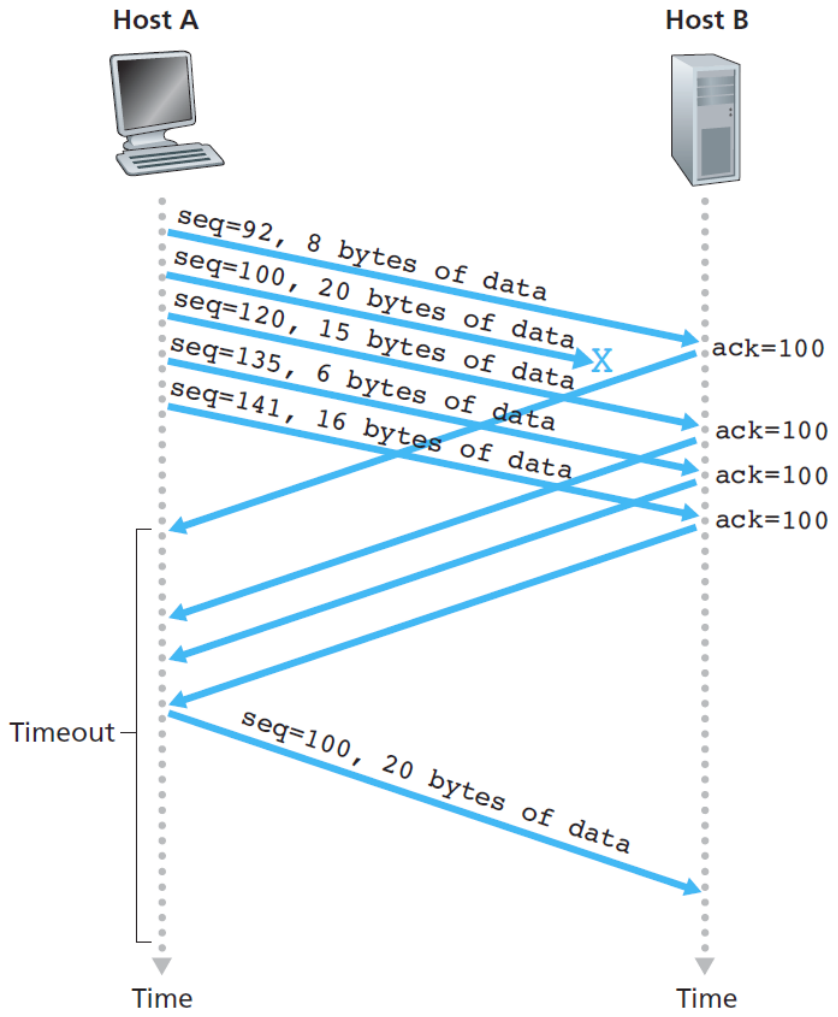


Figure 3.37 ♦ Fast retransmit: retransmitting the missing segment before the segment's timer expires

https://i.stack.imgur.com/fVsBQ.png

TCP header

- Header Length (4bits)
  - Length of header in 32-bit words
  - Note that TCP header has variable length (with minimum 20 bytes)

TCP header

- Flag bits
  - URG: Urgent pointer is valid
    - If the bit is set, the following bytes contain an urgent message in the range:
    - **SeqNo <= urgent message <= SeqNo+urgent pointer**
  - ACK: Acknowledgement Number is valid
  - PSH: PUSH Flag
    - Notification from sender to the receiver that the receiver should pass all data that it has to the application
    - Normally set by sender when the sender's buffer is empty
  - RST: Reset the connection
    - The flag causes the receiver to reset the connection
    - Receiver of a RST terminates the connection and indicates higher layer application about the reset

TCP header

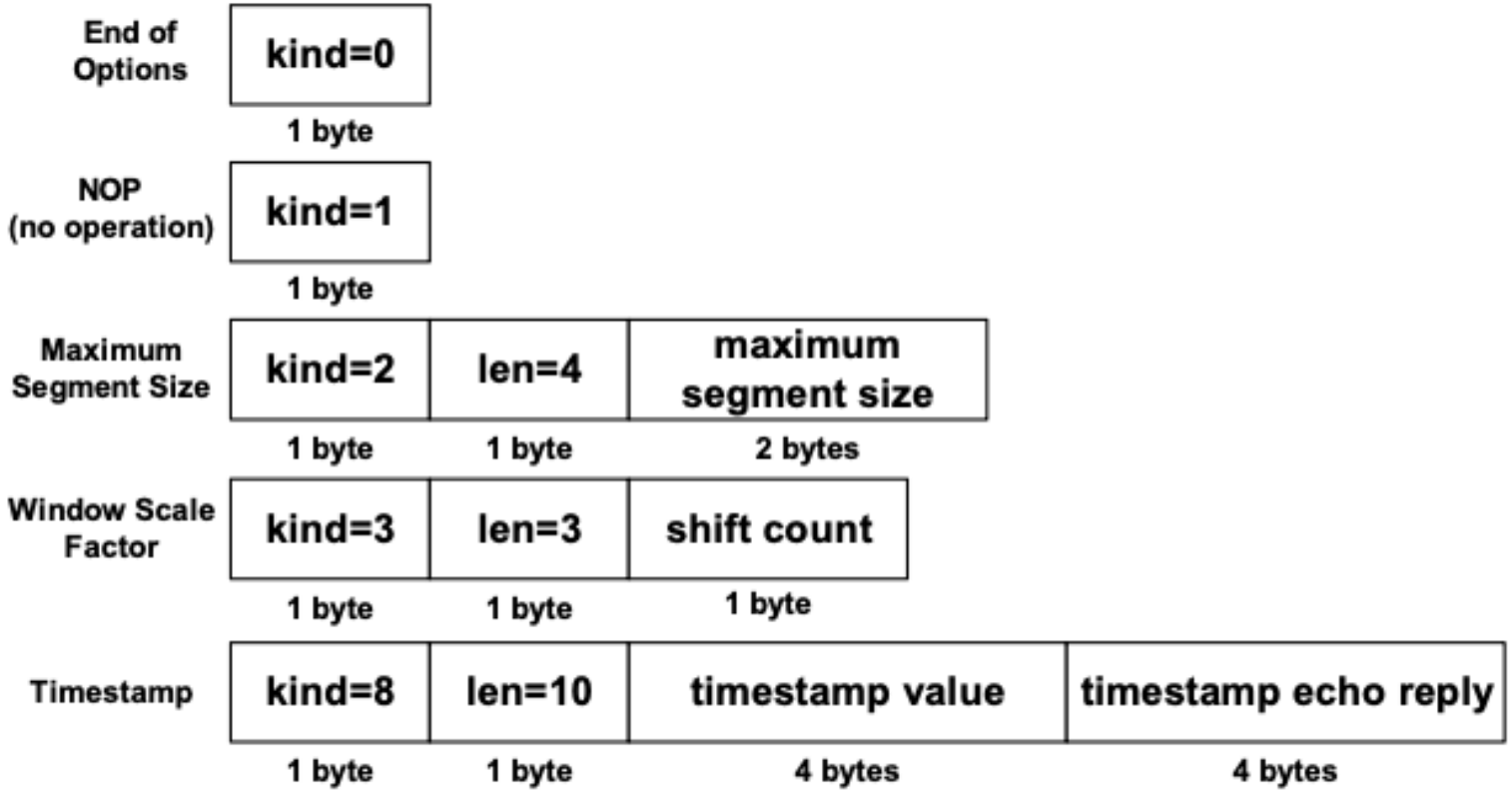
- Flag bits
  - SYN: Synchronize sequence numbers
    - Sent in the first packet when initiating a connection
  - FIN: Sender is finished with sending
    - Used for closing a connection
    - Both sides of a connection must send a FIN
  - Window Size:
    - Each side of the connection advertises the window size
    - Window size is the maximum number of bytes that a receiver can accept.
    - Maximum window size is 216-1= 65535 bytes

TCP header

- Flag bits
  - TCP Checksum:
    - TCP checksum covers over both TCP header and TCP data (also covers some parts of the IP header)
  - Urgent Pointer:
    - Only valid if URG flag is set

TCP header

- Flag bits
  - Options



TCP header

- Flag bits
  - Options
    - **NOP** is used to pad TCP header to multiples of 4 bytes
    - Maximum Segment Size
    - Window Scale Options
      - Increases the TCP window from 16 to 32 bits, I.e., the window size is interpreted differently
      - This option can only be used in the SYN segment (first segment) during connection establishment time
    - Timestamp Option
      - Can be used for roundtrip measurements

TCP Connection Establishment

- TCP uses a three-way handshake to open a connection:
  - ACTIVE OPEN: Client sends a segment with SYN bit set
  - Port number of client
  - initial sequence number (ISN) of client
  - PASSIVE OPEN: Server responds with a segment with SYN bit set
  - initial sequence number of server
  - ACK for ISN of client
  - Client acknowledges by sending a segment with:
    - ACK ISN of server

Connection Setup and Release

- Three-way handshake sets up and releases a connection
- **TCP packet flags:** URG, ACK, PSH, RST, SYN, and FIN
- Packets can have more than one flag set
  - Normally a packet will have only one flag sent, except with SYN/ACK or FIN/ACK
- Connection Setup
  - Source computer delivers a SYN packet to the destination computer
  - Packet has the initial sequence number (ISN)
  - ISN is indicated by whether the SYN bit is "set".
    - E.g., if the SYN bit is set to 1, the 32-bit sequence number represents ISN. if the SYN bit is not set, the 32-bit number represents the (ongoing) sequence number
  - Receiving computer transmits a SYN with an acknowledgment, ACK
  - Source computer sends an ACK to the destination computer as a response
    - With an "in-range" sequence number

Connection Setup and Release

- Connection Release
  - Source computer sends a FIN packet to the destination computer
  - Destination computer then sends a FIN/ACK packet
  - Source computer sends an ACK packet
  - Either computer could send an RST and close the session (reset) immediately

TCP Connection Termination

- Each end of the data flow must be shut down independently ("half-close")
  - If one end is done it sends a FIN segment. This means that no more data will be sent
- Four steps involved:
  1. X sends a FIN to Y (active close)
  2. Y ACKs the FIN,
     - (at this time: Y can still send data to X)
  3. Y sends a FIN to X (passive close)
  4. X ACKs the FIN.

Connection Setup and Release

Figure 2.2. TCP three-way handshake.

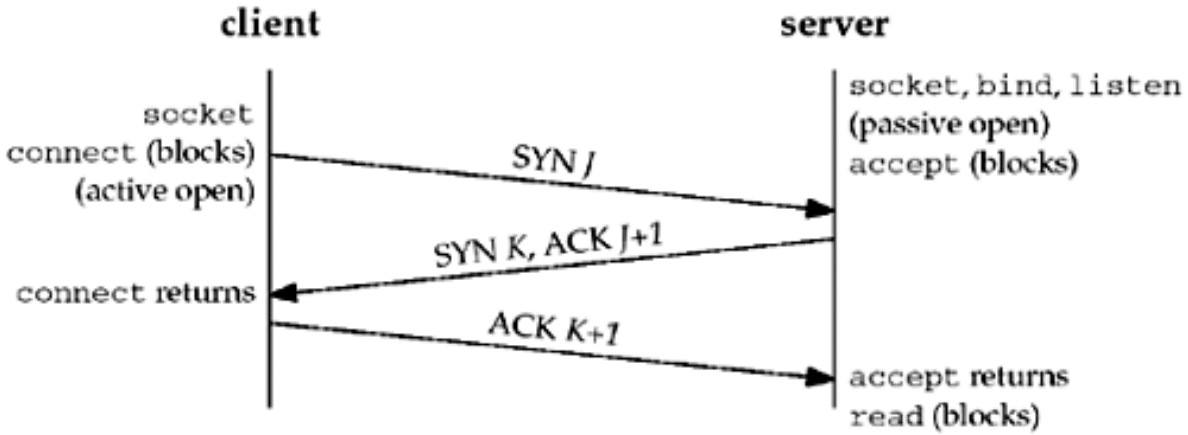
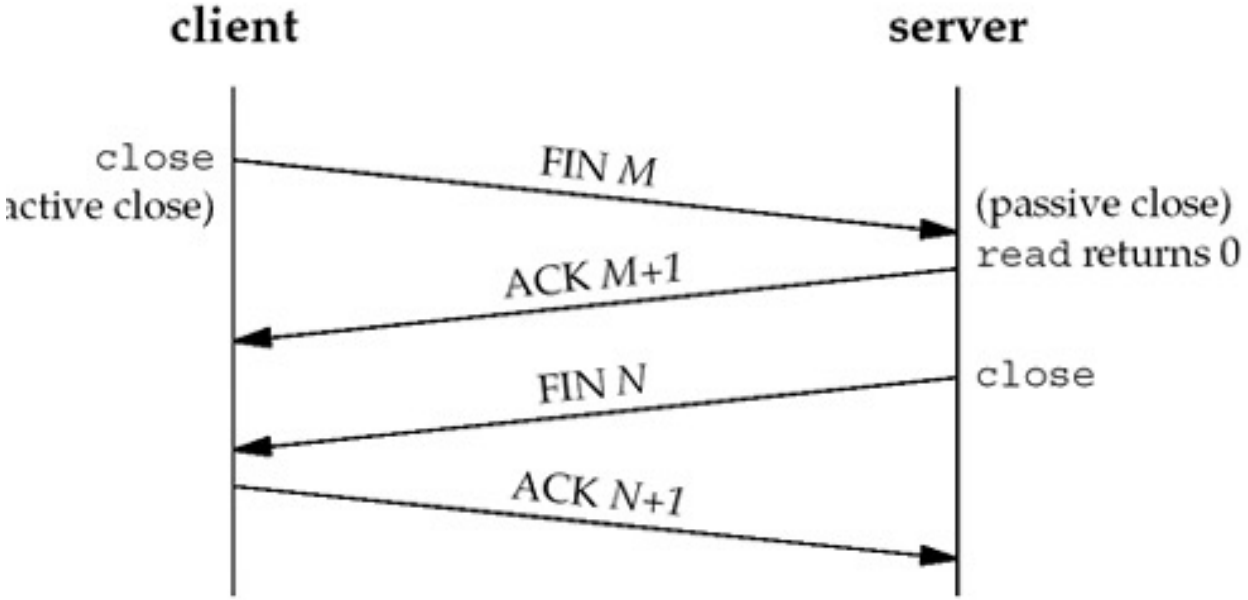


Figure 2.3. Packets exchanged when a TCP connection is closed.

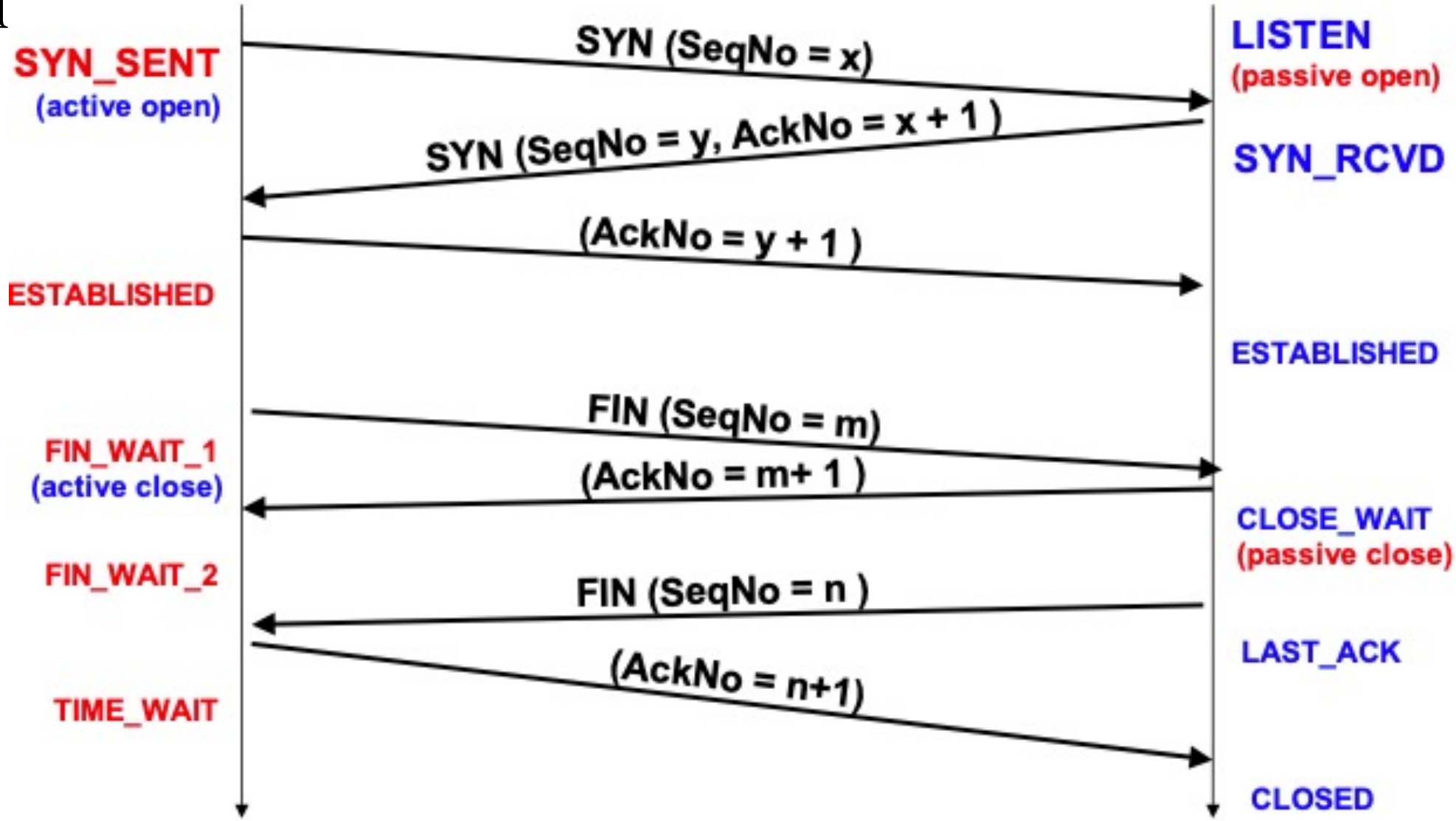


http://books.gigatux.nl/mirror/unixnetworkprogramming/0131411551_ch02lev1sec6.html

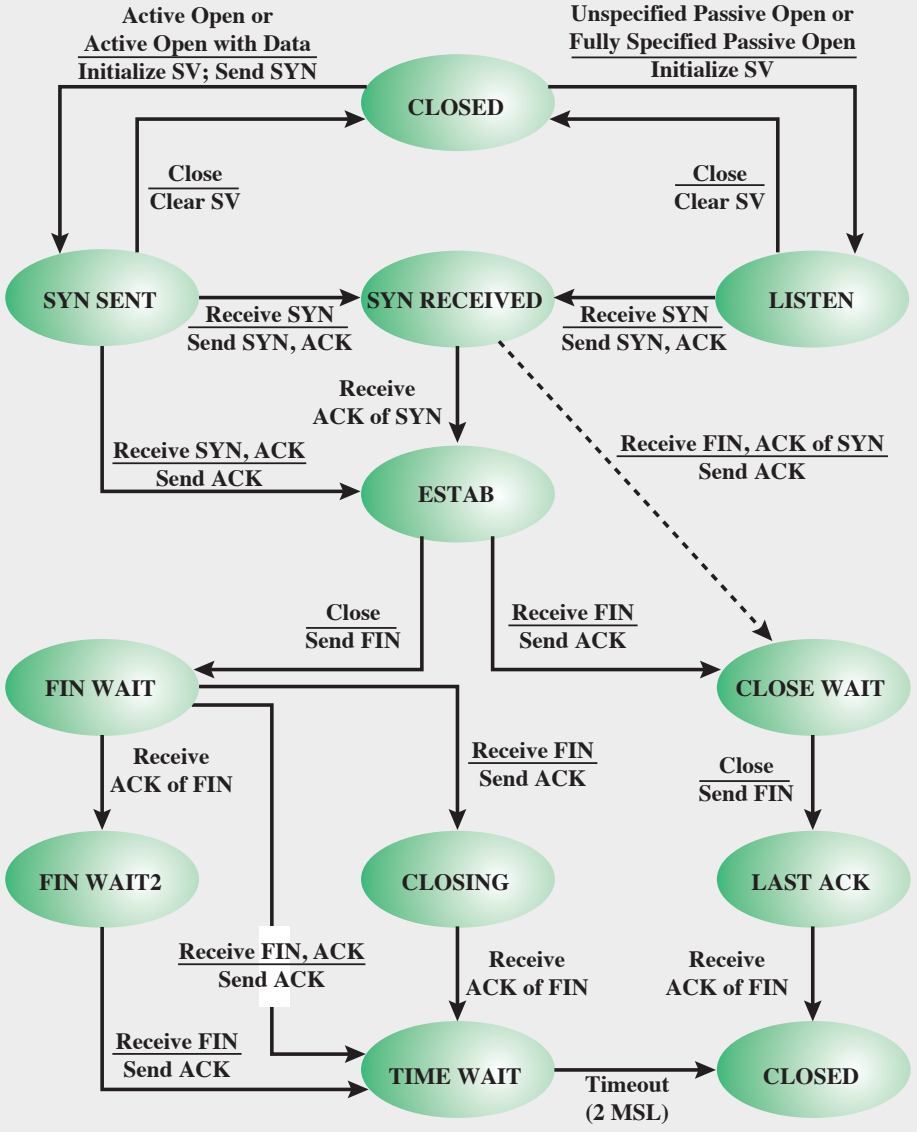
TCP States

State	Description
CLOSED	No connection is active or pending
LISTEN	The server is waiting for an incoming call
SYN RCVD	A connection request has arrived; wait for Ack
SYN SENT	The client has started to open a connection
ESTABLISHED	Normal data transfer state
FIN WAIT 1	Client has said it is finished
FIN WAIT 2	Server has agreed to release
TIMED WAIT	Wait for pending packets (“2MSL wait state”)
CLOSING	Both Sides have tried to close simultanesously
CLOSE WAIT	Server has initiated a release
LAST ACK	Wait for pending packets

TCP States in "Normal" Connection Lifetime



TCP States



SV = state vector
MSL = maximum segment lifetime

Figure 15.8 TCP Entity State Diagram

2MSL Wait State

- 2MSL Wait State = TIME_WAIT
  - When TCP does an active close, and sends the final ACK, the connection must stay in in the TIME_WAIT state for twice the maximum segment lifetime.
- 2MSL= 2 * Maximum Segment Lifetime
- Why?
  - TCP is given a chance to resent the final ACK. (Server will timeout after sending the FIN segment and resend the FIN)
- The MSL is set to 2 minutes or 1 minute or 30 seconds.

Resetting Connections

- Resetting connections is done by setting the RST flag
- When is the RST flag set?
  - Connection request arrives and no server process is waiting on the destination port
  - Abort (Terminate) a connection\
    - Causes the receiver to throw away buffered data
    - Receiver does not acknowledge the RST segment

TCP Timers

- All TCP sessions are tracked with timers built into the TCP protocol
- Timers used by TCP/IP
  - Connection establishment
    - A session will not be established if it takes longer than 75 seconds for the destination server to respond
  - FIN_WAIT
    - Waits for FIN packets. Its default value is 10 minutes
  - TIME_WAIT
    - Default value for this timer is two minutes
    - Waits for packets to arrive at the destination computer

TCP Timers

- Timers used by TCP/IP
  - KEEP_ALIVE
    - Checks to see if the destination computer is active
    - Computer may send a test packet every two hours to verify whether the other computer is alive and idle

Summary

- TCP/IP model
- IP
  - IP header
- TCP
  - Connections
  - Byte stream service
  - Segment
  - Header
  - Connection establishment
  - Connection Release
  - Timers

References

- [Textbook 1] Chapter 5