Silly window syndrome :-

In sliding window protocol either the sending application program creates data slowly or the receiving application program consumes data slowly or both.

* This reduces efficiency of ~~program~~ operation. TCP sends 1 byte of data, it means that a 41 byte datagram (20 byte TCP header + 20 byte IP header) transfers only 1 byte of user data. it indicates that we are using the capacity of network very inefficiently. This problem is called silly window syndrome.

Syndroume created by the Sender

* The sending TCP 1 byte at a time. The Application prog. writes 1 byte at a time.

solution :- Prevent the TCP from sending the data byte by byte. The sending TCP must be forced to wait and collect data to send in a larger block. How long should the sending TCP wait? Nagle find elegant solution.

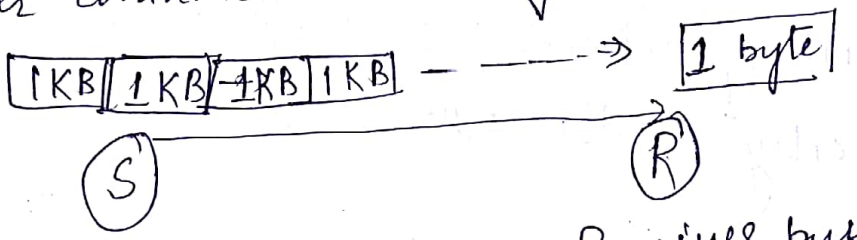
Nagle's Algo :-

1. The sending TCP sends the first piece of data it receives from the sending application program even if it is only 1 byte.

(2) After sending the first segment, the sending TC
accumulates data in the output buffer and wai
until either the receiving TCP sends an ACK or
(No data in flight)
until enough data has accumulated to fill a maximu
size segment. At this time, the sending TCP can send
the segment. Repeat for rest of transmission.

Syndrome created by the Receiver :-

1. Receiver consumes data slowly.



Receiver buffer is full.
Advertises window size = 0.
Now receiver reads 1 byte of data.
Advertises 1 byte data is free

sender sends 1 bytes of data

Process continues

Solution :- (2 solution)

Clark's solution - Send an acknowledgement as soon as the data arrive, but to announce a window size of 0 until either there is enough space to accomodate a MSS or until atleast half of the receiving buffer is empty.

yed Acknowledgement → When a segment arrives, it is not acknowledged immediately. The receiver waits until decent amount of space in its incoming buffer before acknowledging the arrived segment. The delayed ack prevents the sending TCP from sliding its window. This kills the syndrome.

Advantage - reduces traffic by less ACK

Disadvantage - Unnessary retransmission due to Timeout of unacknowledged segment.

(waiting for response data for piggybacking)

Congestion control :-

- A network is considered congested when too many packets try to access the same router's buffer, resulting in a large amount of packets being dropped.
- Impact of congestion may be temporary but if not handled it will be catastrophic.
- In case of network congestion TCP limits sender transmission rate to reduce load in the path between sender and receiver.
- TCP employs a window based scheme to control the transmission rate where size of the window directly impacts transmission.
- Acknowledgements are used to pace the transmission of packets by sender.

Congestion Window : (cnwd)

- It is a variable held by the TCP source for each connection.
- cnwd is set based on the perceived level of congestion in the network.

Max Window size = min (Congestion Window, Advertised Window)

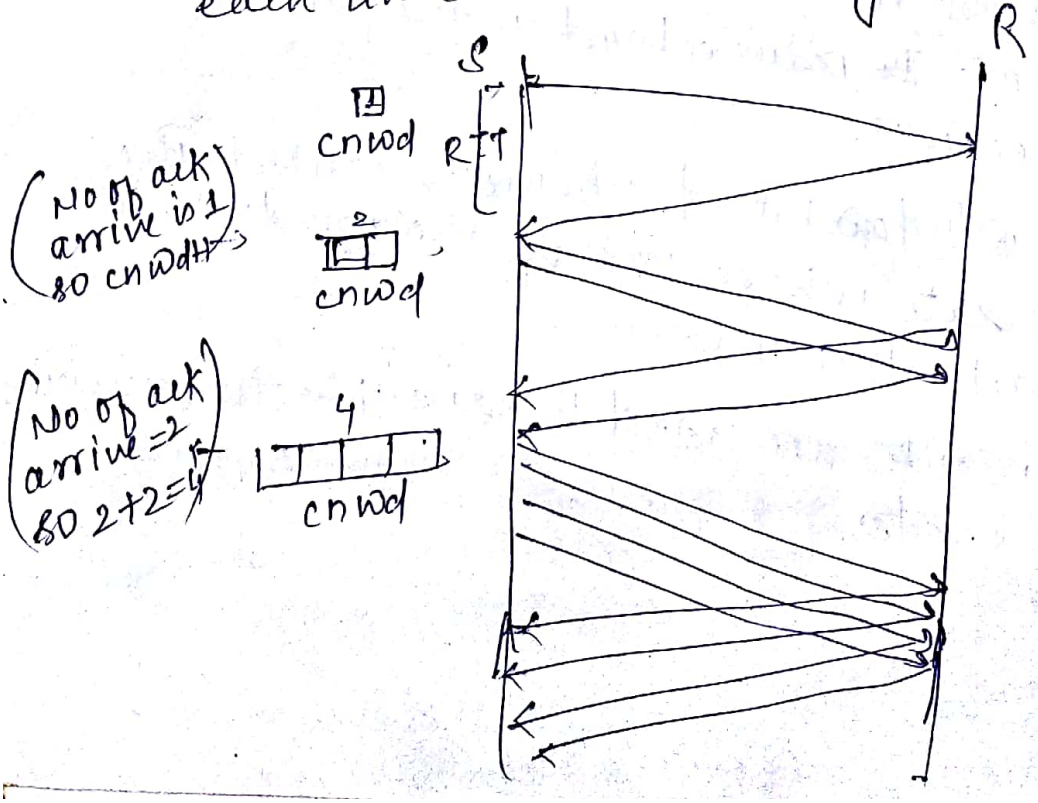
Congestion Policy :-

Handling congestion based on three phases :-

1. slow start
2. congestion avoidance
3. congestion detection.

1. slow start -

- size of the congestion window (cwnd) starts with one MSS. The size of window increase ~~one MSS~~ each time one acknowledgement arrives.



three segments are acknowledged cumulatively the size of cwnd increase by only 1 MSS, not 3 MSS.

- slow start can't continue indefinitely. There must be a threshold to stop this phase. The sender keeps track of variable named ssthresh when the window size in bytes reaches this threshold, slow start stops and next phase starts.

② Congestion Avoidance: Additive Increase.

- To avoid congestion one must slow down this exponential growth. Now in this phase increase the cwnd additively.
- Each time the whole window of segment is acknowledged, the size of the congestion window is increased by 1.
- The increase is based on RTT, not on the number of arrived Ack.

③ Congestion Detection: Multiplicative Decrease

- The only way a sender can guess the congestion has occured is the need to retransmit a segment. This is a major assumption in TCP.
- Retransmission can occur in two way —
  1. RTO Timer times out
  2. Three duplicate Acknowledgement

1. If timeout occur then there is a strong possi
of congestion.

a. $ssthreshold = \frac{\text{current window size}}{2}$

b. cwnd = 1

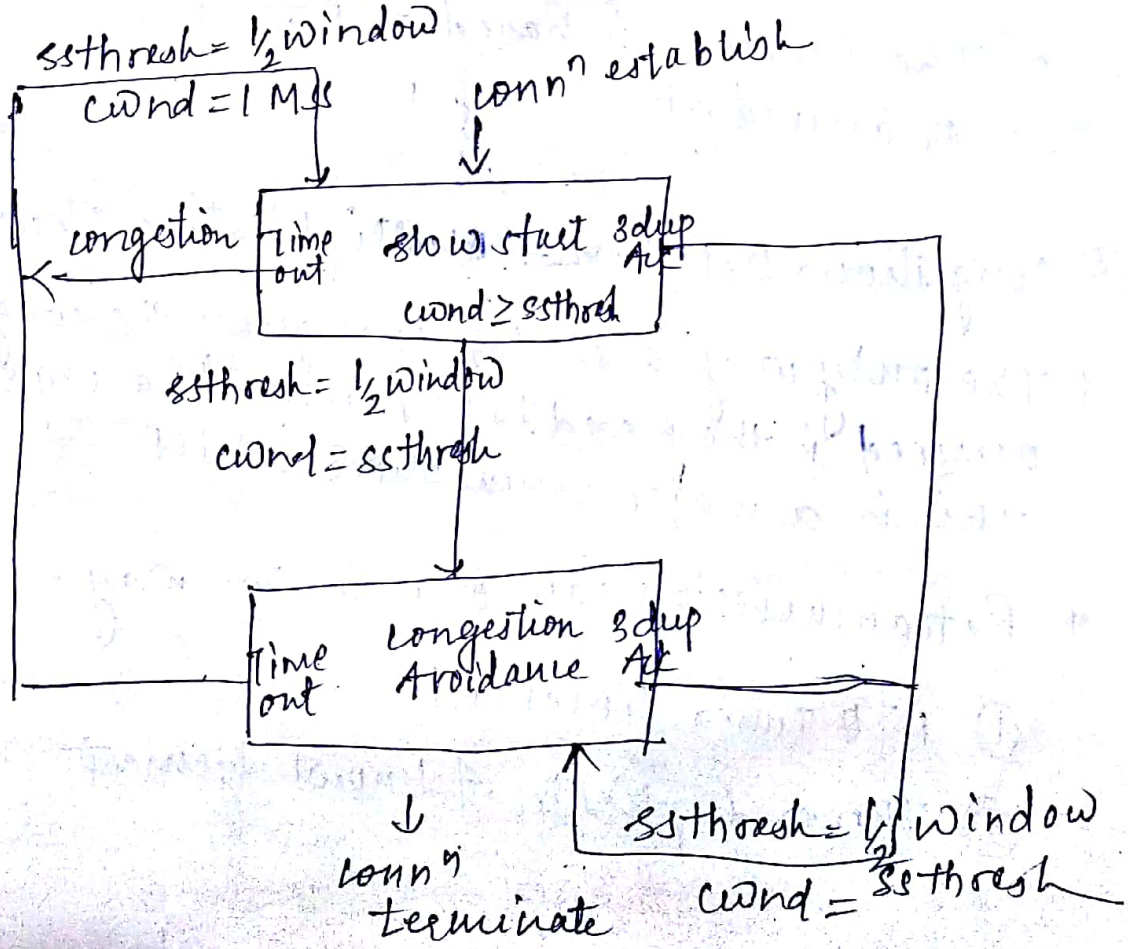
c. start slow start phase again.

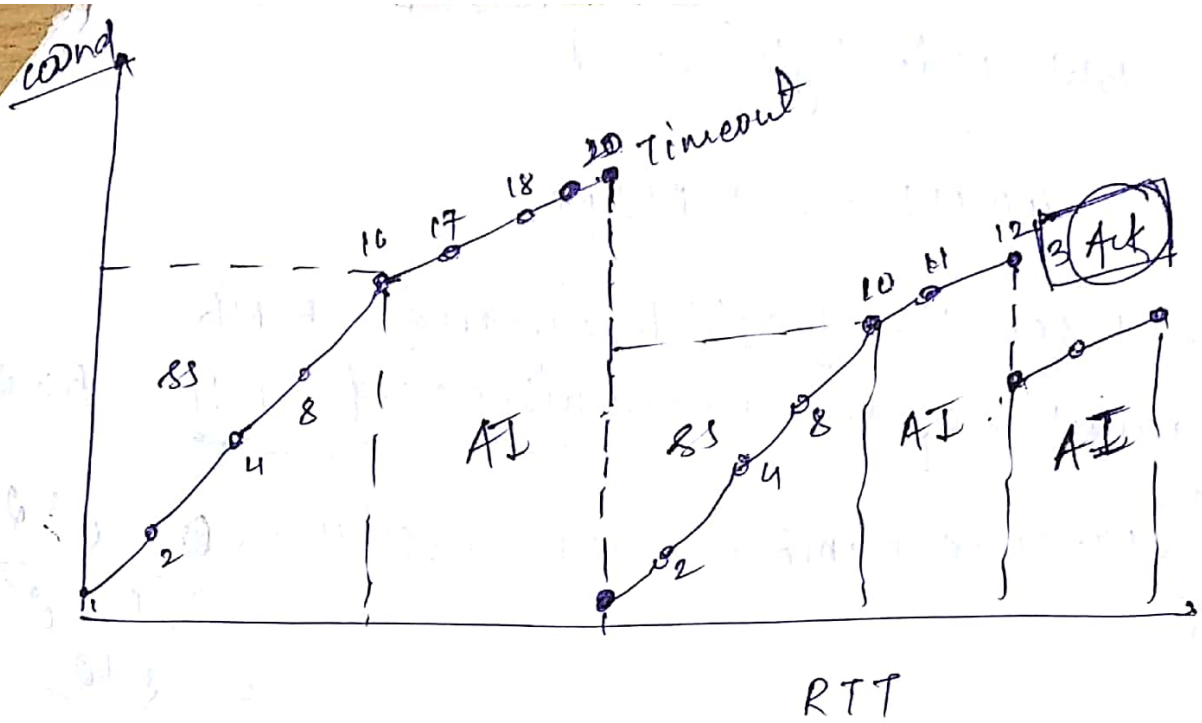
2. If three duplicate Ack are received, weaker possibility of congestion

a. $ssthreshold = \frac{\text{current window size}}{2}$

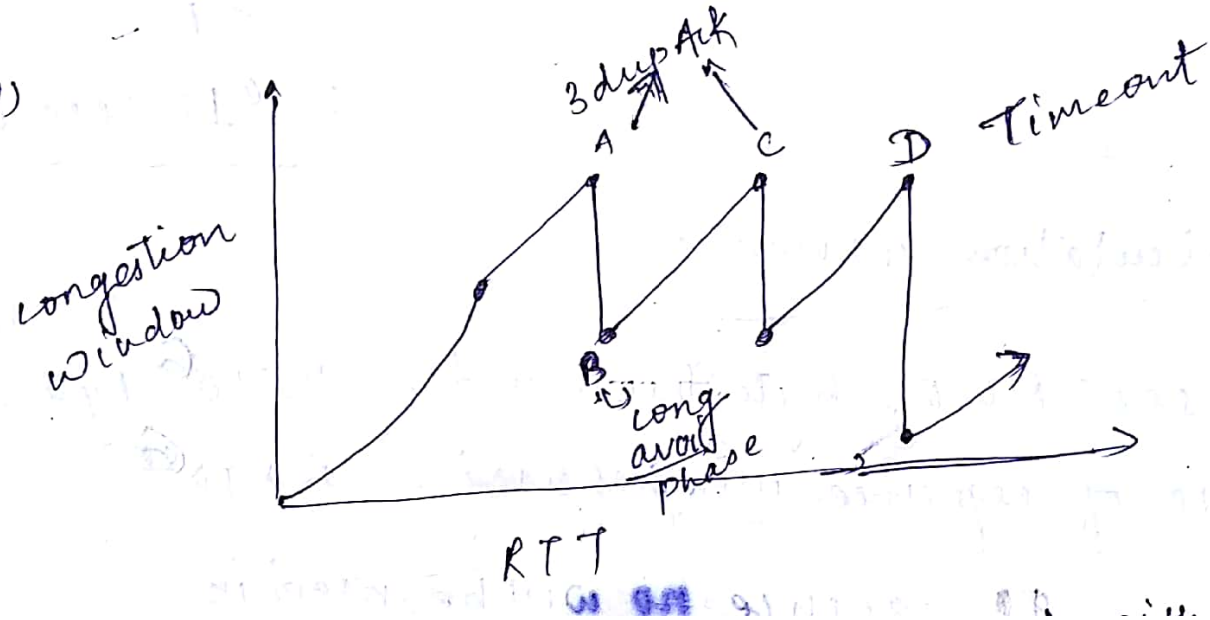
b. cwnd = ssthreshold

c. start congestion avoidance phase.





(a)



TCP timer
- → Retransmission
- → Persistence → to deal with 0 window size.
- → keepalive → To close long idle connection
- → Time Wait → during connection termination

Retransmission Timer

To calculate retransmission Timeout we need to calculate the round trip time.

Measured RTT (RTT_M) → RTT for a segment is amount of time between segment is sent & when Ack is received. Only one RTT measurement can be in progress at a time.

Smoothed RTT (RTT_S) - TCP maintains avg of measured RTT, it is weighted avg of RTT_M.

for first seq ($RTT_S = RTT_M$)

$$RTT_S = (1-\alpha) RTT_S + \alpha \cdot RTT_M$$

$$\alpha = 1/8, 0.125$$

$$RTT_S = 0.875\ RTT_S + 0.125 \cdot RTT_M$$

RTT Deviation (RTT_D) → variation in RTT.

After first measurement = $RTT_D = RTT_M/2$

After each measurement =

$$RTT_D = (1-\beta)\, RTT_D + \beta \times |RTT_S - RTT_M|$$

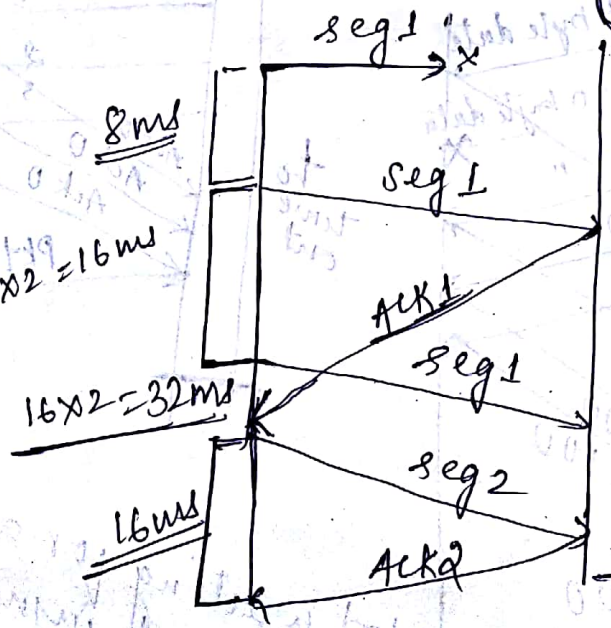
$\beta = 1/4 / 0.25$

Rule
① At any point of time, RTT_S is being estimated for only one of the transmitted but currently unacknowledged segment;
② When a segment is retransmitted, then the segment is not used for RTT calculation.

Retransmission Timeout

$$RTO = RTT_S + 4 \times RTT_D$$

Q.2 (first)
Q. For a given initial value of RTO as 8 ms, the first segment of TCP connection is sent. Retransmission of segment 1 is sent at timeout. RTT of this first segment in its second transmission attempt is 20 ms. Seg 2 is sent when the Ack of retransmitted segment is received successfully. RTT of segment 2 is 16 ms. Draw timeline diagram and calculate RTO.



(According to Karn Algo for calculation of RTT, retransmission acknowledgement should not be used)

seg 1 received + use exponential backoff ↓ double the timeout

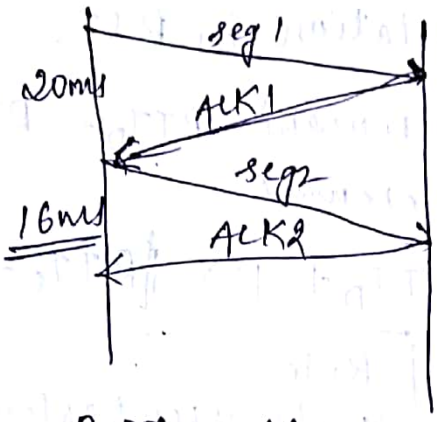
duplicate discard

seg 2 rcvd

$RTT_M = 16$ ms, $RTT_S = 16$ ms, $RTT_D = \frac{16}{2} = 8$ ms

$RTO = 16 + 4 \times 8 = 48$ ms

(Q.2)



seg1

$RTT_M = 20ms$

$RTT_S = 20ms$

$RTT_D = \frac{20}{2} = 10ms$

$RTO = 20 + 4 \times 10 = 60ms$

seg2 → $RTT_M = 16ms$

→ $RTT_S = 0.875 \times 20 + 0.125 \times 16$

$= 19.5ms$

→ $RTT_D = 0.75 \times 10 + 0.25 \times |19.5 - 16|$

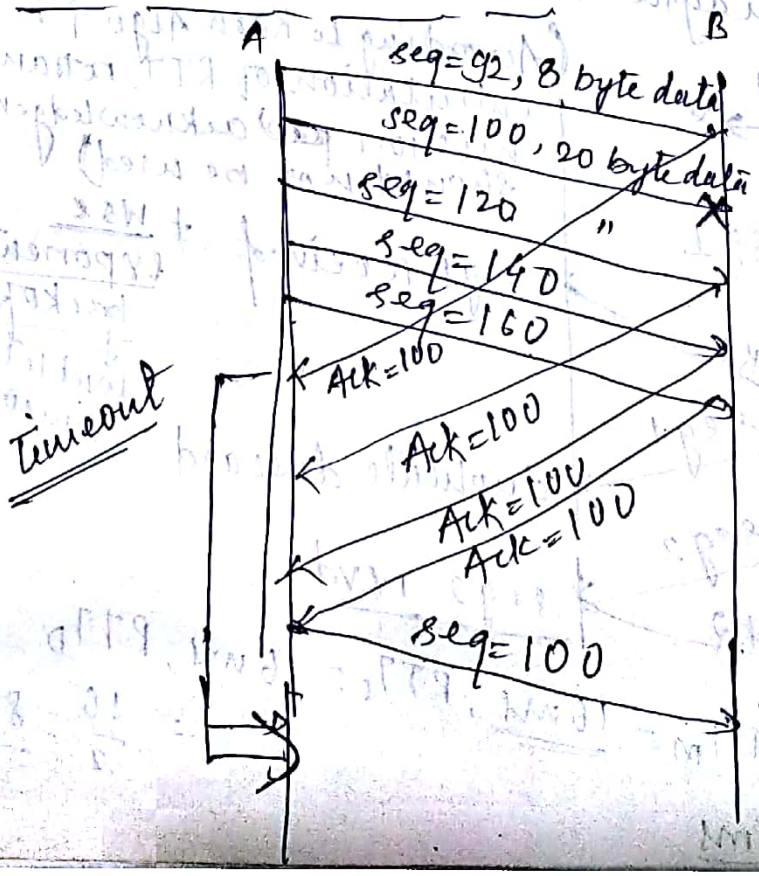
$= 0.75 \times 10 + 0.25 \times 3.5$

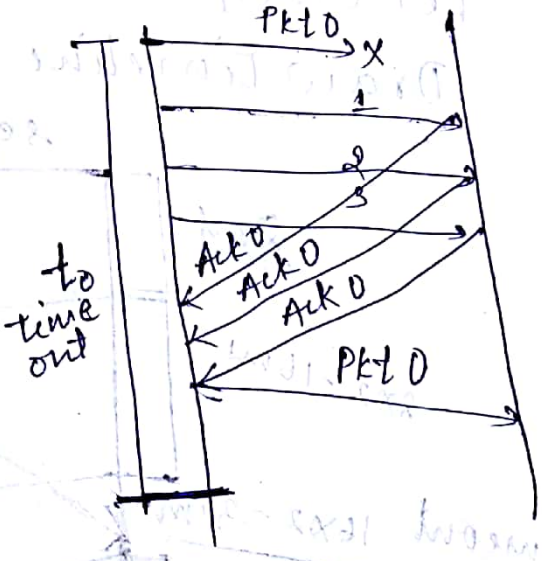
$= 8.375ms$

→ $RTO = 19.5 + 4 \times 8.375$

$= 53ms$

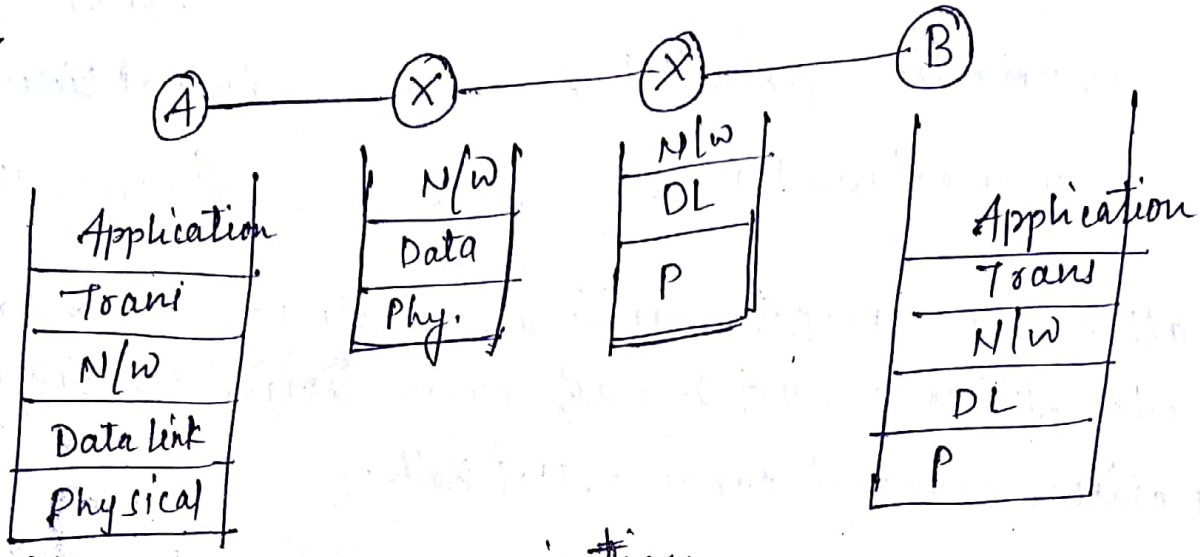
Fast Retransmission





Not waiting for expiration of timeout immediately sending data packet

Network layer



→ Host to host communication

→ Forwarding - When a packet arrives at a router's input link, the router must move the packet to appropriate output link.

→ Routing - The N/w layer must determine the route or path taken by the packets as they flow from sender to receiver. The algorithms that calculate this path are called routing algorithms.

→ Forwarding table - Every router has a table, by examining value of header field it decides where to forward.

→ Services Provided by Network Layer

① Best effort service → Timing b/w packets are not guaranteed, packets are not guaranteed to be received in order, nor the eventual delivery of transmitted packet is guaranteed.

Virtual Circuit and datagram Networks :

TCP — Transport — connection oriented — — — → Virtual Circuit

N/W — connectionless — — — → Datagram N/W

* In all major computer N/W architecture the N/W layer provides either a host to host connection less service or connection oriented service, Not both.

* **Virtual Circuit Networks :-**

Path (b/w source and destination)

↓

VC Number, identifier

↓

entries in forwarding table

Phases

VC setup → N/W layer determine path between senders and receives. The Network layer also determine VC number for each link along the path. Finally N/W layer adds entry in the forwarding table in each router along the path.

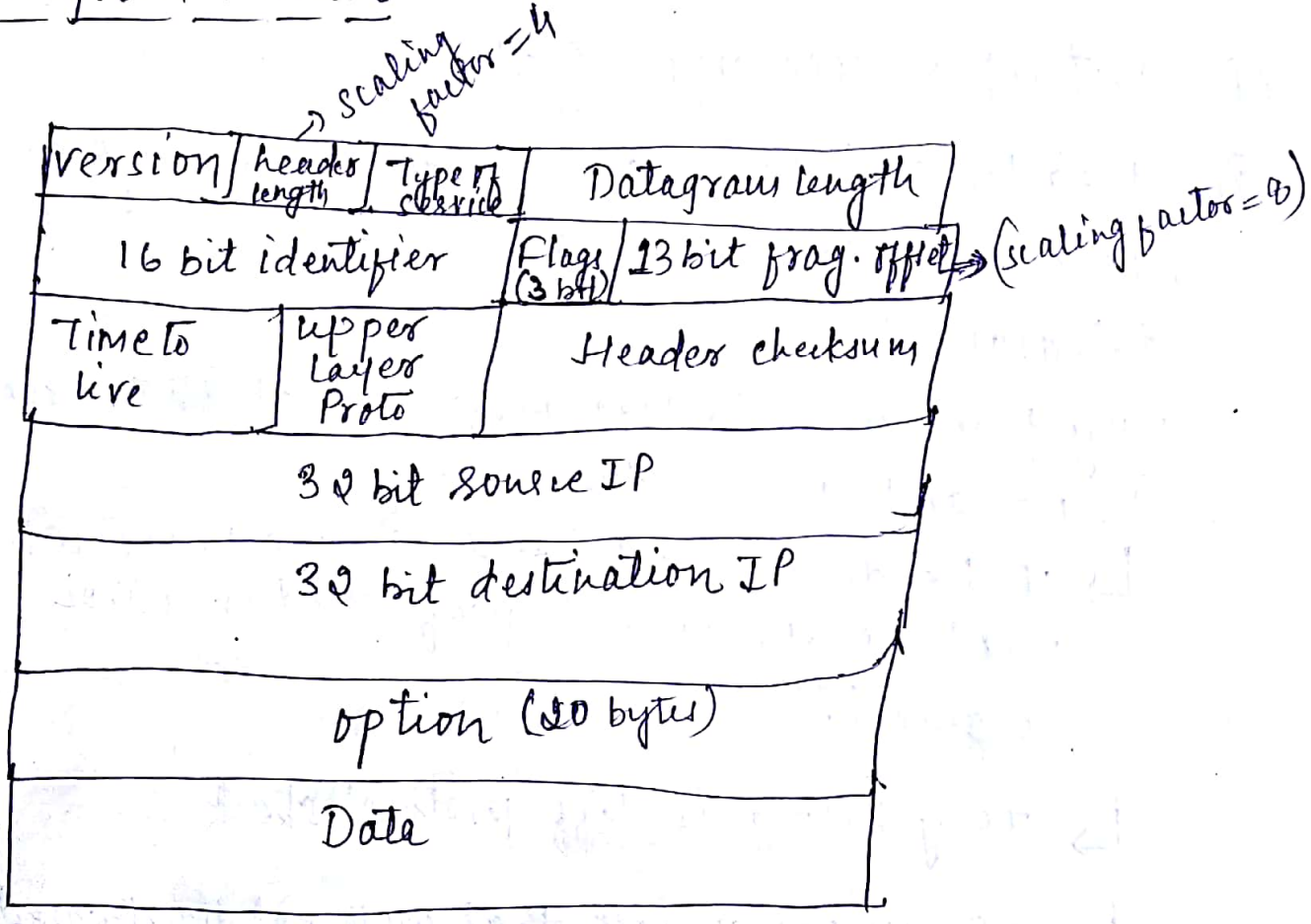
Data transfer –

VC teardown :- senders inform N/W layer. The N/W layer will then typically inform the end system. update the forwarding table.

Datagram Network :-

Each router uses packet's destination address to forward the packet. The router uses the packet destination address to lookup the appropriate output link interface in the forwarding table.

Datagram Format



(1) version Number - IPv4 or IPv6 (4 bit)

(2) Header length - 4 bits, To determine header length.

(3) Type of service → low delay
high throughput
reliability

(4) Datagram length - total length of IP datagram (header + data)

⑤ Identifier, flags, fragmentation → used in IP fragmentation.

⑥ Time-to-live - It is included to ensure that datagram do not circulate forever, in the N/w. This field is decremented by one each time the datagram is processed by routers.

⑦ Protocol → TCP/UDP (6/17)

⑧ Header Checksum - The header checksum aids a router in detecting bit error in a received IP datagram.

→ Why does error checking performed at Transport Layer and Network Layer?

↳ IP header is checksumed at N/w layer while TCP/UDP checksum is performed on whole segment.

↳ They belong to diff protocol stack.

↳ IP can carry data that will not be passed to TCP/UDP.

⑨ source and destination IP address

⑩ option:-

⑪ Data :- TCP segment as well as ICMP message.

Fragmentation

→ Not all link layer protocol can carry network layer packets of the same size.

→ Ethernet carries 1500 bytes, whereas frames of some wide area links can carry no more than 576 bytes.

→ The maximum amount of data that a link layer frame can carry is called Maximum transmission Unit (MTU). (diff than MSS)

→ Problem → How you are going to squeeze the over sized IP datagram into the payload field of the link layer frame!

→ solution → Fragment the data in the IP datagram into two or more smaller IP datagram, encapsulate each of these smaller IP datagram in a seperate link layer frame, and send these frames over the outgoing link. Each of these smaller datagrams is reffered to as fragment.

→ Fragments needs to reassembled before they reach the transport layer at the destination. Reassembly is done at receiver.

Header used in Fragmentation

Identification – A datagram when created, the sending host stamps the datagram with an identification number. When router needs to fragment a datagram,

each resulting datagram have same datagram identification numbers.

Flags

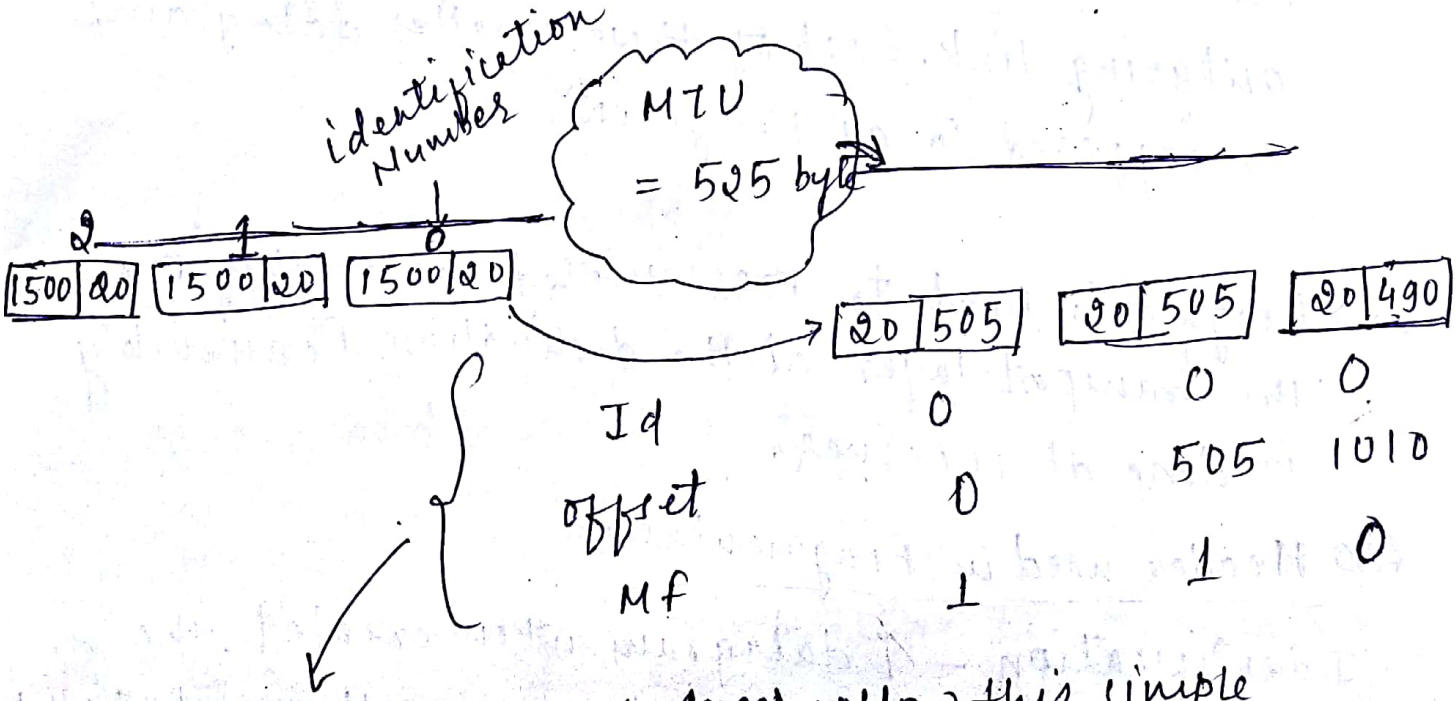
DF – don't fragment – datagram is not permitted to fragment.

MF (more fragment) – (Used in reassembly)

MF = 1 → there is more fragment ahead of current fragment.

MF = 0 → Last fragment, no more fragment ahead.

offset → This field is used to know how many data bytes are ahead of this fragment. in a particular fragment.



No, we don't follow this simple mechanism.

20	504

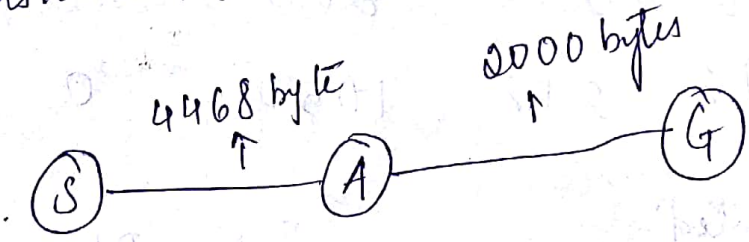
20	504

20	504

Id	0	0	0
offset	0	$\frac{504}{8} = 63$	$63 + \frac{504}{8} = 126$
MF	1	1	0
DF	0	0	0

Dividing the offset by 8 allows it to fit in 13 bits instead of 16, this means every packet but the last must contain a number of data bytes that is a multiple of 8.

(Q) An IP datagram carrying 12000 bytes of data with Packet ID = 818, must be sent from the node S to G. The link SA support MTU of 4468 bytes and the link AG supports MTU of 2000 bytes. How many fragments would be generated? Show total length, Packet id, offset, DF and MF of the IP header in each packet transmitted over the link.



Soln For link SA

	Length	PktId	offset	DF	MF
F1:	4448	218	0	0	1
F2:	4448	218	556	0	1
F3:	3104	218	1112	0	0

Now F1 is fragmented in link AG, Packets are:-

Length	PktId	offset	DF	MF
1976	218	0	0	1
1976	218	247	0	1
496	218	494	0	1

F2 is fragment in AG

Length	PktId	offset	DF	MF
1976	218	556	0	1
1976	218	803	0	1
496	218	1050	0	1

F3 is fragmented:-

Length	PktId	offset	DF	MF
1976	218	1112	0	1
1128	218	1359	0	0

The link state Routing Algorithm

* In link state algorithm, the network topology and all link costs are known.
* In general cost could be a function of distance bandwidth, average traffic, communication cost, mean queue length, measured delay, router processing speed etc.
* **Dijkstra's Algorithm:-** computes the least cost path from one node to all other nodes in the network. (single source shortest path).

$D(v)$: cost of the least cost path from the source node to destination v

$p(v)$: previous node along the current least cost path from the source to v.

N' : subset of nodes v

Algo → Initialization

$N' = \{u\}$

for all nodes v

if v is a neighbour of u

$D(v) = c(u,v)$

else

$D(v) = \infty$

loop

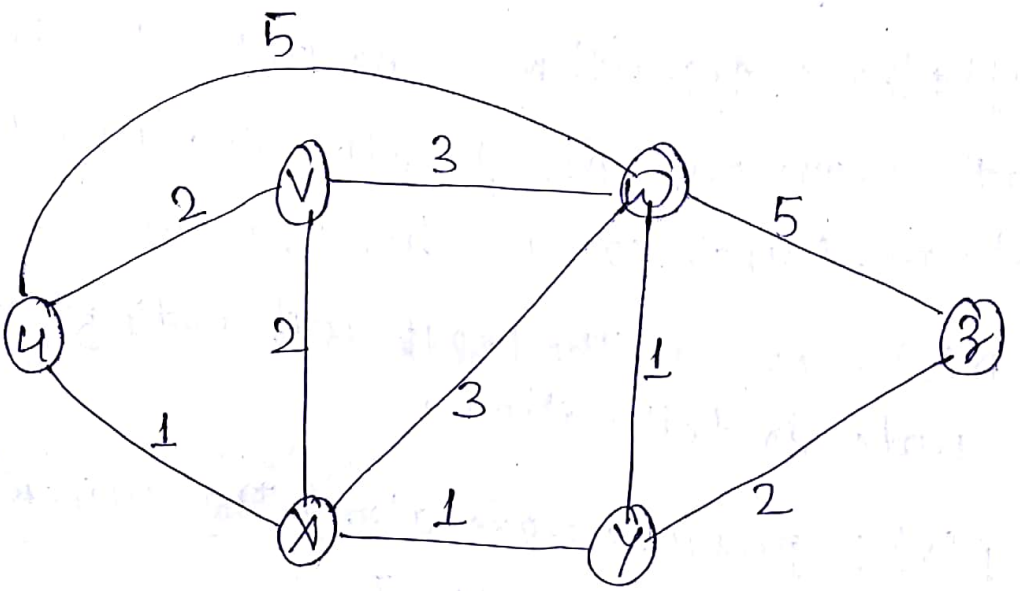
find w not in N'

add w to N'

update D(v) for each neighbour v of w and not in N'

$$D(v) = \min(D(v), D(w) + c(w,v))$$

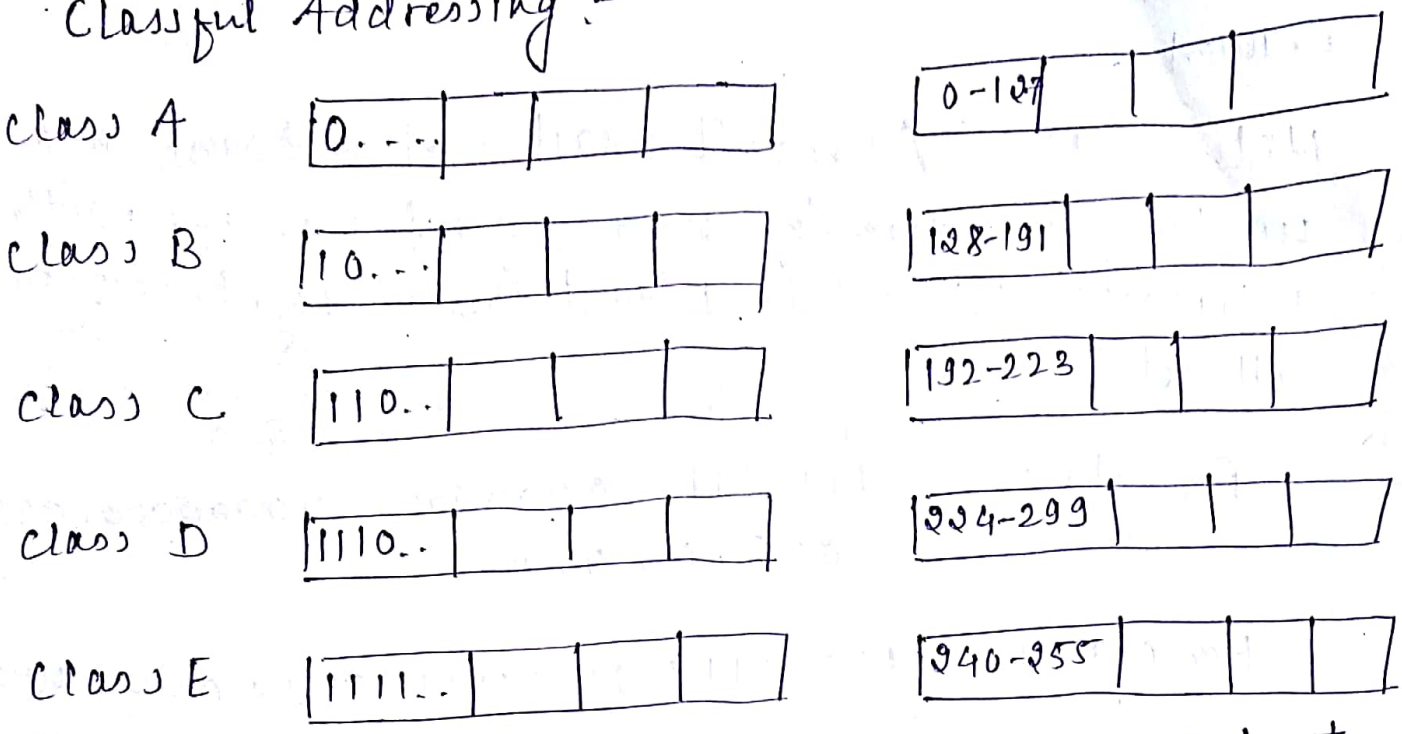
until N'=N



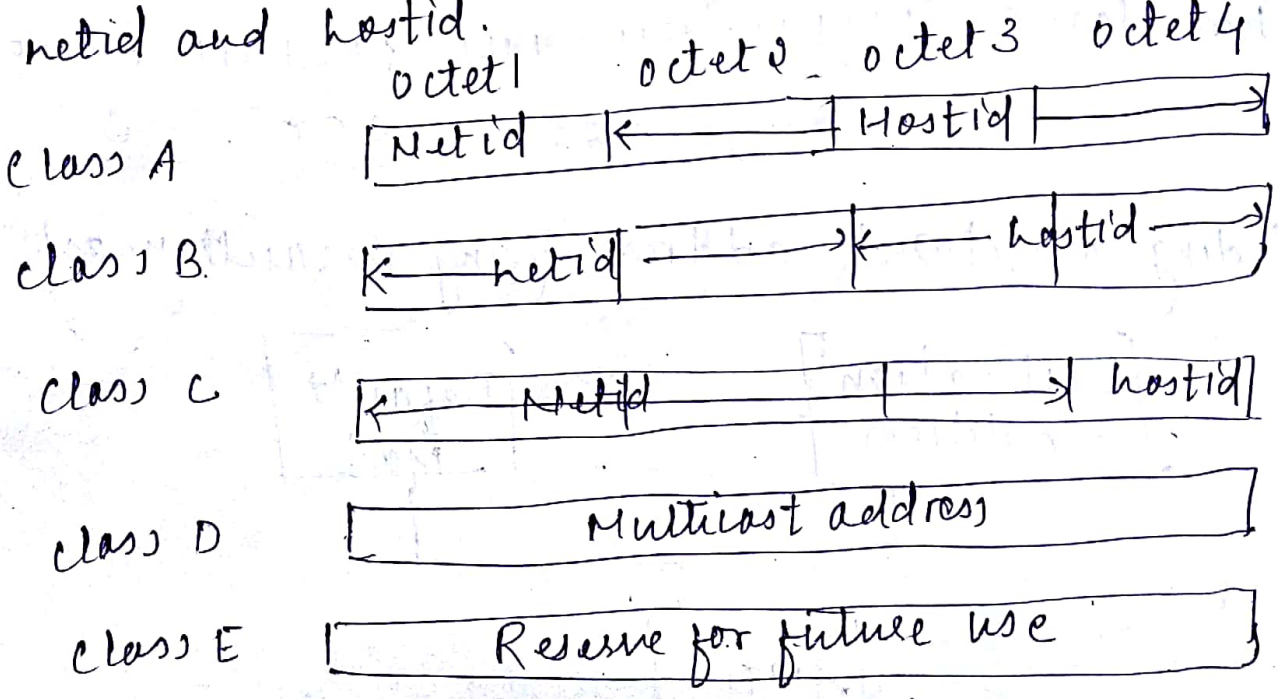
N'	D(v), p(v)	D(w), p(w)	D(x), p(x)	D(y), p(y)	D(z), p(z)
u	2, u	5, u	1, u	∞	∞
ux	2, u	4, x		2, x	∞
uxy	2, u	3, y			4, y
uxyv		3, y			4, y
uxyvw					4, y
uxyvwy					

IP Addressing

* Classful Addressing :-



* Netid and Hostid :- classes A, B and C are divided into netid and hostid.



Class A → 2^7 ~~addresses~~ blocks, each ~~address~~ block has 2^{24} addresses

Class B → 2^{14} ~~addresses~~ blocks, each block has 2^{16} addresses

Class C → 2^{21} blocks, each block has 2^8 addresses

Network Address – It is used in routing a packet to its destination network. It is the identifier of the network.

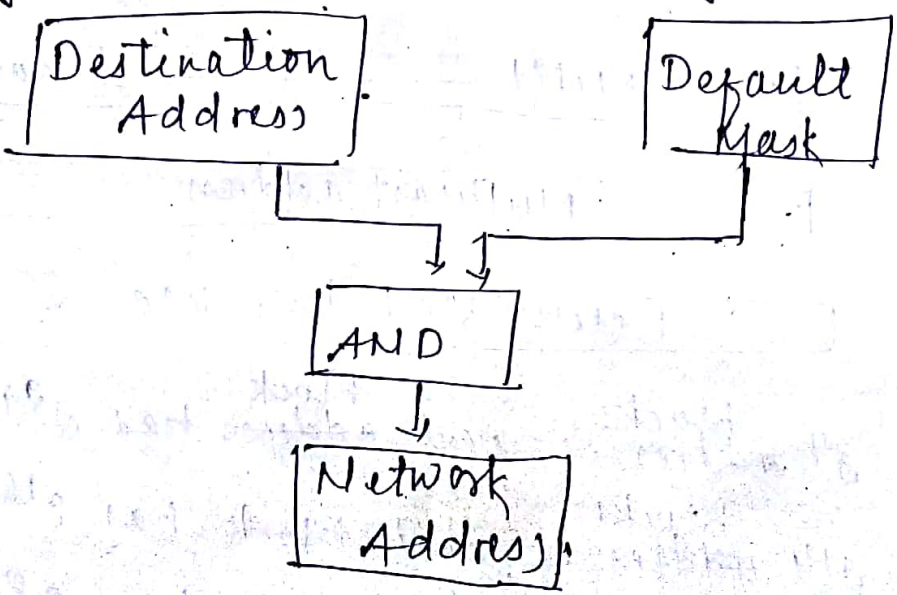
Network Mask / Default mask – A network mask in classful addressing is a 32 bit number with n leftmost bits set to 1 and (32-n) rightmost bits all set to 0s.

For class A 11111111 00000000 00000000 00000000
255.0.0.0

For class B 11111111 11111111 0000 0000 00 00000
255 . 255 . 0 . 0

For class C 11111111 11111111 11111111 00000000
255 . 255 . 255 . 0

Finding a network address using Default mask



Destination Address	201.	24.	67.	32
Default mask	255.	255.	255.	0
Network address	201.	24.	67.	0

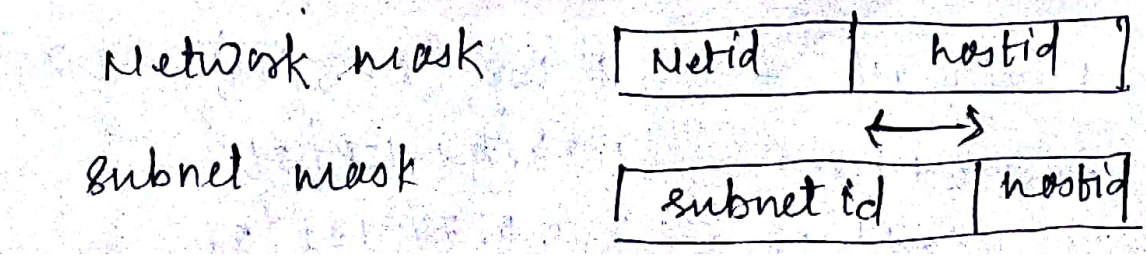
Subnetting → A large network is divided into several subnetwork for better security and management. In subnetting, a network is divided into smaller subnet with each subnet having its own subnetwork address.

Advantages :- (i) It improves security.
(ii) Restructuring of internal network is possible without affecting other network.
(iii) Maintance and administration is very simple.

Disadvantage - Identification problem contain 4 steps :-

(a) Determine network
(b) Determine subnet
(c) Host
(d) Process

Subnet Mask :- When we divide a network into subnetwork we need to create subnet mask for each subnetwork. For this purpose we borrow bits from hostid part.



Dest. Address	141 . 14 . 120 . 77
subnet mask	255 . 255 . 192 . 0
Subnet Address	141 . 14 . 64 . 0

(Q) An organization is granted a block of addresses with the begining address to use in its three subnets:-

1. one subblock of 120 addresses (7 bit)
2. one subblock of 60 addresses (6 bits)
3. one subblock of 10 addresses (4)

Begining address = 14.24.74.0/24 $32-24=8$

Sol → There are 2^{32-24} = 256 addresses in this block.

First address - 14.24.74.0/24

Last address - 14.24.74.255/24

a. The number of address in first block is not power of 2. We allocate 128 addresses. The first address can be used as network address and the last as broadcast address. There are remaining 126 addresses which can be assigned to host.

$$\text{subnet mask} = 24 + \log_2 (256/128)$$
$$= 24 + 1 = 25$$

The first address in this block 14.24.74.0/25

Last address 14.24.74.127/25

b. subnet mask = $24 + \log_2 (256/64) = 26$

First address = 14.24.74.128/26

Last address = 14.24.74.191/26

(c) subnet mask = $24 + \log_2(256/16) = 28$

First address = 14.24.74.192/28

Last address = 14.24.74.207/28.

* Supernetting :- Several networks are combined to create a supernetwork. For e.g. an organization can combine several class C blocks to create large range of addresses.

→ Supernet mask : For combining the network we create supernet mask, in this we borrow bits from netid. The length of the supernetid can found using formula

$$n_{super} = n - \log_2 C$$

in which n_{super} defines the length of supernetid in bits and C defines the number of class C blocks that are combined.

Problem arises due to supernetting :-

(i) The number of blocks to combine needs to be a power of 2.

(ii) It complicates the routing of packets.

(d) Determine the supernet mask if 16 class C blocks are combined to make supernetwork.

Soln → As we need 16 blocks, 4 bits are required to be borrowed from netid. To aggrigate make rest bits 0.

default mask of class c

11111111 11111111 11111111 00000000

11111111 11111111 11111111 00000000

supernet mask 255.255.240.0

(2) Perform supernetting and determine ~~subnet~~ super net mask and supernet Address.

205.100.0.0

205.100.1.0

205.100.2.0

205.100.3.0

variable bit

11001101.01100100.00000000.00000000

11001101.01100100.00000001.00000000

11001101.01100100.00000010.00000000

11001101.01100100.00000011.00000000

~~subnet~~ supernet mask will be bits which are not variable. Hence

supernet mask = 255.255.252.0

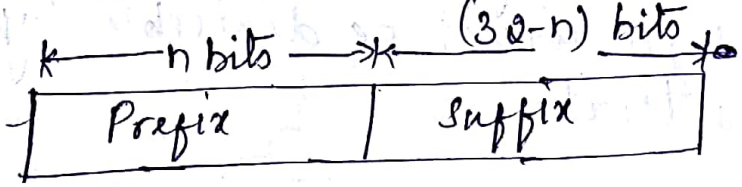
supernet Add = 205.100.0.0

Classless Addressing

With the growth of internet, a larger address space was needed. The long-range solution has already been derived, is called IPV6. The short term solution still uses IPv4 addresses, but it is called classless addressing.

- In classless addressing, the whole address space is divided into variable length blocks. Theoretically we can have a block of $2^0, 2^1, 2^2 \ldots 2^{32}$ addresses.

- In classless addressing an organization is granted a block of addresses, the block is actually divided into two parts, the prefix and the suffix. All addresses in the block have the same prefix, each address has different suffix.



Restriction:-

- Number of Addresses in a block – The number of addresses in a block must be a power of 2.
- Begining addresses → The begining addresses must be evenly divisible by the number of addresses.

Ex: If a block contains 4 addresses, the begining addresses must be divisible by 4.

- Make the rightmost octet in the begining address as zero.

(Q) Which of the following can be begining address of block that contain 256 addresses?

(a) 205.16.37.32 (b) 190.16.42.0
(c) 17.17.32.0 (d) 123.45.24.52

When rightmost byte is 0, the total addresses is divisible by 256.

(Q) Which of the following can be the begining address of a block that contains 1024 addresses?

(a) 205.16.37.32 (b) 190.16.42.0
(c) 17.17.32.0 (d) 123.45.24.52

Ans → 1024 = 256 × 4
so the rightmost byte should be 0, and second rightmost byte must be divisible by 4. Only c satisfies that:

Mask

in classful addressing the mask is implicit, but in classless addressing when an address is given we must also have its mask to know the its belonging block. Thus the address must be accompanied by the mask. Use the CIDR notation the number of 1's in the mask.

* Classless interdomain routing - The slash notation is ~~used~~ formally referred as classless interdomain routing.

byte.byte.byte.byte/n

Prefix length

(Q) One of the addresses in a block is 167.199.170.82/27 Find the number of addresses in the network, the first address, last address.

Soln → n = 27

(a) No of addresses in network $2^{32-n} = 2^{32-27} = 2^5 = 32$

(b) First Address

Address given AND Network Mask = First Address

10100111	11000111	10101010	01010010
11111111	11111111	11111111	11100000
10100111	11000111	10101010	01000000

= 167.199.170.64/27

(c) Last Address

Address given OR complement of N/w mask = Last Add

10100111	11000111	10101010	01010010
00000000	00000000	00000000	00011111
10100111	11000111	10101010	01011111

= 167.199.170.95/27

Special Addresses

All zero's Address — 0.0.0.0/32

It is reserved for communication when a host needs to send an IPv4 packet but it does not know its own address. This is normally used by a host at bootstrap time when it does not know its IPv4 address.

Limited Broadcast Address (255.255.255.255/32)

A host that wants to send a message to every other host can use this address as a destination address in IPv4 packet. However, a router will block a packet having this type of address to confine the broadcasting to local network.

Loopback Address (127.0.0.0/8)

It is used to test the software on a machine. When this address is used, a packet never leaves a machine. client process to send a message to a server process on same machine.

Private Address :- A number of blocks are assigned for private use. They are not recognized globally.

10.0.0.0/8 — 192.168.0.0/16

172.16.0.0/12 — 169.254.0.0/16