# Lec 64 TCP: Transmission Control Protocol – TCP Header

TCP
- → Byte Streaming
- → Connection Oriented
- → Full Duplex
- → Piggybacking
- → Error Control
- → Flow Control
- → Congestion Control

Byte Streaming – Divides data into segments (collection of bits)

Connection Oriented – Reliability – 3-way handshaking

Piggybacking – Go Back N, Selective Repeat

TCP Header (20–60 B)

| | | |
|---|---|---|
| Source Port 16 Bit | Destination Port 16 Bit | Source Port |
| Sequence Number 32 Bit | | 0–65535 numbers<br>0–1023 – Well known |
| Acknowledge No. 32 Bit | | |
| HLEN 4 Bit \| U R G \| A C K \| P S H \| R S T \| S Y N \| F I N \| Window Size 16 Bit | | Sequence number is the representation of sequence of bytes |
| Checksum 16 Bit | 16 bit Urgent Pointer | |
| Options & Padding 40 Bytes | | |

L-65 Part 2

→ Length of header

HLEN → 4 bits

Scale of 4

Eg: If HLEN = 1000 → 8

header length = 8 × 4 = 32

URG → 1 if the data contains urgent data

ACK → 1 if the other side has acknowledged

PSH → 1 if the data has to be pushed immediately

RST → Reset

SYN → Synchronization

FIN → 1 if the other side has to terminate

Window Size → Size of the receiving window.

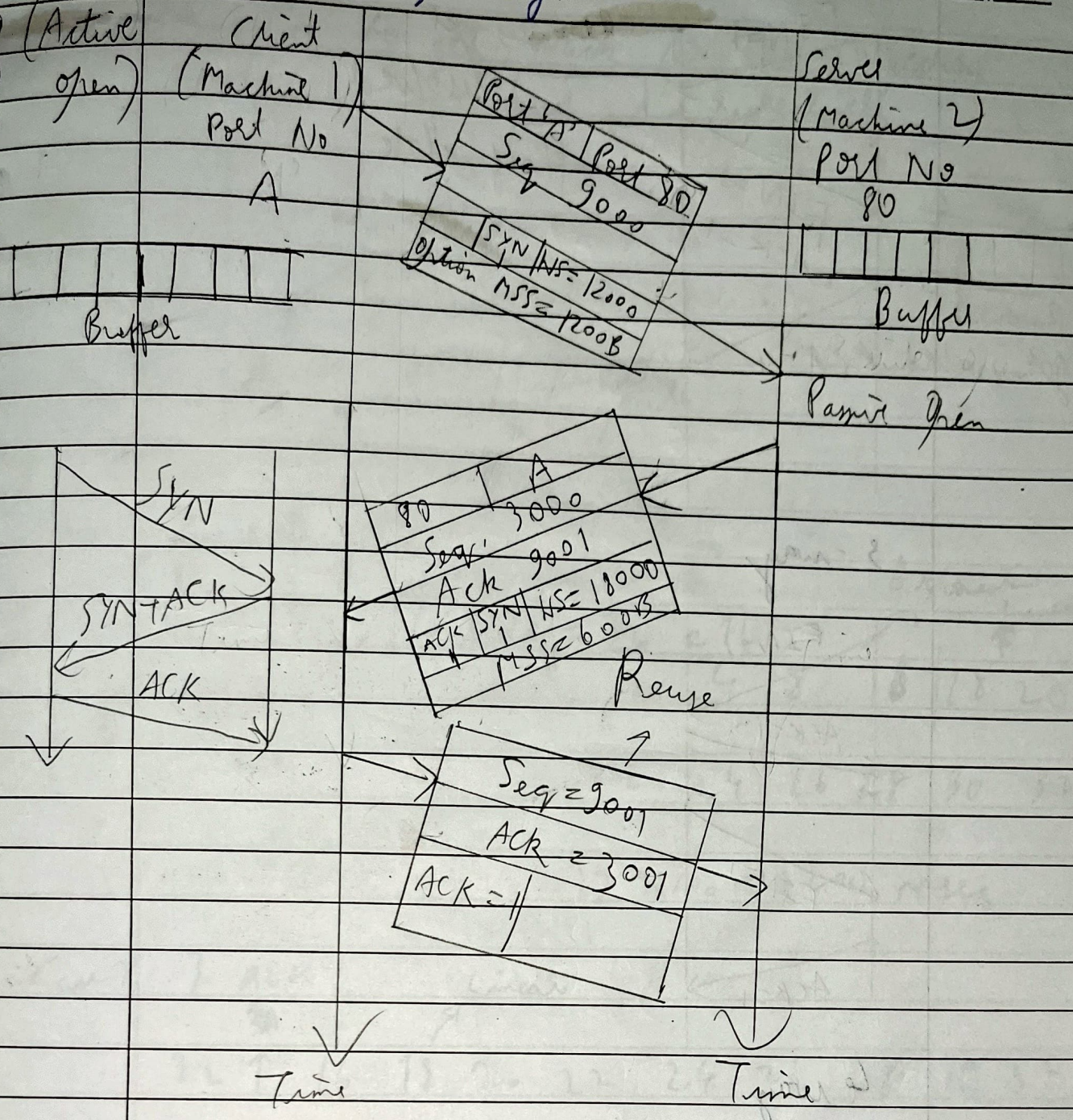
URGENT Pointer → Tells where the data is urgent

Max Segment Size → Is of a network

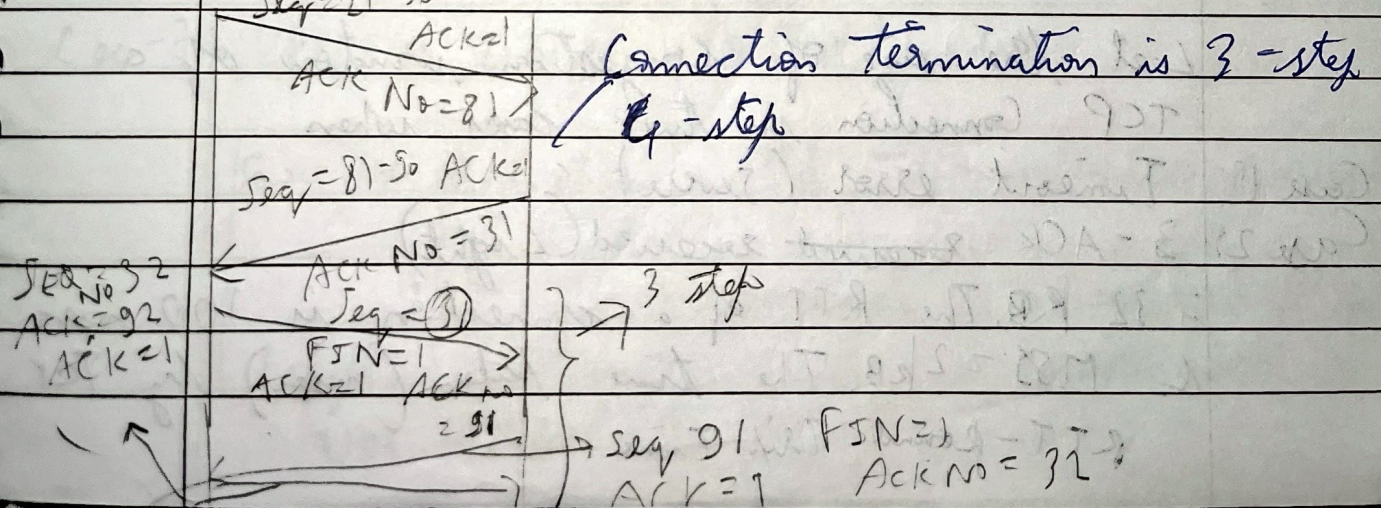
L 66 TCP connection establishment and connection termination

# 3 Way connection

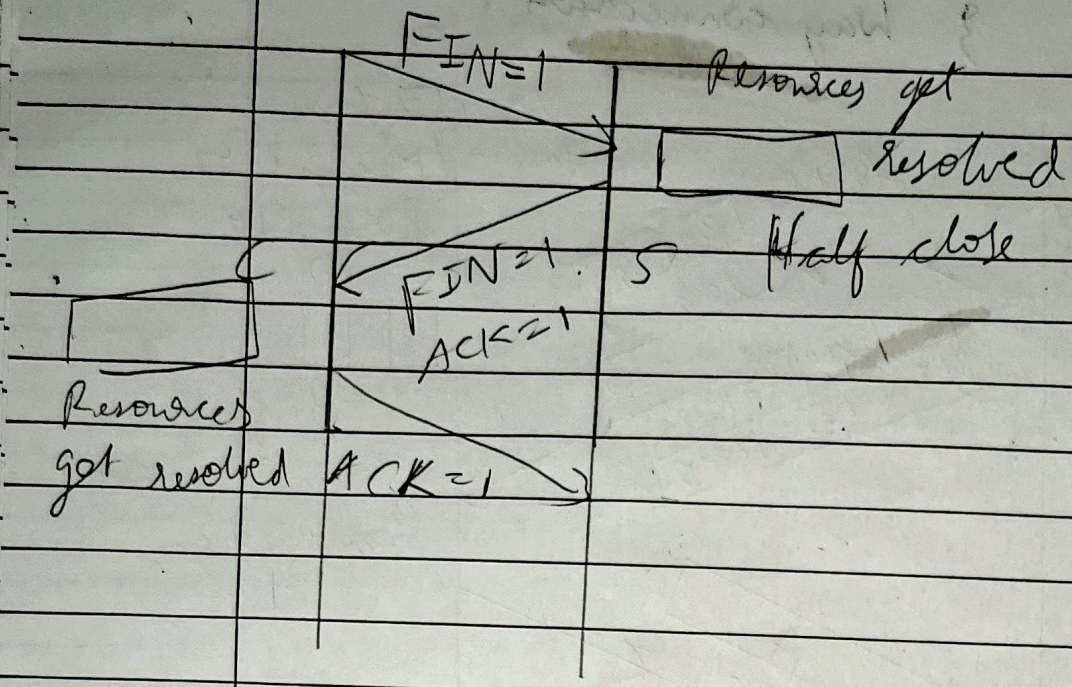




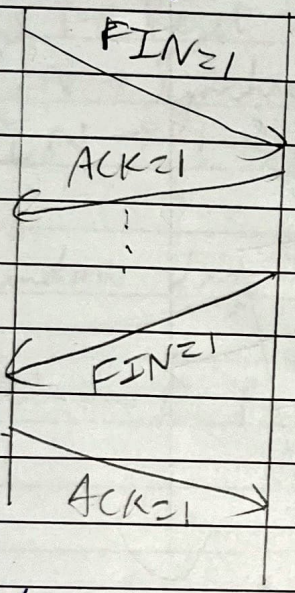

## Lec-68 Connection Termination in TCP




Connection termination is 3-step / 4-step




3 - way




4 way

L-69 TCP Congestion Control

Let the size of congestion window of a TCP Connection in two cases when

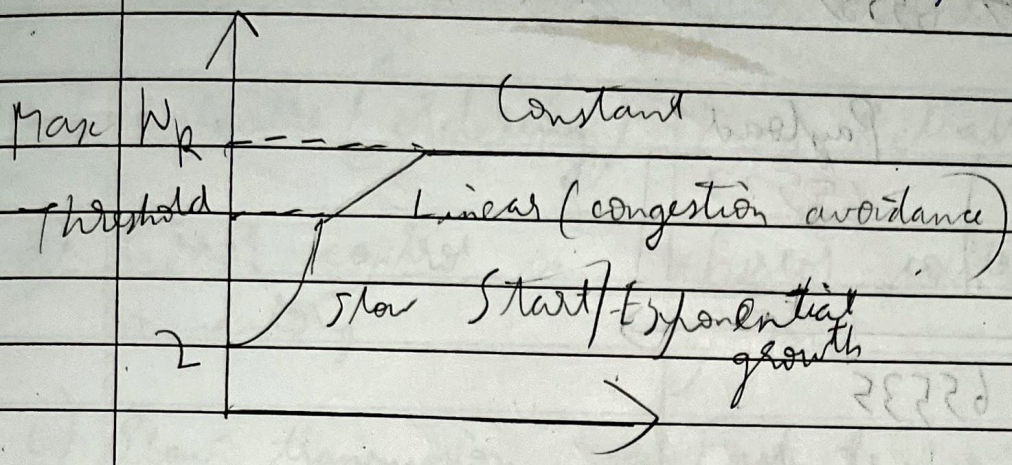
Case 1: Timeout error (Severe)

Case 2: 3-ACK ~~received~~ received (Light)

is 32 KB. The RTT of a connection is 100 msec & MSS = 2 KB. The time taken (msec) by TCP

RTT - Round Trip Time

Connection to get back to 3kB congestion window is ___ and ___ respectively




Case 1 Timeout

32 ↑ 2 [threshold = 32 ÷ 16

Exponential Linear

2 4 8 16 18 20

22 24 26 28 30 32

12 × 100 = 1200 msec

Case 2 } ACK

Linear

32 ↑ 16 18 20 22 24 26 28 30 32

9 × 100 = 900 ~~msec~~ ms

Sec-79 UDP (User Datagram Protocol) header

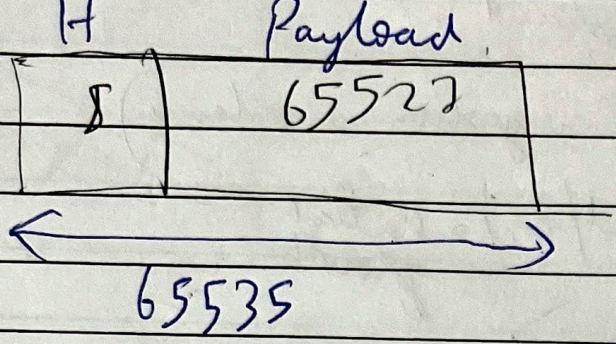
| Source port | Destination port |
|---|---|
| 16 | 16 |
| 16 | 16 |
| Length | Checksum |

UDP Header (8B fixed)

- Connectionless (unreliable)
- Checksum = UDP Header + UDP Data + Pseudo header of IP

→ No order

$2^{16} \rightarrow 65535$




Checksum in IPv4 is optional
Checksum in IPv6 is mandatory

---

Lec 71 Advantages of UDP protocol over TCP

UDP Applications Less overhead

1) Query Response Protocol (One Request one reply) [DNS] [RIP]
2) Speed (Online games, Voice Over IP)
3) Broad Casting / Multicasting [RIP]
4) Continuous Streaming [Skype, Youtube] (Stateless)

---

Lec 72 TCP vs UDP differences

| | TCP | UDP |
|---|---|---|
| 1) | Connection oriented | 1) Connectionless |
| 2) | Reliable (Ordering) | Less Reliable (No order) |
| 3) | Error control is mandatory | Error control is optional |
| 4) | Slow transmission | Fast transmission |
| 5) | More overhead | Less overhead |
| 6) | Flow control, Congestion Control | No Flow control, Congestion control |
| 7) | HTTP, FTP use TCP | DNS, BOOT P DHCP, RIP use UDP |

L:-78 HTTP FTP, SMTP, POP - All application layer protocols

* Port no. 80

* Itself not reliable, but use TCP to achieve reliability

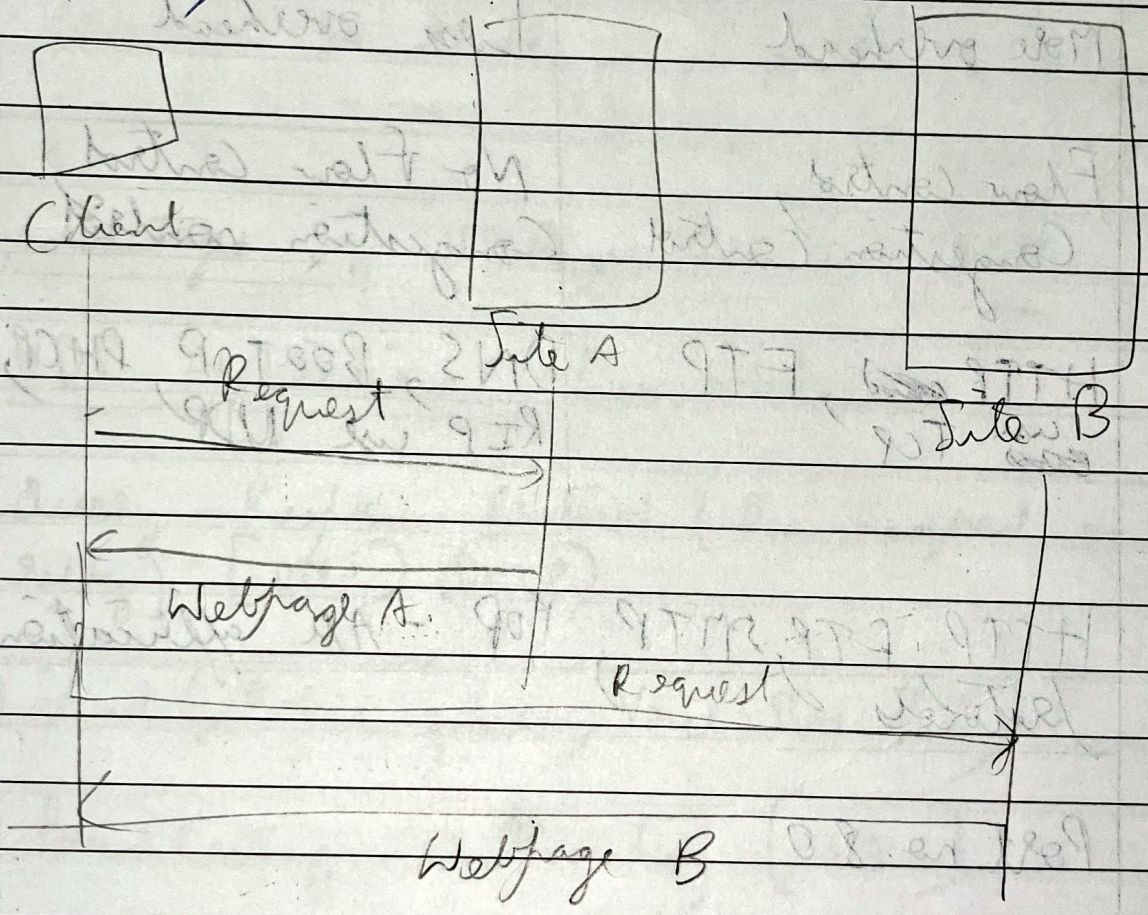
* Inband Protocol

* Stateless → Does not store the state of the device used, but some Organisations use cookies to store information.

* HTTP 1.0 Non-Persistent → Once the webpage is closed, the access to the webpage is lost.

* HTTP 1.1 Persistent → The access to the webpage is not lost as long as the user is not logged out.

* Commands (Head, Get, Post, Put, Delete, Connect)




GET → Requests a document from the server

HEAD → Requests information about a document but not the document itself

POST → Sends some information from the client to the server

PUT → Sends a document from the server to the client

TRACE → Echoes the incoming request
CONNECT → Reserved
OPTION → Inquires about available options

## Lec 29 : Persistent vs Non-Persistent HTTP

| | Persistent HTTP (HTTP/1.1) | Non-Persistent HTTP (HTTP/1.0) |
|---|---|---|
| 1) | Server leaves connection open after sending response | It requires 2 RTTs per object |
| 2) | Less overhead | More overhead |