1. (5 points) What are NAP and Private Peering in the Internet Architecture? Why are they useful?

Sol: NAP stands for National Access Point (somewhere called Network Access Point) (1’). It is public peering, which is interconnection utilizing a multi-party shared switch fabric such as an Ethernet switch. (2’)

Private peering, however, is interconnection utilizing a point-to-point link between two parties. (2’)

![NAP and Private Peering Diagram]
2. (5 points) What are the five layers in Internet model? Describe the major function of each layer.

Sol:

**Internet protocol stack**

- **application**: supporting network applications
  - ftp, smtp, http
- **transport**: host-host data transfer
  - tcp, udp
- **network**: routing of datagrams from source to destination
  - ip, routing protocols
- **link**: data transfer between neighboring network elements
  - ppp, ethernet
- **physical**: bits “on the wire”

Each layer (1’ * 5)
3. (10 points) Describe the process of close a TCP connection

Sol:

4-way handshake (8’), timed wait (2’
4. (10 points) What is the sliding window scheme that we discussed in the class? Describe the function of sender’s window and receiver’s window. How does the receiver’s window size correspond to GBN or SR schemes?

Sol: The sliding window scheme we discussed in class is GBN and SR. For both schemes, the sender window will move forward when a packet with least seq # is acked. And both receiver windows will move forward when successfully they receive the packet with least seq# successfully.

The difference is that when packets do not arrive in order at the receiver side, GBN will discard them. Because it only remembers the expected packet (in other word, its window size at the receiver side is 1). However, SR will buffer N packets at the receiver side, even they do not come in order (in other word, its window size at the receiver side is N).

**Go-Back-N**

Sender:
- k-bit seq # in pkt header
- “window” of up to N, consecutive unack’ed pkts allowed

- ACK(n): ACKs all pkts up to, including seq # n - “cumulative ACK"
  - may deceive duplicate ACKs (see receiver)
- timer for each in-flight pkt
- timeout(n): retransmit pkt n and all higher seq # pkts in window
**Selective Repeat**

- Receiver *individually* acknowledges all correctly received pkts
  - buffers pkts, as needed, for eventual in-order delivery to upper layer
- Sender only resends pkts for which ACK not received
  - sender timer for each unACKed pkt
- Sender window
  - $N$ consecutive seq #’s
  - again limits seq #’s of sent, unACKed pkts

---

Sliding window scheme (2'), Function of window (4'), Receiver’s window size (4')
5. What is the congestion control scheme used by TCP? Describe both slow-start and congestion avoidance phases in details.

Sol:

TCP congestion control:

- "probing" for usable bandwidth:
  - ideally: transmit as fast as possible (Congwin as large as possible) without loss
  - increase Congwin until loss (congestion)
  - loss: decrease Congwin, then begin probing (increasing) again

- two "phases"
  - slow start
  - congestion avoidance

- important variables:
  - Congwin
  - threshold: defines threshold between two slow start phase, congestion control phase

TCP Slowstart

Slowstart algorithm

initialize: Congwin = 1
for (each segment ACKed)
  Congwin = Congwin * 2
until (loss event OR CongWin > threshold)

- exponential increase (per RTT) in window size (not so slow!)
- loss event: timeout ( Tahoe TCP) and/or or three duplicate ACKs ( Reno TCP)

Note there’s a typo here.
For (each segment ACKed)
Congwin = Congwin + 1
TCP Congestion Avoidance

Congestion avoidance

/* slowstart is over */
/* Congwin > threshold */
Until (loss event) {
    every w segments ACKed:
        Congwin++
}
threshold = Congwin/2
Congwin = 1
perform slowstart

1: TCP Reno skips slowstart (fast recovery) after three duplicate ACKs

Describe congestion control (2’), slow start (4’), congestion avoidance (4’)}
6. (10 points) Describe the Distance Vector routing algorithm. What is the distance vector for a given node? How does a router based on the algorithm to decide its routing table?

**Distance Vector Routing Algorithm**

**iterative:**
- continues until no nodes exchange info.
- self-terminating: no "signal" to stop

**asynchronous:**
- nodes need not exchange info/iterate in lock step!

**distributed:**
- each node communicates only with directly-attached neighbors

**Distance Table data structure**
- each node has its own
- row for each possible destination
- column for each directly-attached neighbor to node
- example: in node X, for dest. Y via neighbor Z:

\[
D(X,Y) = \text{distance from } X \text{ to } Y, \text{ via } Z \text{ as next hop}
\]

\[
= c(X,Z) + \min \{D^Z(Y,w)\}
\]

Describe DV (2’), how to calculate DV (4’), how to get routing table (4’
7. (10points) A station sends a packet via packet switching to another station via 5 hops away. Describe the total delay in term of all possible component delays.

Sol: There are 4 kinds of delay: transmission delay, propagation delay, node processing delay, and queueing delay.

a. Transmission delay
b. Propagation delay
c. Queueing delay
d. Nodal processing delay

For a packet switched 5 hops away, there will be 4 nodal processing delays, 4 transmission delays, 4 queueing delays, and 4 queueing delays.

Each delay (2’ * 4), describe the 5 hop process (2’).
8. (10 points) When Bob decided to send an email message to Alice at a remote site, describe the process that this message will go through before Alice receives the message.

Sol:

Sender and receiver’s mail server (3* 2), the protocol in each process, SMTP, POP3/IMAP, (2’ * 2)
9. (10 points) In TCP/IP network, the timeout period is closely related to the estimated RTT. Describe how the RTT and timeout were determined in TCP/IP network.

**TCP Round Trip Time and Timeout**

**Q: how to set TCP timeout value?**
- longer than RTT
  - note: RTT will vary
- too short: premature timeout
  - unnecessary retransmissions
- too long: slow reaction to segment loss

**Q: how to estimate RTT?**
- **SampleRTT**: measured time from segment transmission until ACK receipt
  - ignore retransmissions, cumulatively ACKed segments
- **SampleRTT** will vary, want estimated RTT "smoother"
  - use several recent measurements, not just current SampleRTT
TCP Round Trip Time and Timeout

EstimatedRTT = (1-x) * EstimatedRTT + x * SampleRTT

- Exponential weighted moving average
- Influence of given sample decreases exponentially fast
- Typical value of x: 0.1

**Setting the timeout**

- EstimatedRTT plus “safety margin”
- Large variation in EstimatedRTT → larger safety margin

\[ \text{Timeout} = \text{EstimatedRTT} + 4 \times \text{Deviation} \]

\[ \text{Deviation} = (1-x) \times \text{Deviation} + x \times |\text{SampleRTT} - \text{EstimatedRTT}| \]