CS2105 AY23/24 SEM1 Midterms Cheat Sheet

The Network Core

- Backbone that enables communication and data transfer between various nodes within a network.
- Made up of a mesh of interconnected routers
- Data is transmitted through the network in 2 methods.

Circuit Switching e.g LAN lines	End-end resources allocated to & reserved for "call" between source & dest: • call setup required • circuit-like (guaranteed) performance • circuit segment idle if not used by call (no sharing) • commonly used in traditional telephone network
Packet Switching	Host sending function: • breaks application message into packets, of length L bits • transmits packets onto the link at transmission rate R $packet\ transmission\ delay = \frac{L\ (bits)}{R\ (bits/s)}$ Store and Forward • Packets passed from one router to the next, across links. (Entire packet must arrive @router before transmission) $End-to-end\ Delay\ = 2 \times \frac{L}{R}$ (assuming no other delays) Routing & Addressing: • Routers determine source-dest route taken by packets, via routing algo. • Addressing: each packet needs to carry source & destination information

Internet Structure:

- The internet is a network of networks, organized into Autonomous Systems (AS), each is owned by an organization.
- Hosts connect to Internet via access ISPs (Internet Service Providers)
  - Residential, Company and University ISPs
- Access ISPs in turn must be interconnected. (everything together is very complex)

Types of Delays ** Packet length, L = Number of bits to transmit**

Propagation Delay: d_{prop}	Processing Delay: d_{proc} (nodal processing)
$d_{prop} = {}^{d}/_{S}$ • d → length of physical link • s → propagation speed (~2 x 10^8 m/s)	• Check bit errors • Determine output link (routing algo) • Typically < ms
Queuing Delay: d_{queue}	**Transmission Delay: d_{trans}**
• Time in the queue for transmission • Depdns on congestion lvl of router	$d_{trans} = {}^{L}/_{R}$ • L → packet len; R → link bandwidth

- End-to-end packet delay is the time taken for a packet to travel from source to destination. It consists of: transmission, propagation, processing, queueing delay

Return Trip Time: RTT $RTT = d_{prop} + d_{queue} + d_{proc}$	
Throughput (for end-to-end comm) $Throughput = {}^{L}/_{t}$ • Throughput = bits per unit time	
Link Utilization: ${}^{d_{trans}}/_{(d_{trans} + RTT)}$	

Exp.	Explicit	Prefix	Exp.	Explicit	Prefix
10^{-3}	0.001	milli	10^{3}	1,000	Kilo
10^{-6}	0.000001	micro	10^{6}	1,000,000	Mega
10^{-9}	0.000000001	nano	10^{9}	1,000,000,000	Giga
10^{-12}	0.000000000001	pico	10^{12}	1,000,000,000,000	Tera
10^{-15}	0.000000000000001	femto	10^{15}	1,000,000,000,000,000	Peta
10^{-18}	0.000000000000000001	atto	10^{18}	1,000,000,000,000,000,000	Exa
10^{-21}	0.000000000000000000001	zepto	10^{21}	1,000,000,000,000,000,000,000	Zetta
10^{-24}	0.000000000000000000000001	yocto	10^{24}	1,000,000,000,000,000,000,000,000	Yotta

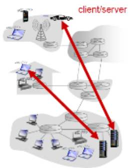
HTTP Request Methods:

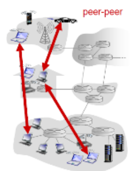
GET	POST	DELETE	PUT	PATCH	HEAD
Retrieve data	Add data to existing file	Delete data	Update/replace existing file	Update a file partially	Retrieve header

Application & Transport Layer (HTTP, DNS, TLS (transport layer security))

- Creating network applications involves writing programs that
  1. Run on different hosts and
  2. communicate over a network. (e.g webserver software ⇔ browser software)
- Done via **client-server** and/or **peer-to-peer (P2P)**.

Client-Server		P2P
Client	**Server**	
• Talks to server + request stuff • For web, client is usually impl. In browser	• Waits for req • Provide requested svc to client • Scale w data centers	• No always-on server • Arbitrary end systems directly comms • Peer request & provide services - Self-scalable (↑P → ↑scale) • Peers are intermittently connected & change IP address (complex management)





Essential transport services for apps

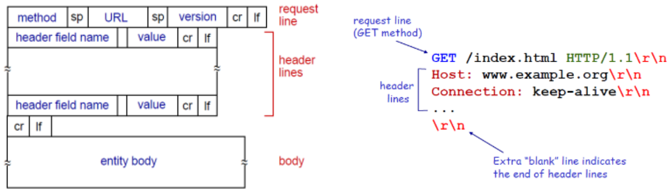
Data Integrity (banking vs streaming) • Some apps require 100% reliable data transfer • Other apps can tolerate some data loss • To achieve data integrity, **TCP** usually used	**Throughput** • Different apps have varying requirements for throughput. • E.g Multimedia requires a min amount of bandwidth to deliver smooth video streaming or high-quality audio. • E.g other apps like file transfer can utilize whatever throughput is available without stringent requirements.
Timing (e.g online interactive games) • Some apps require low delay/latency	
Security • Encryption, Data Integrity, Authentication. • TLS will provide these security features	

Transport Layer Protocols

TCP	• A TCP handshake must be established before the client and host connect. • **Reliable Data Transfer** - Uses ACK, NAK & retransmission. - Data is received in proper format, without error/duplicates/missing data - If any segments are lost/corrupted etc, they are retransmitted • TCP has Flow and Congestion Control (load balancing, prevent overloading etc)
UDP	• **Unreliable Data Transfer** - Not reliable, no guarantee of packet reaching in order - No guarantee that the packet will reach its destination • No Flow & Congestion Control (might overload receiver)

HTTP

- HTTP is an application layer protocol, → it is the foundation of data communication on the WWW.
- It enables the exchange of hypertext, which includes text, images, links, and other media, between web clients (typically browsers) and web servers.
- Stateless Protocol → Cookies hold stateful info (enables server to recognize/rmb client identity/state)
  - Note: Server maintains no information about past client requests. (Cookie comes in handy)



Non-Persistent HTTP (HTTP/1.0 style), TCP	Persistent HTTP (HTTP/1.1 style), TCP
• At Most One Object Per Connection • Connection Closure • Multiple Connections for Multiple Objects **Response time = 2 * RTT + $t_{transmission}$**	• Multiple Objects on a Single Connection • Connection Reuse: • Efficiency and Reduced Overhead

Note: HTTP1.1 → No need to keep requesting to access multiple times (just need 1 TCP connection)

- However, this is still not fast enough since TCP always have a RTT cost.
- Hence, HTTP is changed from TCP to UDP → Current: HTTP 3
- There are cost/benefits for this, e.g reliable/unreliable. TCP → reliable, UDP → not reliable

200: OK	301: Moved Permanently	403: Forbidden	404: Not Found

- Clients (Web browser) can cache a resource and store it locally. However, these cached resources can become stale if the server's resources are updated. Hence, a "If-modified since:<data>" header is in the HTTP request to cross check with the server if the cache is dated.
- If cache not outdated → server returns 303 Not Modified, else updates local cache

DNS (Domain Name Server) holds Resource Records (RR)

Hostname: www.example.com • Hostnames are part of domain names and are typically used to identify web servers, email servers, and other networked devices.	IP address: 93.184.216.34 • A numerical label assigned to each device that uses the IP for communication. • It serves as the address that routers & other networking devices use to forward data to its dest.

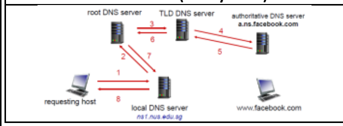
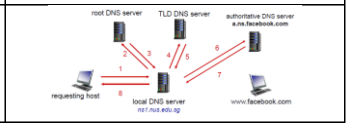
`RR format: (name, value, type, TTL)`

- **Name**: The domain name related to the resource record.
- **Value**: Data associated with the record (IP address, hostname, mail server name, etc.).
- **Type**: The type of the resource record (NS, A, CNAME, MX, etc.).

Type NS: Name Server Name: Domain, Value: Hostname of Auth DNS	Type CNAME: Canonical Name Name: Alias name, Value: Canonical name
Type A: Address Name: Hostname, Value: IP address	Type MX: Mail Exchange Name: Email Server. Value: Name of mail server

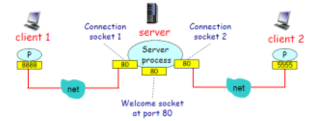
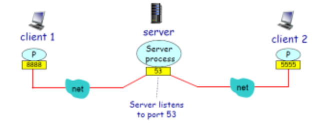
- **TTL** (Time to Live): The expiry date of cache, before they have to refresh/update it.
- Commands: `nslookup`, `dig`

- DNS uses caching to improve query response times & reduce loads on Authoritative DNS servers.
- Cached DNS have a TTL
- DNS runs over UDP (Why?)
  - Most of the time, the Local DNS is queried instead → low chance of package loss/corruption
  - As speed is in mind for this architecture, UDP's fast speed made it appealing over TCP

Recursive DNS (Rarely Used)	Iterative DNS
	

- Note, Iterative is better, as its better for network security. If theres a DNS poisoning, the recursive method will infect more servers, as compared to the iterative approach.

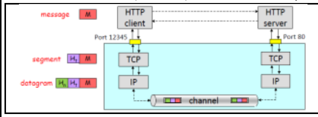
Socket Programming: TCP → reliable, byte stream-oriented, UDP: unreliable datagram socket

TCP Socket	UDP Socket
• When contacted by client, server TCP creates new socket. • Server uses (client IP + port #) to distinguish clients. • When client creates its socket, client TCP establishes connection to server TCP	• Server uses one socket to serve all clients. • No connection established b4 sending data. • Sender explicitly attaches destination IP address and port #to each packet. • Data may be lost/received out-of-order.
	

Transport/Network Layer

Transport layer protocols run in hosts.

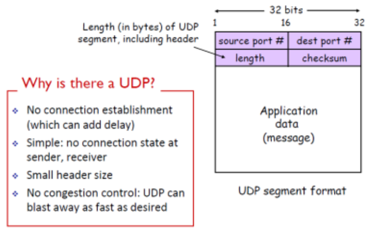
- **Sender side**: breaks app message into segments/packets & passes to network layer for efficiency
- **Receiver side**: reassembles segments/packets into message, passes it to app layer.
- **Packet switches** (routers) in between: only check destination IP address to decide routing.



- Each IP datagram contains source & destIP addr.
- Receiving host is identified by destIP address.
- Each IP datagram carries 1 transport-layer segment.
- Each segment contains source and dest port #.

Connectionless Transport: UDP

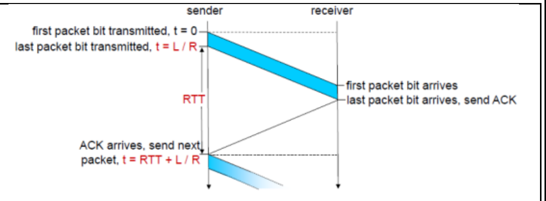
- UDP adds very little service on top of IP:
  - **Multiplexing at sender**: UDP gathers data from processes, forms packets & passes them to IP
  - **De-multiplexing at receiver**: UDP receives packets from lower layer and dispatches them to the right processes.
  - **Checksum**: UDP has a checksum mechanism to detect errors in the transmitted data
  - UDP transmission is **unreliable**

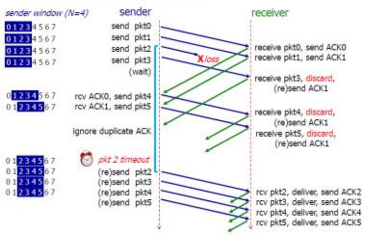


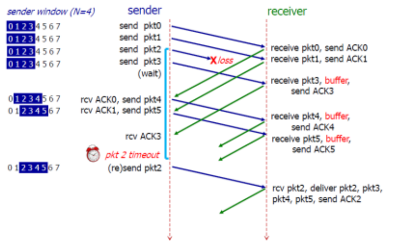
- Transport layer resides on end hosts and provides process-to-process communication.
- Network layer provides host-to-host, best-effort and unreliable communication.

rdt Version	Scenario	Features Used
1.0	no error	nothing
2.0	data Bit Error	checksum, ACK/NAK
2.1	data Bit Error ACK/NAK Bit Error	checksum, ACK/NAK, sequence Number
2.2	Same as 2.1	NAK free
3.0	data Bit Error ACK Bit Error packet Loss	checksum, ACK, sequence Number, timeout/re-transmission

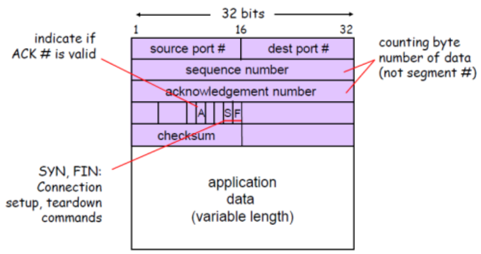
Other Better rdt 3.0 versions:

Stop-and-Wait **(No pipeline)**	 $U_{sender} = \frac{L / R}{RTT + L/R} = \frac{0.008}{30 + 0.008} = 0.00027$

Go-Back-N (GBN) **(Pipelined)**	Go-back-N In Action 
	• Sender can have up to N-unACKed packets in pipeline
	• Receiver only sends cumulative ACKs
	  - Out-of-order packets discarded
	• Sender sets timer for the oldest unACKed packet
	  - when timer expires, retransmit all unACKed packets

Selective Repeat (SR) **(Pipelined)**	Selective Repeat In Action 
	• Sender can have up to N-unACKed packets in pipeline
	• Receiver sends individual ACK for each packet
	  - Out-of-order packets buffered
	• Sender maintains timer for each unACKed packet
	  - When timer expires, retransmit only that unACKed packet

TCP Header:



(some fields are not shown)

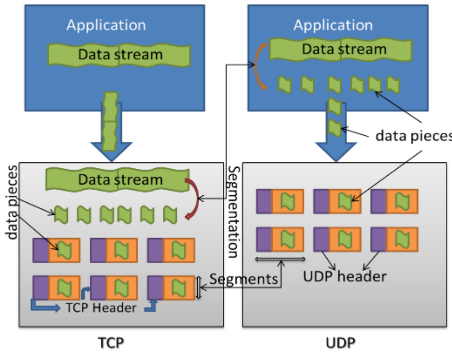
$$\text{EstimatedRTT} = (1-\alpha) * \text{EstimatedRTT} + \alpha * \text{SampleRTT}$$
$$\text{DevRTT} = (1-\beta) * \text{DevRTT} + \beta * |\text{SampleRTT} - \text{EstimatedRTT}|$$
$$\text{Timeout} = \text{EstimatedRTT} + 4 * \text{DevRTT}$$

TCP Segment Header Format

Bit #	0–7	8–15	16–23	24–31
0	Source Port		Destination Port	
32	Sequence Number			
64	Acknowledgment Number			
96	Data Offset / Res	Flags	Window Size	
128	Header and Data Checksum		Urgent Pointer	
160...	Options			

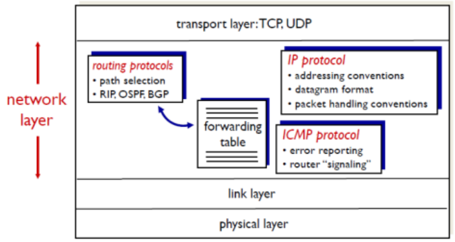
UDP Datagram Header Format

Bit #	0–7	8–15	16–23	24–31
0	Source Port		Destination Port	
32	Length		Header and Data Checksum	



Network Layer

- The Network Layer delivers packets to receiving hosts.
- The Router examine header fields of IP datagrams passing it.

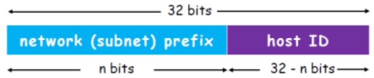


DHCP (Dynamic Host Configuration Protocol)

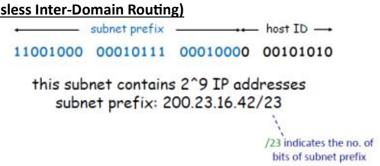
- DHCP allows a host to dynamically obtain its IP address from DHCP server when it joins network.
  - IP address is renewable
  - allow reuse of addresses (only hold address while connected)
  - support mobile users who want to join network.
- DHCP is used to automatically assign IP addresses.
- DHCP may also provide a host additional network information:
  - IP address of first-hop router (equiv to default gateway)
  - IP address of local DNS server
  - Network mask (indicating network prefix versus host ID of an IP address)
- DHCP runs over UDP
  - DHCP server port number: 67
  - DHCP client port number: 68

Special Addresses	Present Use
0.0.0.0/8	Non-routable meta-address for special use
127.0.0.0/8	Loopback address. A datagram sent to an address within this block loops back inside the host. This is ordinarily implemented using only 127.0.0.1/32.
10.0.0.0/8 172.16.0.0/12 192.168.0.0/16	Private addresses, can be used without any coordination with IANA or an Internet registry.
255.255.255.255/32	Broadcast address. All hosts on the same subnet receive a datagram with such a destination address.

IP Subnet:



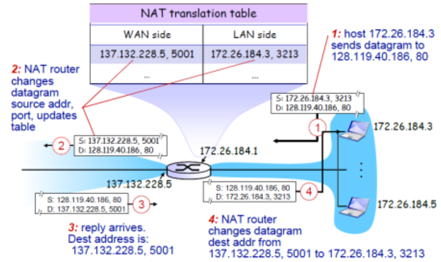
IP Address: CIDR (Classless Inter-Domain Routing)



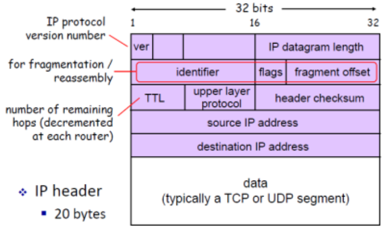
Subnet Mask

- Subnet mask is used to determine which subnet an IP address belongs to.
- Can be made by setting all subnet prefix bits to "1"s and host ID bits to "0"s.

NAT (Network Address Translation)



IPv4 Datagram Format



(some fields are not shown)

IP Fragmentation

- Flag (frag flag) is set to
  - 1 if there is next fragment from the same segment.
  - 0 if this is the last fragment.

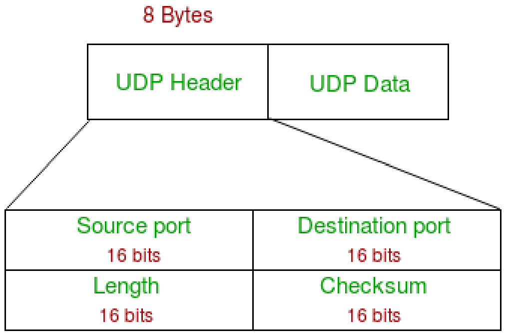
Routing Algorithms (Intra-AS Routing)

- Routers → Vertices; Edges → Links
- Routing == Find least cost path, i.e SSSP Algorithm
- Bellman-Ford: $d_x(y) = min_v\{c(x,v) + d_v(y)\}$
- Common Protocols: RIP, OSPF

ICMP (Internet Control Message Protocol)

Type	Code	Description
8	0	echo request (ping)
0	0	echo reply (ping)
3	1	dest host unreachable
3	3	dest port unreachable
11	0	TTL expired
12	0	bad IP header

Selected ICMP Type and subtype (Code)



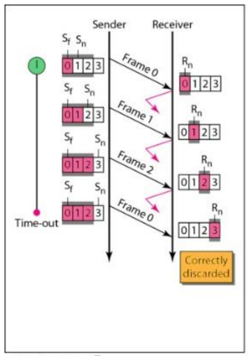
Transmission Control Protocol (TCP) Header

20-60 bytes

source port number (2 bytes)	destination port number (2 bytes)
sequence number (4 bytes)	
acknowledgement number (4 bytes)	
data offset (4 bits) · reserved (3 bits) · control flags (9 bits)	window size (2 bytes)
checksum (2 bytes)	urgent pointer (2 bytes)
optional data (0-40 bytes)	

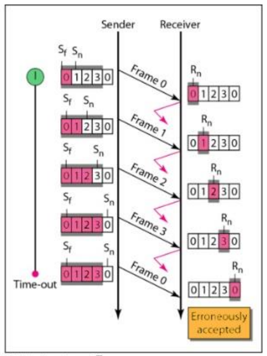
IP HEADER

Version (4 bits)	IHL (4 bits)	Type of Service (8 bits)	Total Length (16 bits)	
Trusted Host ID (16 bits)			Flags (3 bits)	Fragment Offset (13 bits)
Time to Live (8 bits)	Protocol (8 bits)	Header Checksum (16 bits)		
Source Address (32 bits)				
Destination Address (32 bits)				
Options and Padding (multiples of 32 bits)				



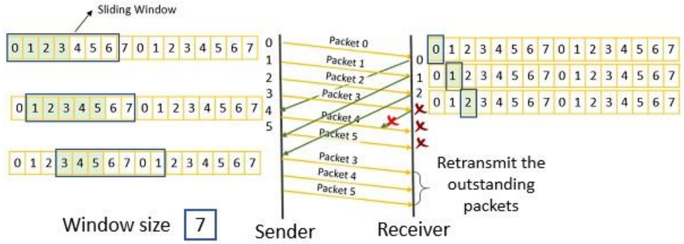
a. Window size $< 2^m$

Or window size $= 2^{m-1}$

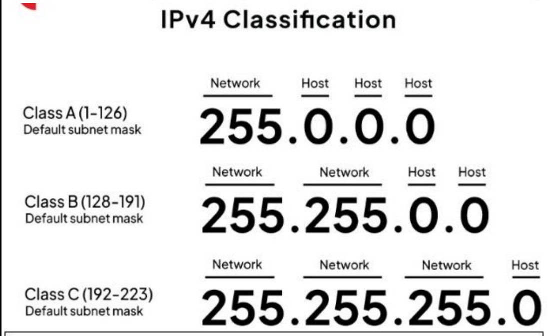


b. Window size $= 2^m$

Or Window size $> 2^{m-1}$



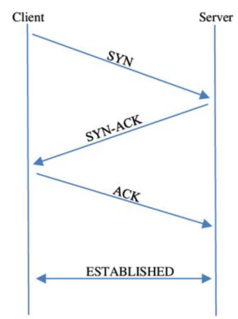
Go-Back-N Protocol



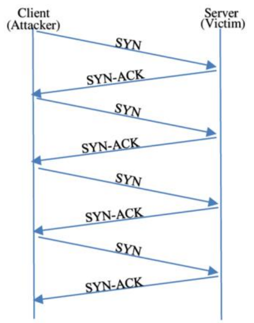
Subnet Mask

Suffix	Hosts	32-Borrowed=CIDR	2^Borrowed = Hosts	Binary=> dec = Suffix
.255	1	/32	0	11111111
.254	2	/31	1	11111110
.252	4	/30	2	11111100
.248	8	/29	3	11111000
.240	16	/28	4	11110000
.224	32	/27	5	11100000
.192	64	/26	6	11000000
.128	128	/25	7	10000000

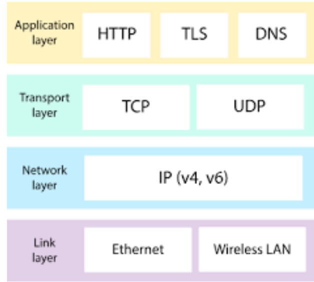
TCP:



Three-Way Handshake



TCP-SYN Flood Attack



For CS2105, we assume receiver will do nothing upon receiving corrupted packet. rdt 3.0 sender will stick to the timer for timeout and retransmission. Hence, corrupted ACK or duplicate ACK can all be ignored. That's why rdt 3.0 receiver can choose not to send duplicate ACK when receiving a corrupted packet (since this duplicate ACK is no use to the sender).

A TCP connection is identified by source/dest IP addresses and source/dest port numbers. Such information needs to be embedded in respective IP/TCP headers (in order for TCP sender and receiver to recognize each other). Moreover, to ensure reliable transmission, you need additional information (e.g. checksum) that is also embedded in TCP header.

- ALL TCP, UDP, TP headers have CHECKSUM!
- GBN and TCP use cumulative ACK, SR uses Non-cumulative ACK
- GBN and TCP uses 1 timer, SR uses multiple timer

	GBN	SR	TCP
ACK No (m)	ACK m means that the receiver has received all the packets up to packet m	ACK m means that the receiver has received packet m. But there is no implication on the receipt of other packets	ACK m is the sequence number of the nxt byte of data expected by the receiver (i.e nxt is m, received till m-1 bytes)
Out-of-order packet	Receiver discards and sends the ACK for the expected packet	SR allows OOOP. Will buffer OOOP	Wont happen. TCP guarantees in-order delivery of data (seq no)
Seq No.	Sequence number represents the number assigned to each packet in the order they are sent. These sequence numbers are used to identify and order packets.		Sequence numbers are used to identify each byte of data within a stream, not just packets.

TCP is GBN or SR?

- GBN: ACK number is seq # of pkt being ACKed. TCP: ACK number represents the expected next number.
- GBN: No buffering at Receiver, TCP: buffering at Receiver
- GBN sender retransmits the pkt n and all higher seq # pkts in window at timeout(n). But, TCP retransmits only pkt n.