Experiment No. 4

Title: TCP Header implementation

Batch: A1 **Roll No.: 16010421015** **Experiment No.**

Aim: To write a program to implement TCP header

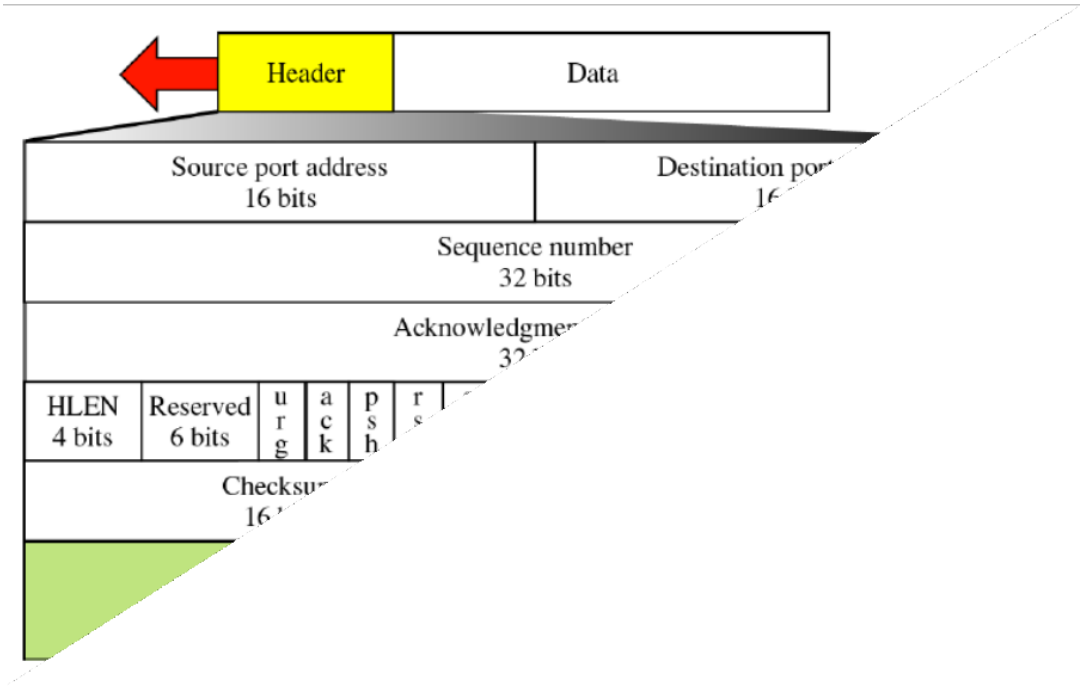
Resources Used: Java /C/C++/Python

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:

```
import numpy as np
input =
"01000110100010010001001101111111010000001010101100000101101000100011010000101100101101111101001001100011011101010000000001011010110010101101011111010001010111100"

def binaryTodecimal(n):
    decimal = 0
    power = 1
    while n>0:
        rem = n%10
        n = n//10
        decimal += rem*power
        power = power*2

    return decimal

source_port_address = input[0:15]
a = int(source_port_address)
print("Source Port Address:",binaryTodecimal(a))

destination_port_address = input[16:31]
b = int(destination_port_address)
print("Destination Port Address: ", binaryTodecimal(b))

sequence_number = input[32:63]
c = int(sequence_number)
print("Sequence Number:",binaryTodecimal(c))
```

```
acknowldegement_number = input[64:95]
d = int(acknowldegement_number)
print("Acknowledgement Number:",binaryTodecimal(d))

hlen = input[96:99]
e = int(hlen)
print("HLEN:",binaryTodecimal(e))

reserved = input[100:105]
f = int(reserved)
print("Reserved:", binaryTodecimal(f))

urg = input[106]
g = int(urg)
print("URG:", binaryTodecimal(g))

ack = input[107]
h = int(ack)
print("ACK:", binaryTodecimal(h))

pch = input[108]
i = int(pch)
print("PCH:", binaryTodecimal(i))

rst = input[109]
j = int(rst)
print("RST:", binaryTodecimal(j))

syn = input[110]
k = int(syn)
print("SYN:", binaryTodecimal(k))

fin = input[111]
l = int(fin)
print("FIN:", binaryTodecimal(l))

window_size = input[112:127]
m = int(window_size)
print("Window Size:", binaryTodecimal(m))

checksum = input[128:143]
n = int(checksum)
print("Checksum:", binaryTodecimal(n))

urgent = input[144:159]
o = int(urgent)
print("Urgent Request:", binaryTodecimal(o))
```

Output:

```
Source Port Address: 9028
Destination Port Address:  2495
Sequence Number: 1084949922
Acknowledgement Number: 875345874
HLEN: 6
Reserved: 13
URG: 1
ACK: 0
PCH: 1
RST: 0
SYN: 1
FIN: 0
Window Size: 90
Checksum: 19159
Urgent Request: 20830
>
```

Questions:

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

d) None of the Above

3) What is the difference between TCP and UDP?

Feature	TCP	UDP
Connection status	Requires an established connection to transmit data (connection should be closed once transmission is complete)	Connectionless protocol with no requirements for opening, maintaining, or terminating a connection
Data sequencing	Able to sequence	Unable to sequence
Guaranteed delivery	Can guarantee delivery of data to the destination router	Cannot guarantee delivery of data to the destination
Retransmission of data	Retransmission of lost packets is possible	No retransmission of lost packets
Error checking	Extensive error checking and acknowledgment of data	Basic error checking mechanism using checksums

Method of transfer	Data is read as a byte stream; messages are transmitted to segment boundaries	UDP packets with defined boundaries; sent individually and checked for integrity on arrival
Speed	Slower than UDP	Faster than TCP
Broadcasting	Does not support Broadcasting	Does support Broadcasting
Optimal use	Used by HTTPS, HTTP, SMTP, POP, FTP, etc	Video conferencing, streaming, DNS, VoIP, etc

Outcomes:

CO2: Enumerate the layers of the OSI model and the TCP/IP model, their functions and protocols

Conclusion:

We understood about the different layers of the TCP model and also wrote a program to show values of each TCP header

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