



# Batch: *A4* Roll No.:*16010422211* Experiment No.:*4*

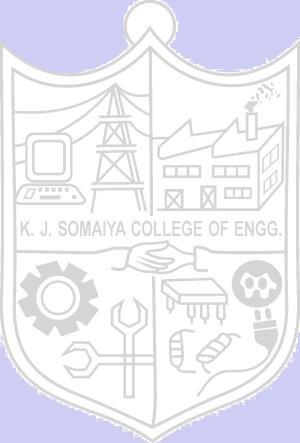
**Aim:** To write a program to implement TCP header



**Resources Used:** Java /C/C++/Python

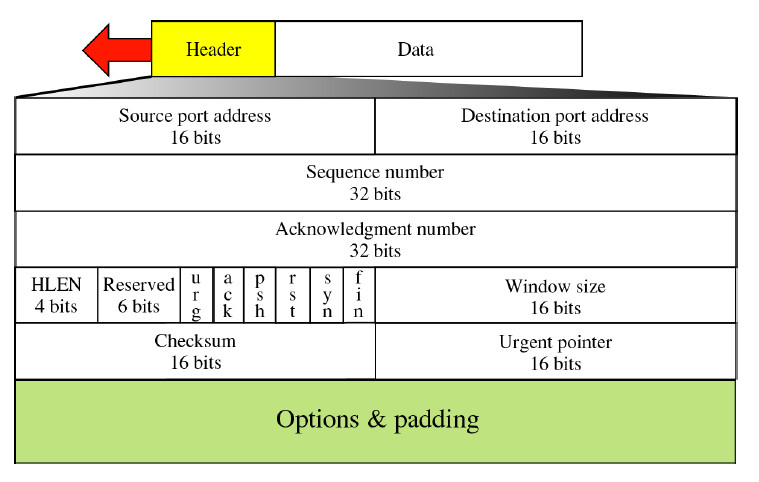


# Theory:

The transport service is implemented by a transport protocol used between the two transport entities. Transport protocols resemble the data link protocols. Both have to deal with error control, sequencing, and flow control, among other issues. Differences are due to major dissimilarities between the environments in which the two protocols operate. The Internet has two main protocols in the transport layer, Connectionless protocol: UDP Connection-oriented protocol: TCP TCP (Transmission Control Protocol) was designed to provide a reliable end-to- end byte stream over an unreliable internetwork .

**The TCP Protocol:** Every byte on a TCP connection has its own 32-bit sequence number. Separate 32-bit sequence numbers are used for acknowledgements and for the window mechanism. The sending and receiving TCP entities exchange data in the form of segments. A TCP segment consists of a fixed 20-byte header (plus an optional part) followed by zero or more data bytes. Two limits restrict the segment size. Each segment, including the TCP header, must fit in the 65,515-byte IP payload. Each network has a Maximum Transfer Unit, or MTU, and each segment must fit in the MTU. In practice, the MTU is generally 1500 bytes (the Ethernet payload size) and thus defines the upper bound on segment size. A segment that is too large for a n/w can be broken into multiple segments by a router. The basic protocol used by TCP entities is the **sliding window protocol**. When a sender transmits a segment, it also starts a timer. When the segment arrives at the destination, the receiving TCP entity sends back a segment (with data if any exist, otherwise without data) bearing an acknowledgement number equal to the next sequence number it expects to receive. If the sender's timer goes off before the acknowledgement is received, the sender transmits the segment again.

**The TCP Segment Header:** The **Source port** and **Destination port** fields identify the local end points of the connection. A port plus its host's IP address forms a 48-bit unique end point (TSAP). The **Sequence number** defines the number of the first data byte contained in that segment and **Acknowledgement number** specifies the next byte expected, not the last byte correctly received. Both are 32 bits long. The **TCP header length** tells how many 32-bit words are contained in the TCP header.





# Activity:

Write a program to accept the input in the hexadecimal form (continuous string) and display the value of each field of TCP header.



Program: a = "000053200170000100000000500207FF00000000"

import re

def is\_valid\_hexadecimal(input\_str):

    # Define a regular expression pattern to match a valid hexadecimal string

    pattern = r'^[0-9A-Fa-f]+$'

    return re.match(pattern, input\_str) is not None

if len(a) == 40:

    sp\_str = a[0:4]

    if is\_valid\_hexadecimal(sp\_str):

        sp = int(sp\_str, 16)

        print("Source port:", sp)

    dp\_str = a[4:8]

    if is\_valid\_hexadecimal(dp\_str):

        dp = int(dp\_str, 16)

        print("Destination port:", dp)

    seq\_no\_str = a[8:16]

    if is\_valid\_hexadecimal(seq\_no\_str):

        seq\_no = int(seq\_no\_str, 16)

        print("Sequence number:", seq\_no)

    ack\_no\_str = a[16:24]

    if is\_valid\_hexadecimal(ack\_no\_str):

        ack\_no = int(ack\_no\_str, 16)

        print("Acknowledge number:", ack\_no)

    hlen\_str = a[24:25]

    if is\_valid\_hexadecimal(hlen\_str):

        hlen = int(hlen\_str, 16)

        print("header length:", hlen \* 4)

    middle\_str = a[26:28]

    if is\_valid\_hexadecimal(middle\_str):

        middle = bin(int(middle\_str, 16)).zfill(8)

        urg\_flag = int(middle[2], 2)

        print("URG:", urg\_flag)

    w\_size\_str = a[28:32]

    if is\_valid\_hexadecimal(w\_size\_str):

        w\_size = int(w\_size\_str, 16)

        print("window size:", w\_size)

    check\_sum\_str = a[32:36]

    if is\_valid\_hexadecimal(check\_sum\_str):

        check\_sum = int(check\_sum\_str, 16)

        print("Check sum:", check\_sum)

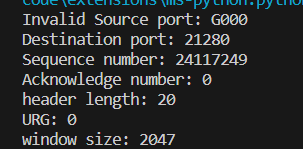
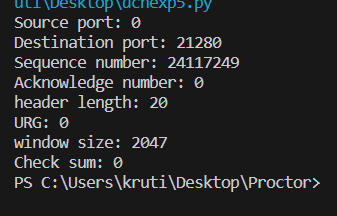
else:

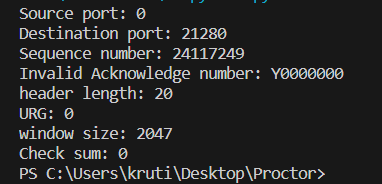
    print("Total length is less than 40")

#All boundaries cases included



**Output:**

****

****



# Questions:

1. The unit of data transfer between two devices using TCP is called *TCP Segment*.
2. Which type of addressing is used at Transport Layer?
   1. Port addressing
   2. Logical addressing
   3. Physical Addressing

Ans-*A) port addressing*

1. What is the difference between TCP and UDP?

Ans- *TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) are two commonly used transport layer protocols in computer networking:*

*1. Reliability:*

*- TCP: Provides a reliable and connection-oriented communication. It ensures that data is delivered in the correct order and without errors. It uses acknowledgments and retransmissions to achieve this reliability.*

*- UDP: Offers unreliable and connectionless communication. It does not guarantee delivery or order of data. It's often used for applications where speed and efficiency are prioritized over reliability.*

*2. Connection:*

*- TCP: Establishes a connection before data transfer and ensures a full duplex (two-way) communication channel.*

*- UDP: Does not establish a connection; it simply sends data packets without prior setup. It allows for broadcasting and multicasting.*

*3. Overhead:*

*- TCP: Has higher overhead due to its reliability mechanisms (acknowledgments, retransmissions, and flow control).*

*- UDP: Has lower overhead because it lacks the overhead associated with ensuring reliability.*

*4. Use Cases:*

*- TCP: Typically used for applications where data integrity and reliability are crucial, such as web browsing, email, file transfer (FTP), and online gaming.*

*- UDP: Suitable for applications where low latency and real-time communication are more important than reliability, such as video streaming, voice over IP (VoIP), online gaming (for real-time interactions), and DNS (Domain Name System).*

*In short, TCP is connection-oriented, reliable, and suitable for applications that require data integrity, while UDP is connectionless, less reliable, and preferable for applications where low latency and real-time communication are essential. The choice between TCP and UDP depends on the specific needs of the application.*

# Outcomes:

# CO2. Enumerate the layers of the OSI model and TCP/IP model, their functions and Protocols.



**Conclusion:** *Learnt about the TCP header in detail.*



# Grade: AA / AB / BB / BC / CC / CD /DD

**Signature of faculty in-charge with date**



# References:

**Books/ Journals/ Websites:**

* Behrouz A Forouzan, Data Communication and Networking, Tata Mc Graw hill, India, 4th Edition
* A. S. Tanenbaum, “Computer Networks”, 4th edition, Prentice Hall