Difference b/w DVR and LSR:

DVR	LSR
→ 1980's	→ 1990's
→ BW is requirement is less because Distance vectors are small and no flooding.	BW requirement is high because flooding is required.
→ local knowledge (only about neighbours)	→ Global knowledge (about entire network)
→ Traffic is less	→ Traffic is very high.
→ Periodic updates are done	→ Periodic updates are done.
→ Converges slowly → Final routing Table will take time	→ Converges faster → Final routing tables will come fast

DVR

→ Count to ∞ problem

→ Persistent looping.

↓

Permanent

→ RIP uses DVR

→ Simple

LSR

→ No Count to ∞ problem

→ Transient looping

↓

Temporary.

→ OSPF uses LSR

→ Complex.

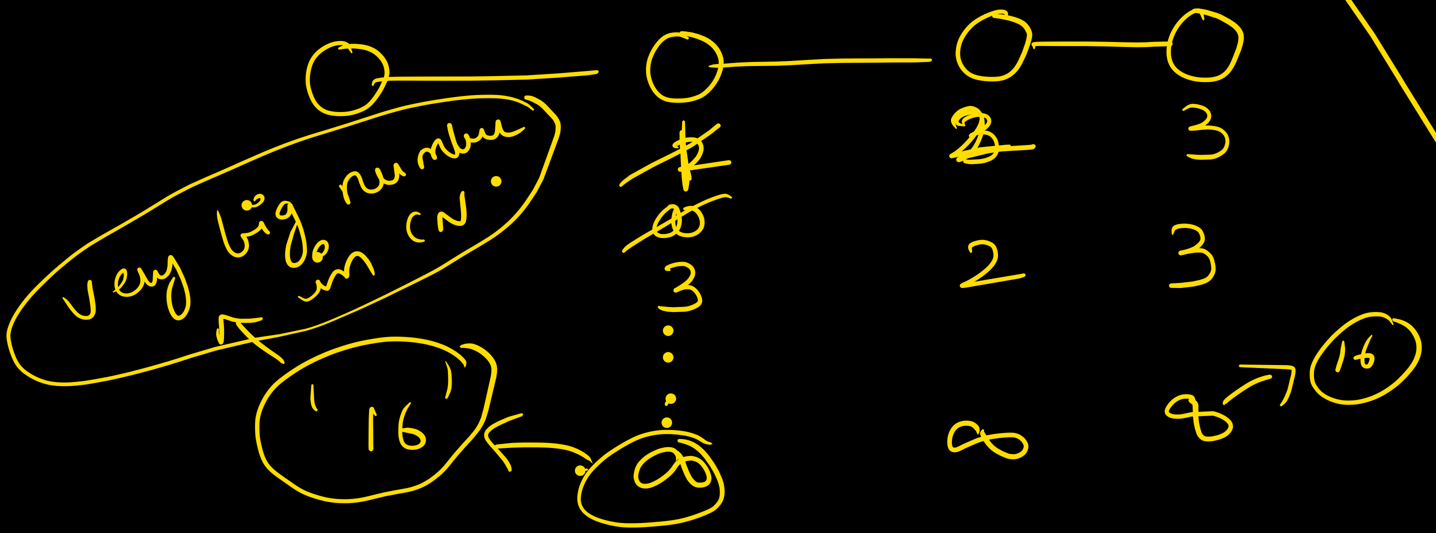
$\boxed{\text{RIP and OSPF}}$

RIP (Routing information Protocol)

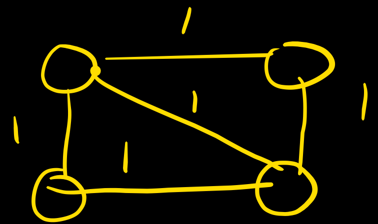
DVR → algorithm

RIP → implementation

In RIP '16' is taken as '∞'



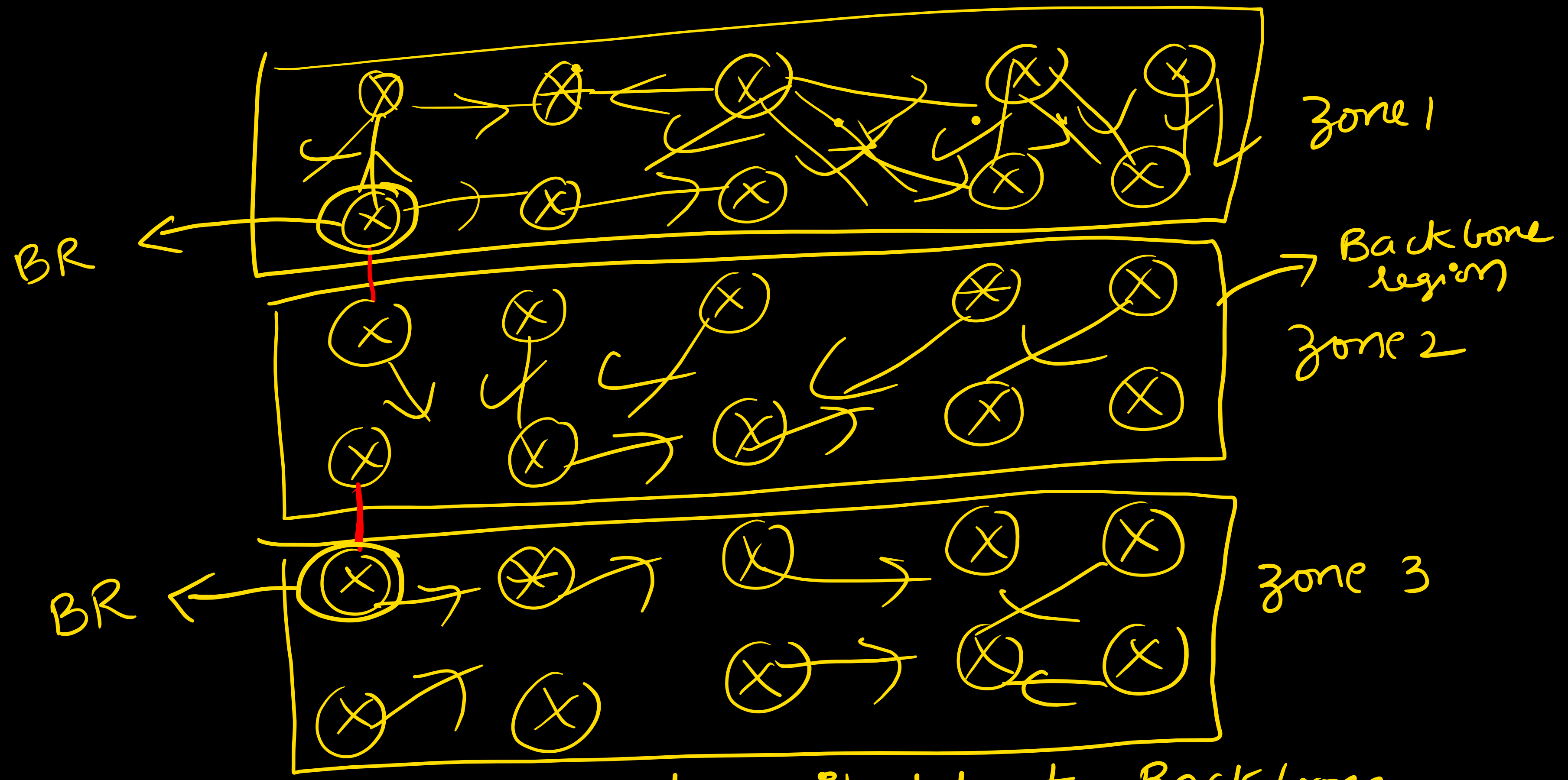
In RIP → Every edge has weight '1'.



16 hops or edges are not really required in real world.

OSPF: open shortest path first. → Implements LRR

All the routers are divided into several regions or zones.



Flooding is done within a zone only.

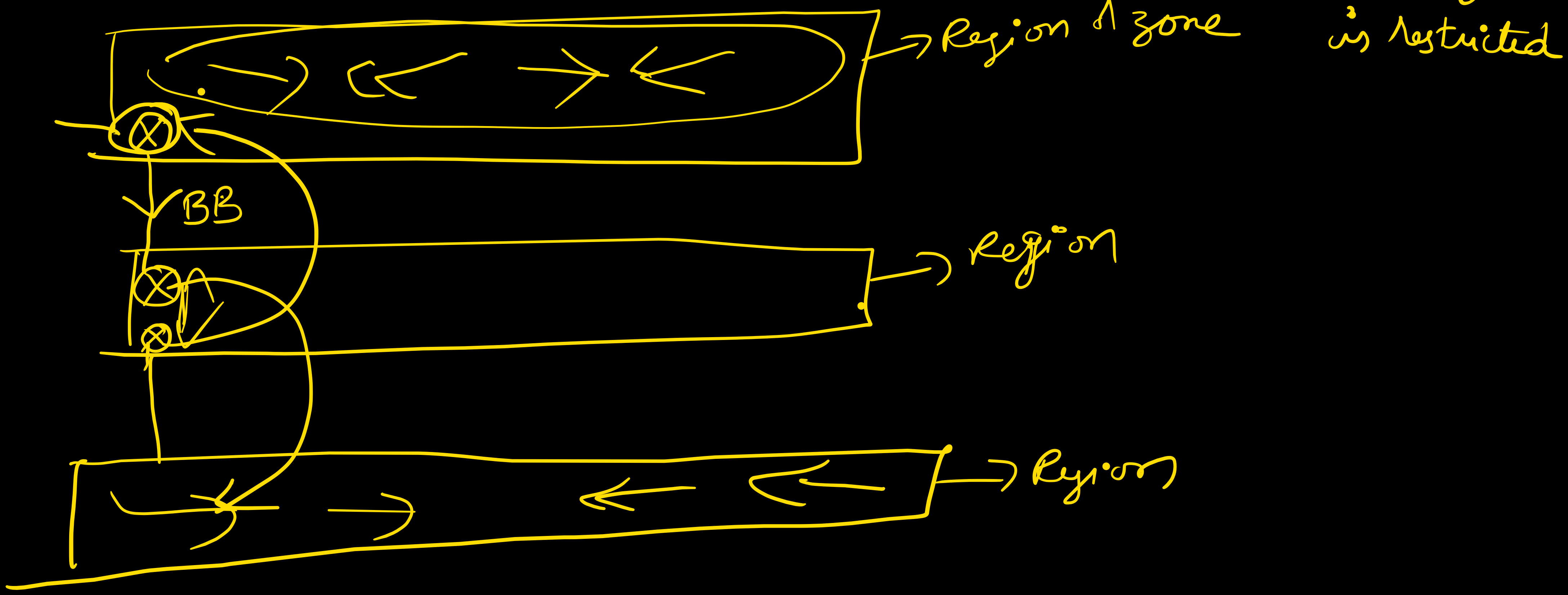
Every region or zone will have a border Router.

Border Router will be connected to Backbone region

Border routers will transmit data to Backbone Region

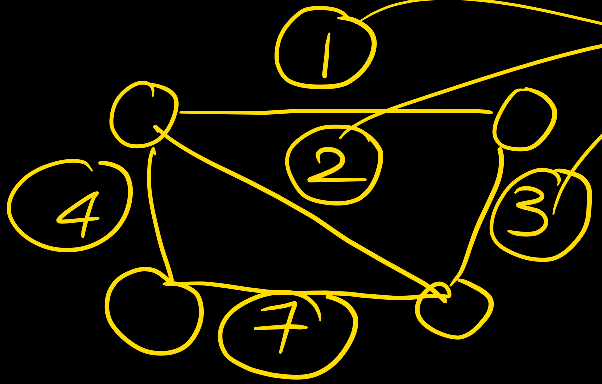
LSR → uses flooding → high traffic

∴ OSPF → uses regions and only Flooding within a region.



The edge weights on the N/w are determined by

$ N/w admin.



$ N/w admin.

In RIP → always '1'.

not in GATE syllabus

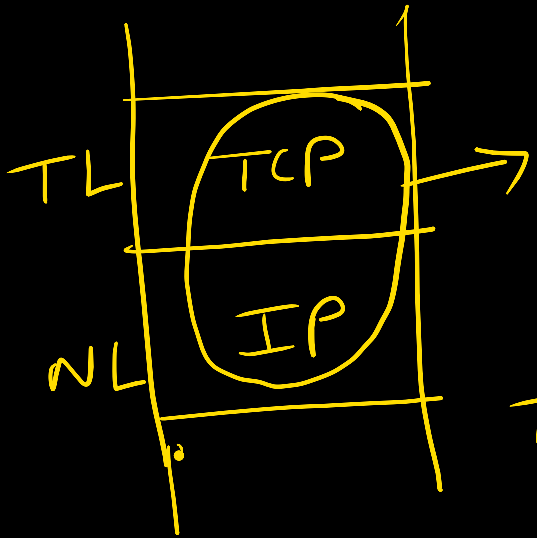
EIGRP : (Enhanced interior gateway Routing protocol)

Combination of RIP and OSPF

Not required.

TCP: Transmission control protocol:

TCP works at Transport layer.



TCP/IP → only 5 layers

ISO/OSI → 7 layers.

TCP → End to end.

AL
TL
NL
DLL
PL

TCP is Byte stream protocol.

↓

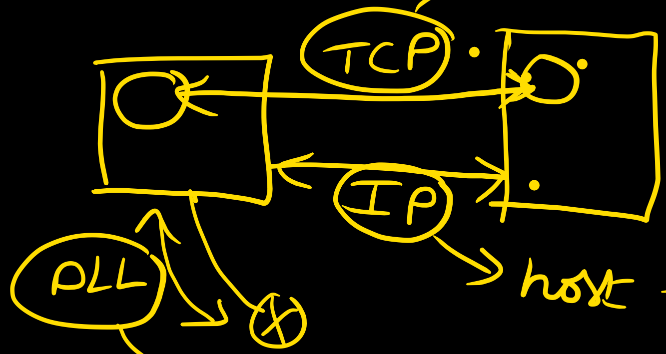
$\boxed{\text{Every Byte is numbered.}}$

IP is packet stream

↓

Every packet is numbered.

→ End to end



→ host to host

→ hop to hop.

ISO-OSI → Theory.

TCP-IP → Implementation

TCP header:

bits

<table>
<tr><td colspan="2">Source port (16)</td><td colspan="7">Destination port (16)</td></tr>
<tr><td colspan="9">Sequence number (32)</td></tr>
<tr><td colspan="9">Acknowledgement number (32)</td></tr>
<tr><td>Header length(4)</td><td>6 bits reserved</td><td>U R G</td><td>A C K</td><td>P S H</td><td>R S T</td><td>S Y N</td><td>F I N</td><td>window size (or) advertisement window (16)</td></tr>
<tr><td colspan="8">checksum (16) bits</td><td>urgent pointer (16)</td></tr>
<tr><td colspan="9">options (0-40) Bytes</td></tr>
<tr><td colspan="9">Data .
⋮</td></tr>
</table>

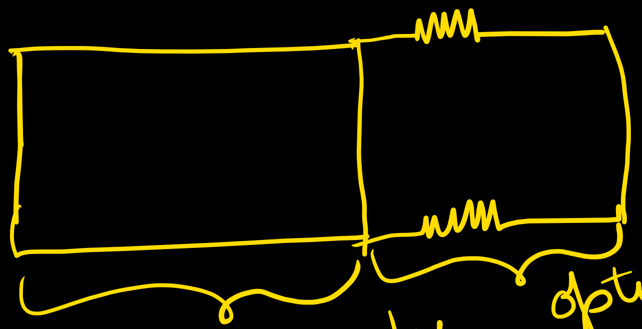
20 Bytes

ack no

ACK flag

different.

flags (1 bit each)



fixed header 20B

options (0-40B)

$\therefore$ header size $\rightarrow$ (20 - 60B).

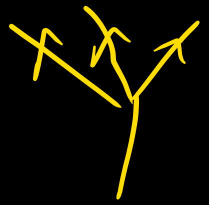
Header length field = 4 bits $\rightarrow$ max $\rightarrow$ 1111 $\rightarrow$ 15.

$$\frac{60B}{15} = 4 \rightarrow \text{scaling factor}$$

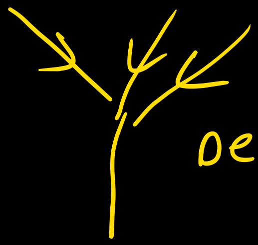
same as IP.

Source port and Destination port :

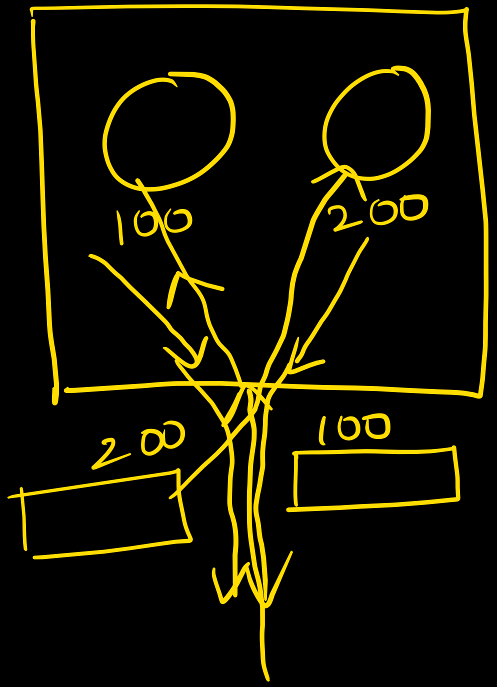
Every process is identified by a port number



mux



demux.



multiplexing.

Ports can do mux and demux.

www. google. com. → HTTP:

IP addr → DNS server.

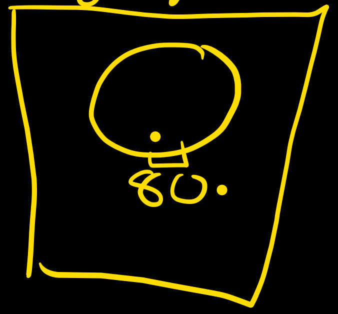
Port number

?

Every well known service will have fixed port numbers.

∴

google.com



Ex:

HTTP — 80

FTP — 21

Telnet — 23

SMTP — 25.

Port number ranges

0
|
1023 — well known ports → guidelines not rules.

1024
|
49151 — Reserved → for future

49152
|
65535 — general public → we can use.

client

~~80~~ confusing

49154 → example

8080

new server.

50001

Port number will be assigned by OS.
we can also change them.

Socket : IP + Port → identify connection.

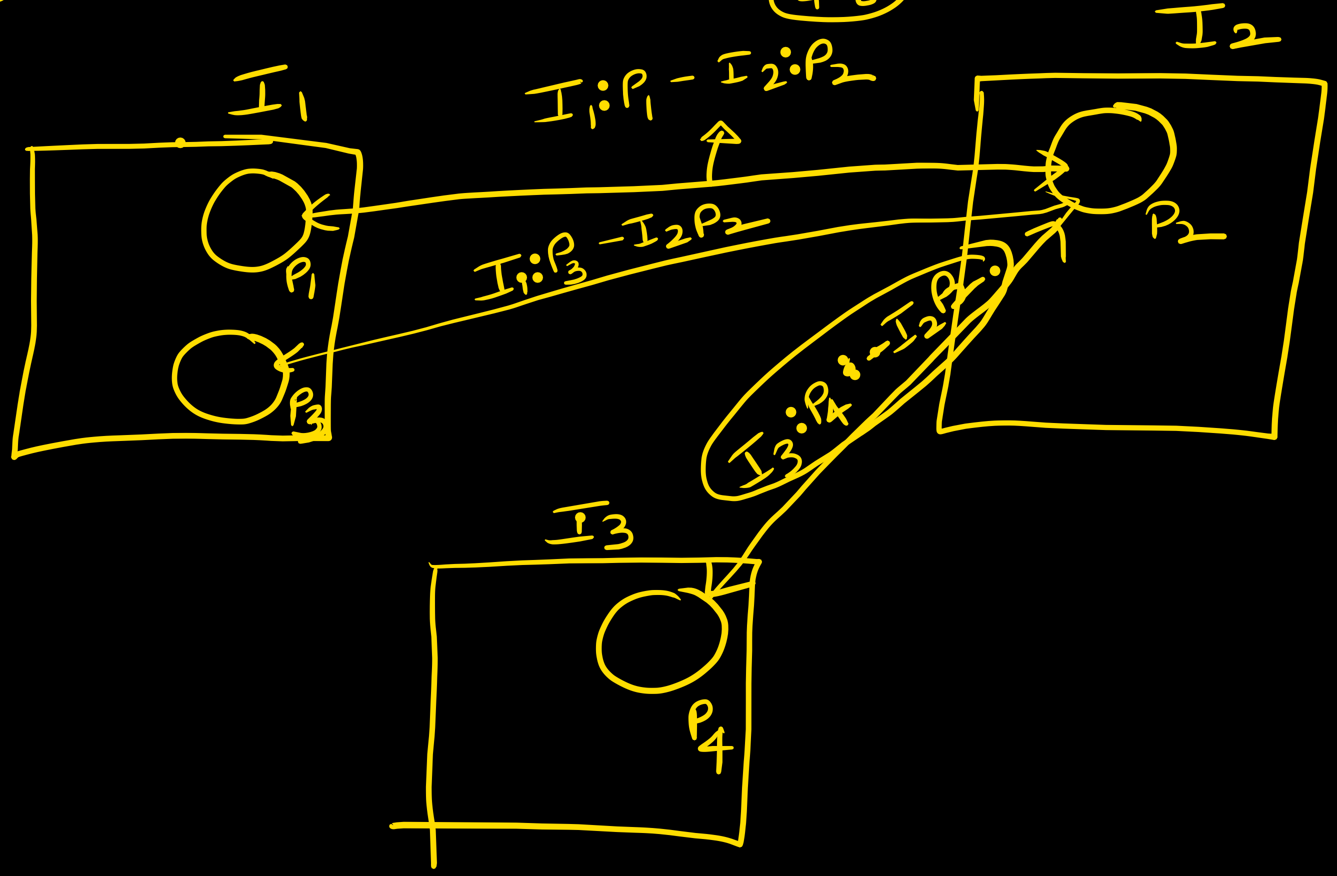
Socket: 48 bits.

IP: 32 bits

Port: 16 bits

$$\begin{array}{r} 32 \\ 16 \\ \hline 48 \end{array}$$

Socket ~~Port~~ number is 48 bit number.

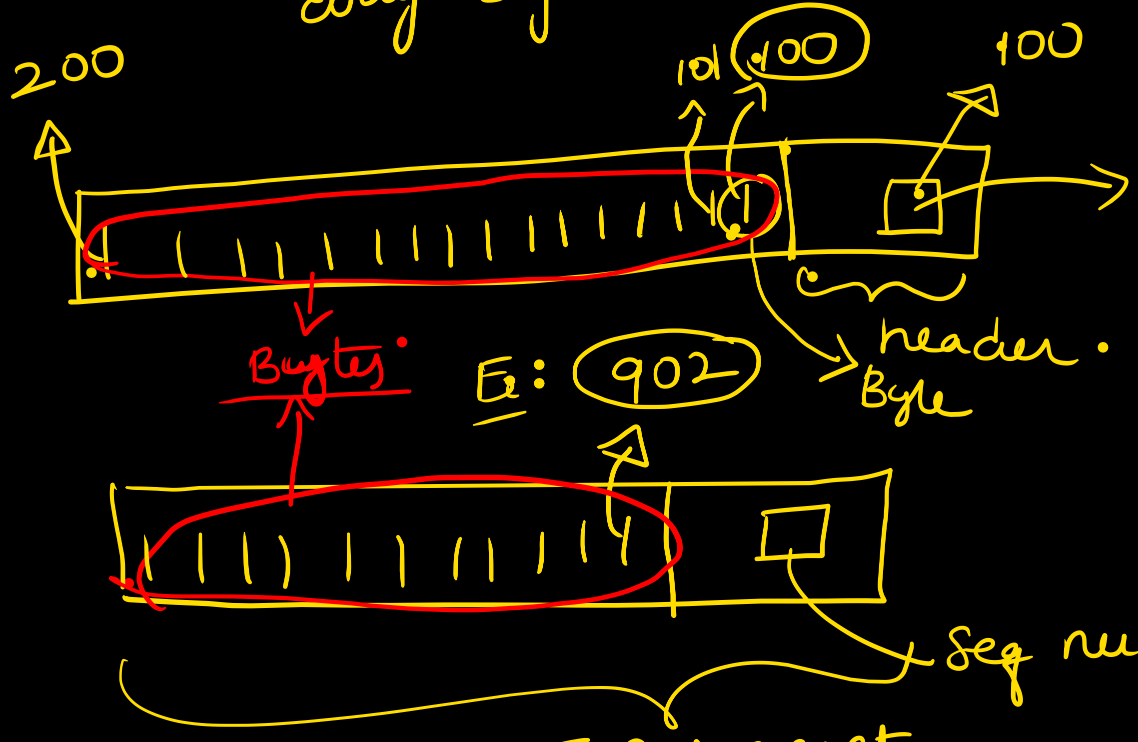


TL – Segment
NL – Datagram

Sequence number:

→ TCP is a byte stream protocol.

Every Byte is numbered.



Seq number field = First Byte sequence number.

Ex: 902

Seq numb = 902

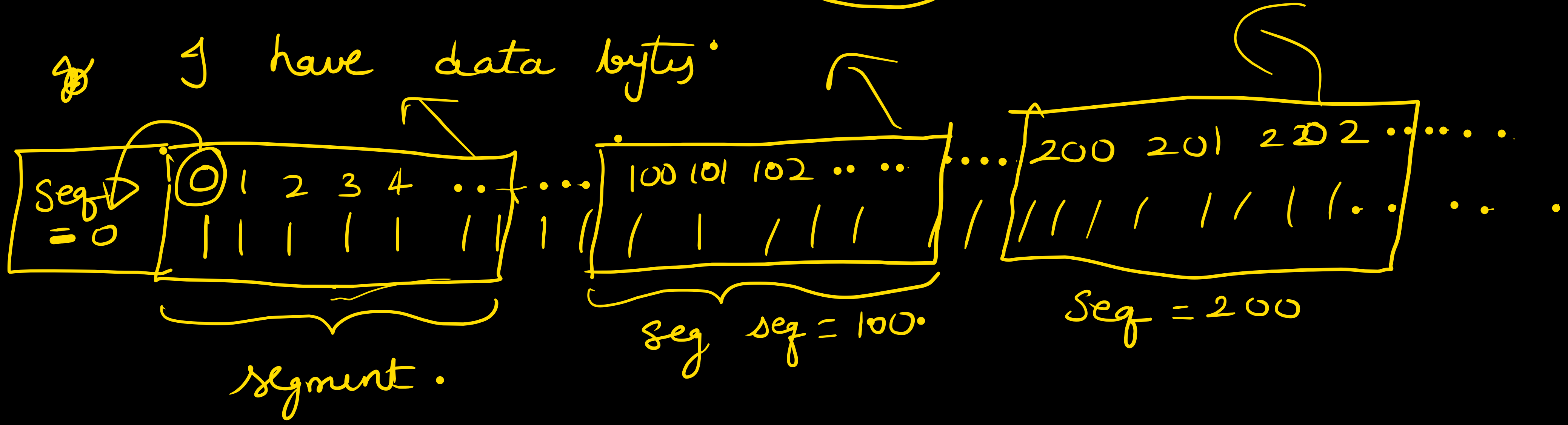
TCP segment

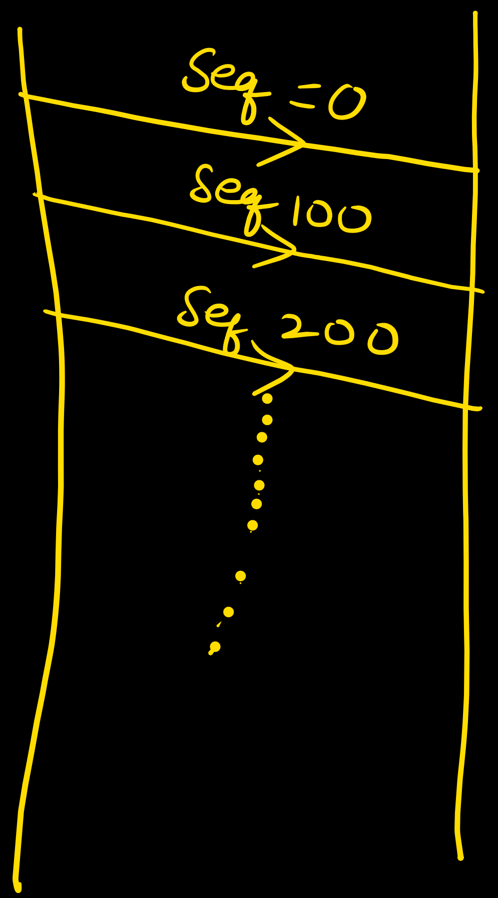
Sequence number → 32 bits.

So what are numbers possible

$$\underbrace{0 \quad 1 \quad 2 \quad \ldots\ldots \quad 2^{32}-1}_{2^{32} = 4G \rightarrow \text{numbers.}}$$

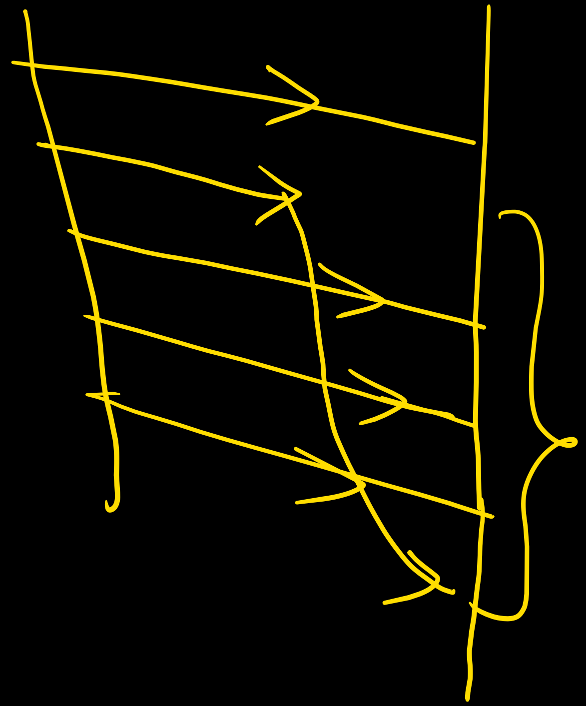
I have data bytes.



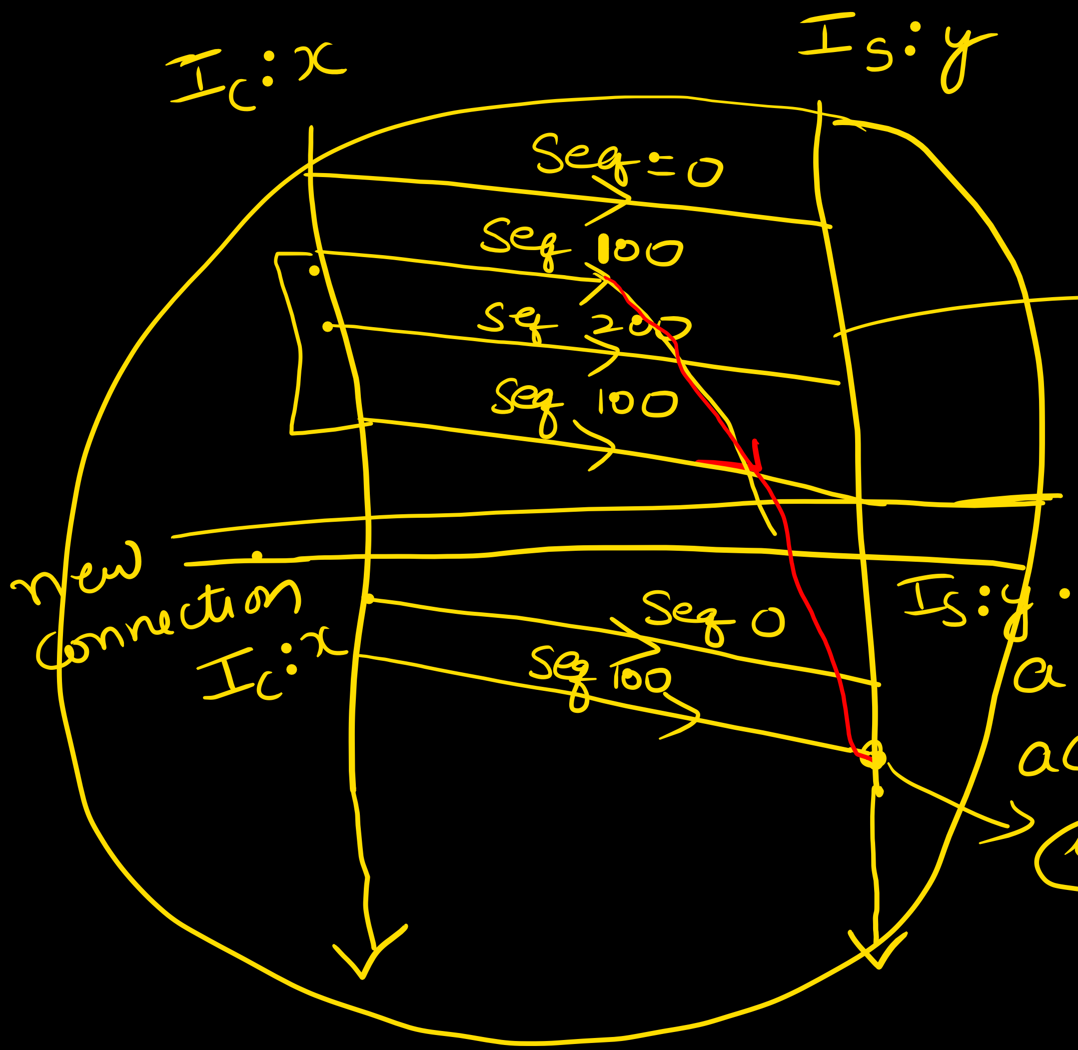


Seq =0
Seq 100
Seq 200

Rem: In Internet, lifetime of a packet is 3 min. which means in worst case the packet may come after 3 min.



3 min in worst case.



we are stating from '0' all the time.

From old connection

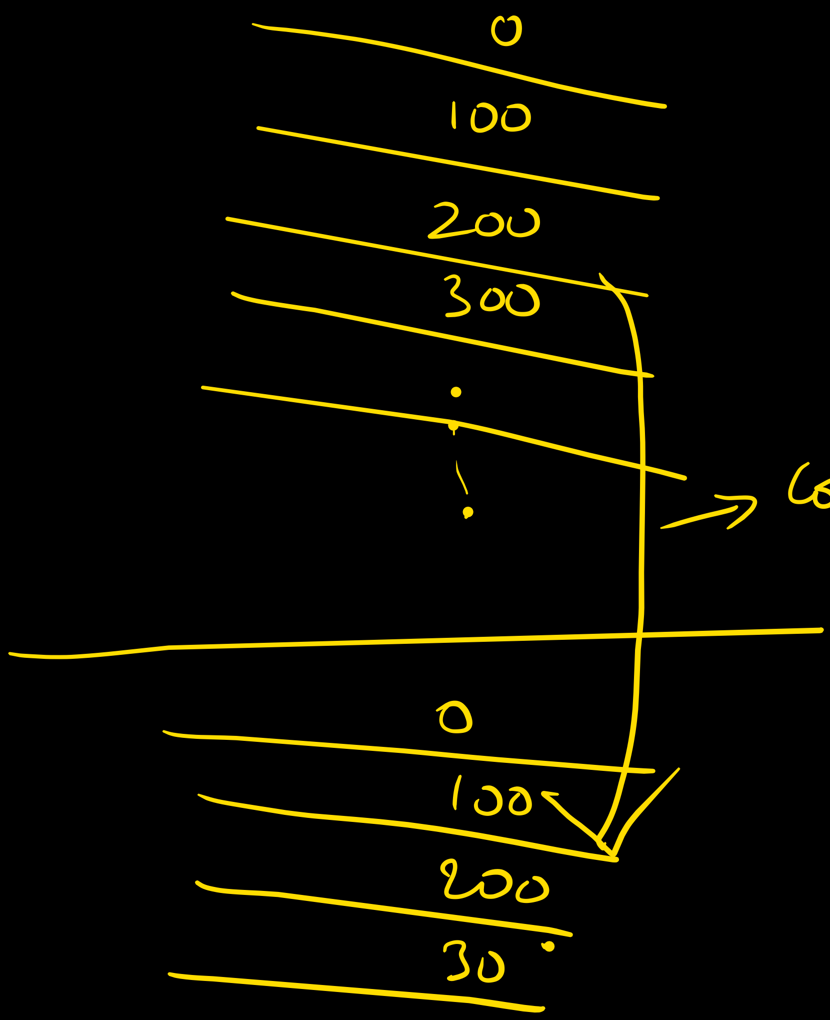
closed

a packet from old connection may be accepted as new packet.

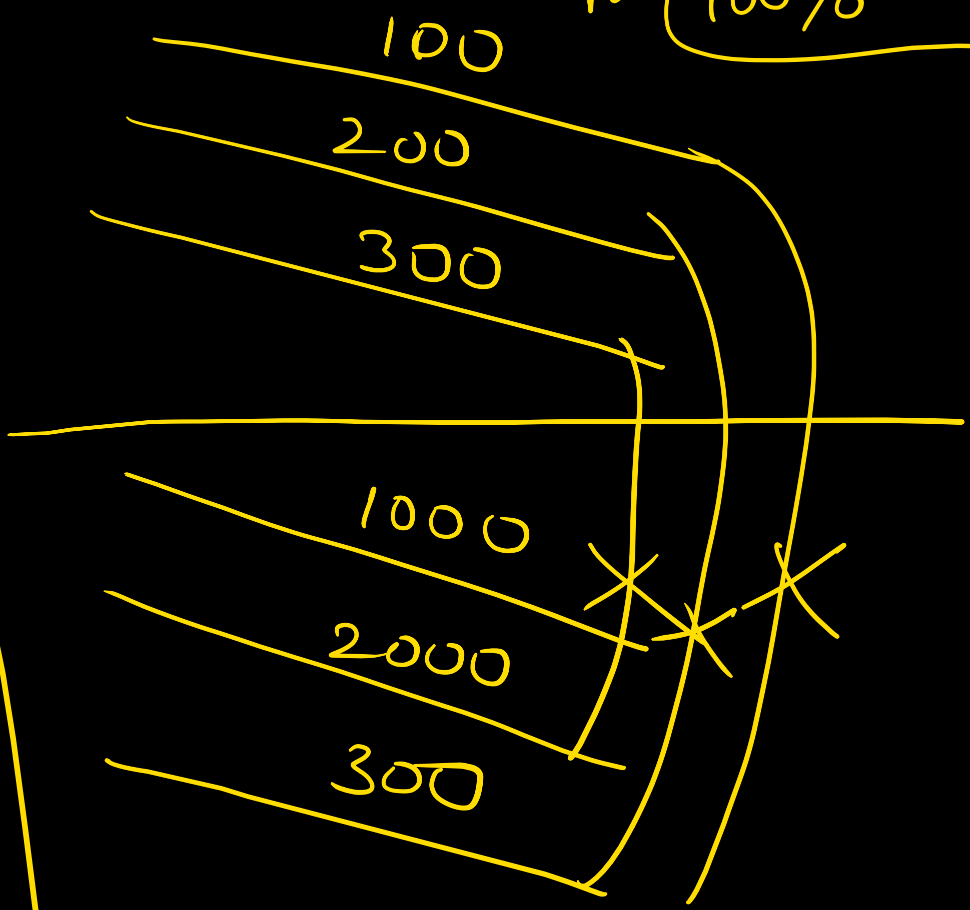
same numbers.

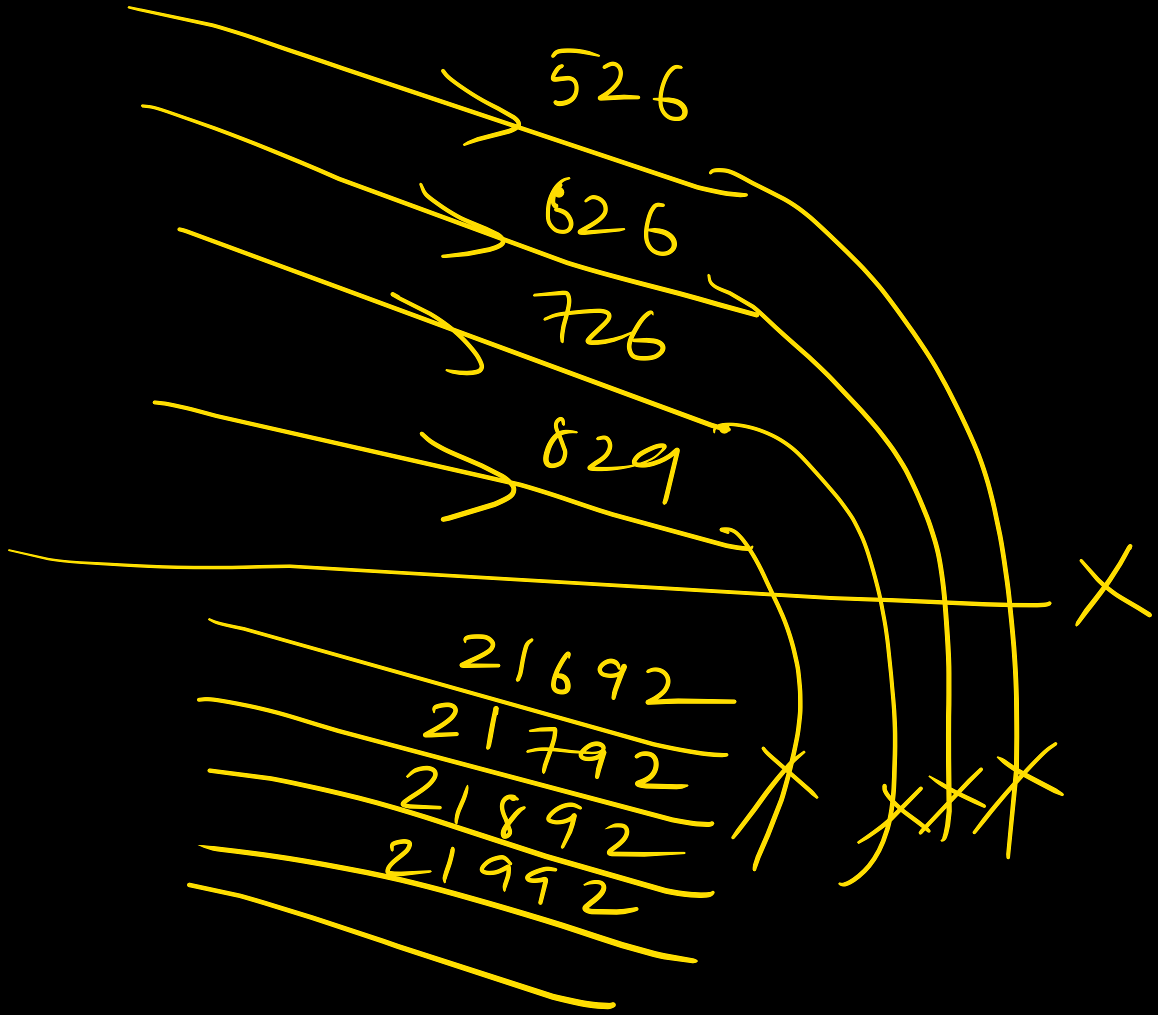
Dont start from '0' start from some random number.

not 100% safe



Come late confusion.





5 min break :

A TCP sender has data

seq = 0

0 1 2 3 . . . . – . . . $2^{32}-1$

all the seq numbers are over.

seq = 0

0 1 2 3 . . . . . . – . . . . . .

wrap ~~aroup~~ around.

Confusion?

Same connection

If two pacets arrive at the same time

Seq number is 32 bits i.e 2^{32} seq number are possible → 4 G

so we can send only 4 G Bytes i.e 4 GB. If we need to send more than 4 GB → then wrap around will occur.

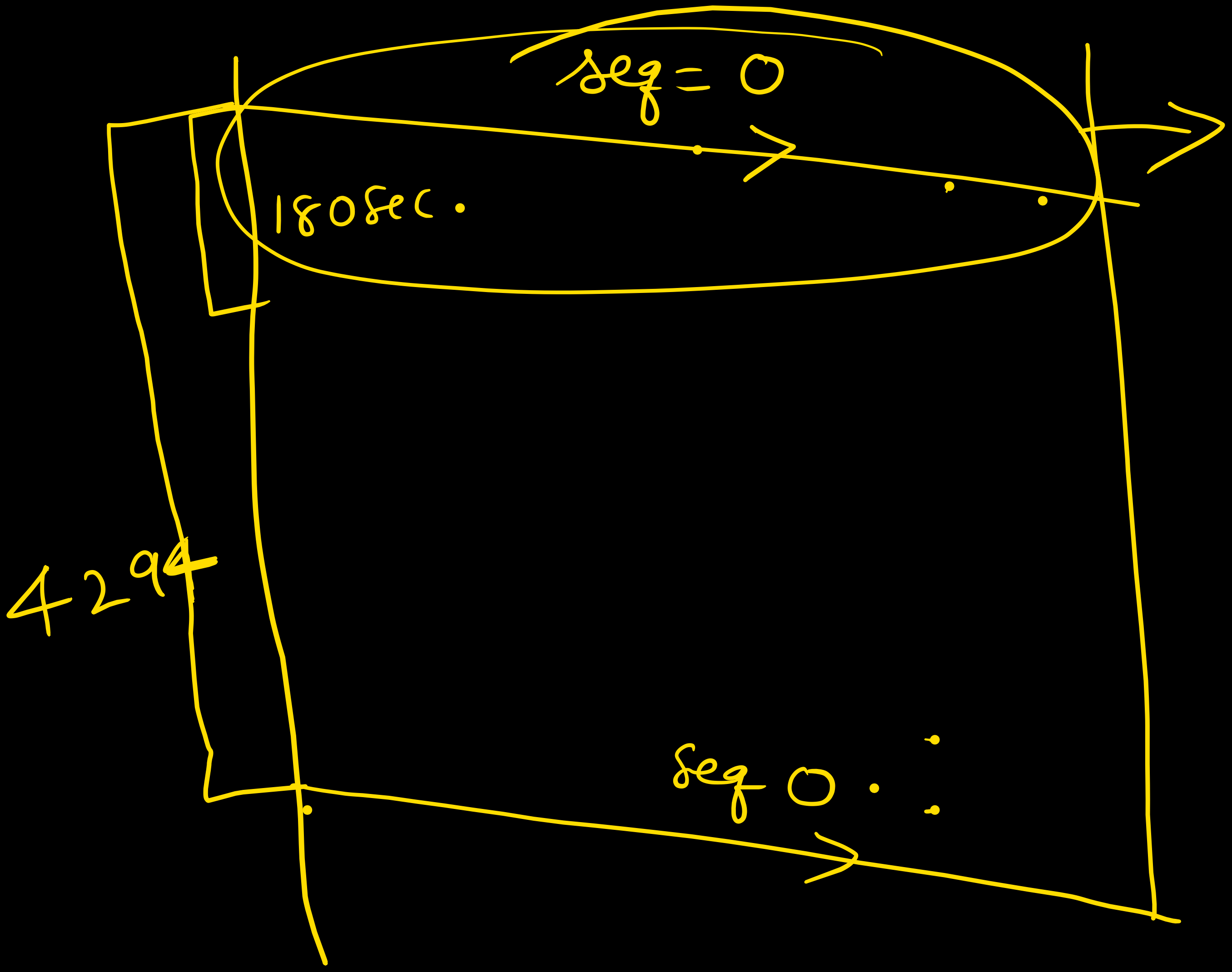
Let us say BW = 1 MBps → Bytes

After 4294 sec, we will reuse same seq number	1 MBps
But LT of a Packet is 3 min = 180 sec.	1 Sec — 10^6 Bytes
'0' $\xrightarrow{\text{alive}}$ 180 sec. after 4294 sec so no problem	1 Sec — 10^6 seq numbers. (Every Byte will take one seq number)

$\therefore$ 1 seq numb — $\frac{1}{10^6}$ sec

How many seq number we have → 2^{32}

$\therefore$ 2^{32} seq nums — $\frac{2^{32}}{10^6}$ sec = 4294.96 sec.


seq = 0
180sec.
4294
seq 0

If ~~WT~~ BW = 1 GBPS, then what is wrap around time?

~~wt o~~

Tommorow: