Indian Institute of Technology Kharagpur
Dept. of Computer Science & Engineering
End-Semester Examination, Spring 2022-23

Subject No.: CS31204/CS31006 **Subject Name: Computer Networks**
Time: 3 Hours **Full Marks: 100**

1. **Answer all parts of Q1 consecutively. Be brief and specific in your answers.**

(a) State two advantages of the datagram approach over the virtual circuit approach. (2)

(i) Since different datagrams can go over different paths, all datagrams are not lost if one path is broken, (ii) no setup time before sending first datagram
(Ok if you write the same thing in different ways, or cite any other reasonable advantage)

(b) What would be the size of an Ethernet frame carrying 32 bytes of data of an application using UDP? Assume that no options are used at any layer. Justify your answer briefly. (3)

Final size of frame transmitted will be data + UDP header size + IP header size + Ethernet header size = 32 + 8 + 20 + 18 =78 bytes. *(1 mark for each header size)*

(c) Consider an Ethernet frame carrying an IP packet, with some pad bits added. How are the pad bits separated from data at the receiver end? (3)

Pad bits will be separated from data at the IP layer using the total length field in IP header. The total length field, which will be put in by the sender, will contain only the IP packet size, so the receiver can look at this value and get the IP packet out ignoring the pad.

(d) Show how 8 servers can be connected using two 12-port switches such that each server can communicate with any other server even if one switch fails. No explanation is needed. (3)

First thing to note is that if the servers all need to remain connected even if one switch fails, each server must be connected to both switches. So the implicit assumption is that each server has at least two NICs. Now just connect each server directly to both switches (one NIC connected to one switch, the other NIC to the other switch). The switches need not be connected to each other directly. Do not worry about which path will be taken by which packet, the question does not ask that.

(e) Suppose that an IP datagram is broken into 6 fragments, numbered from 0 to 5. Fragments 0, 1, 3, 4, 5 arrive at the destination but fragment 2 is delayed. The reassembly timer expires at the destination. Then fragment 2 arrives. What will happen to fragment 2? (3)

Note that unless something is remembered, the destination cannot distinguish between the case above, and the case when all fragments other than 2 are delayed (2 arrives first). First id field is checked to see if reassembly timer is running for this datagram, it will be found that it is not running in this case. Then either one of these two answers are acceptable: (i) fragment 2 is stored, reassembly timer is started again, (ii) the identification field is checked with a stored list to see if other fragments of this datagram have already been dropped; if so, drop 2, else store it and start the timer.

(f) Suppose an organization using NAT needs at most 20,000 connections to be supported at any time. What is the minimum number of public IPs that will be needed (justify your answer in 1 sentence)? Is there any benefit of using more public IPs for NAT'ing than this minimum number of public IPs needed in such a system? Justify briefly. (4)

The minimum number needed is 1 as using 16 bit port field, 20000 connections can be supported by 1 IP and 20000 ports. (2 marks)

Yes. Even though one public IP is enough to support, the outside world sees all traffic originating from one IP only. One example where this is bad is that a service may take policy decisions to ban the IP for overuse; for example, think of all 20000 connections going to one download server (for something) which has a restriction of maximum 1000 downloads from one IP per day.

Any other properly justified example is also ok. (2 marks)

2. (a) When is a routing algorithm said to be (i) robust, and (ii) optimal? (3)

(b) Write the pseudocode for the actions taken by a node X on receiving a distance vector update message from a neighbor Y in a distance vector routing protocol. Assume that when a node finds a route to a destination to be unreachable, it removes the route from the routing table immediately instead of first setting its cost to ∞ for some time (ignore other problems it may cause). Write in pseudocode form as shown in class, do not show list of steps or write paragraphs. (7)

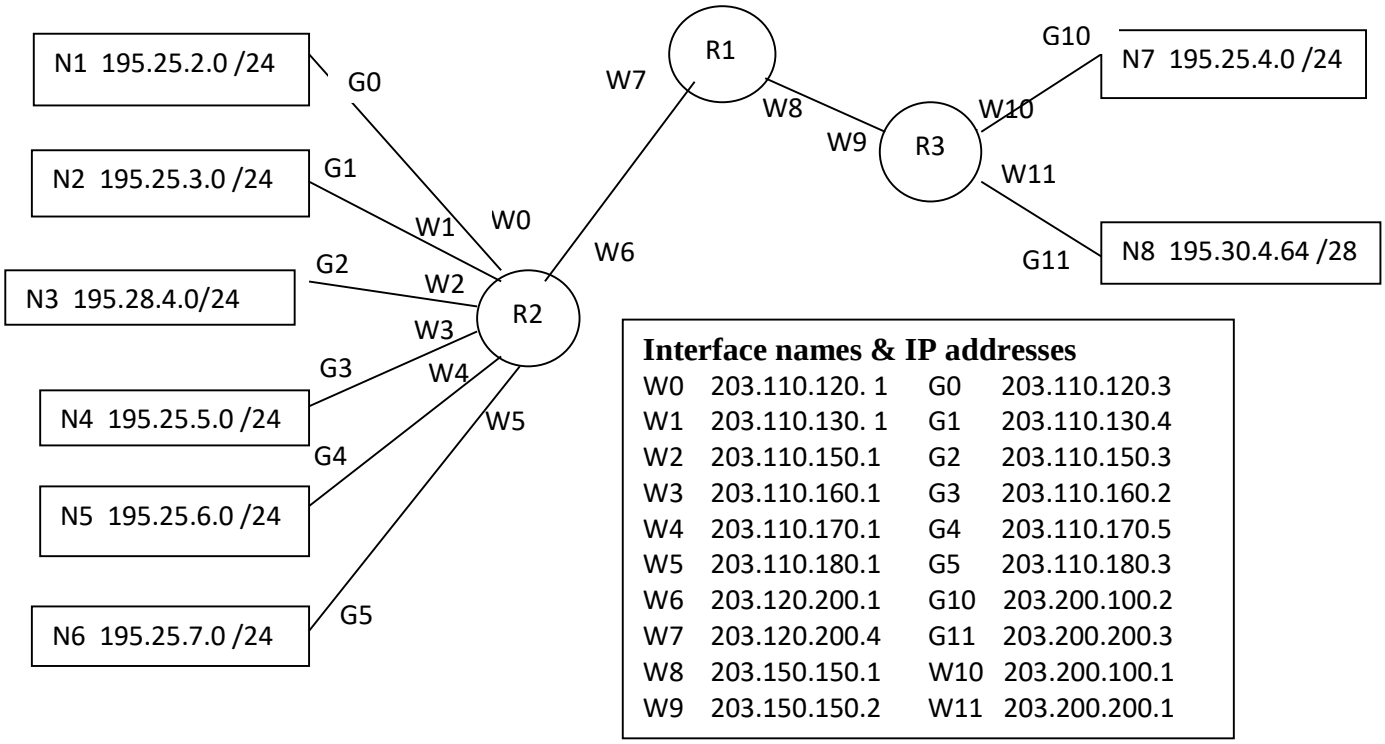
Discussed in class. Same pseudocode as in slide + an additional condition: if X has an entry for a destination d with next hop Y, and if Y's routing table received does not have any entry for destination Y, remove it from X's table too.

(c) Consider a link state routing protocol *L* in an IP-based network with 3 types of messages. *L* runs directly on top of IP. One of these three types is a message called *Link State Update* (LSU). Each LSU contains multiple *Link State Advertisements* (LSA). Each LSA carries information about one link and is a 4-tuple <*node1, node2, cost, status*> where *node1* and *node2* are IP addresses of the nodes at the two ends of the link, *cost* is a 1-byte integer for the cost of the link, and *status* is a flag indicating whether the link is up or down. A node floods an LSU message periodically containing one LSA for each of the links incident on it. An LSU can be forwarded at most 50 times during the flooding, though a node sending a LSU configures the actual value for this in the packet when sending. Design the packet format for the LSU type message. Specifically, your answer should draw the format of a LSU packet showing the position of the fields (similar to header/packet formats you have studied for other protocols), and then list the fields and a 1-line justification of each field beside it as to why the field is needed. (10)

- *Type (1 byte): to distinguish between 3 types of messages (2 bits is also to be given full marks, but to align on byte boundaries, better to have 1 byte or mention that other bits following the 2 bits are reserved)*
- *NumLSA (1 byte): to store how many LSAs are carried in this LSU message. Assuming maximum 255 LSAs (max degree of a node is 255). Ok if you take 2 bytes also.*
- *TTL (1 byte): to store a value between 1 to 50 to indicate how many hops to flood (6 bits is also to be given full marks, but again, to align on byte boundary, better to have 1 byte).*

- *Sequence No. (2 bytes): to identify stale LSU messages, since it is run on IP and so order may not be maintained. Ok if you have chosen other no. of bytes, though 1 byte is too less.*
- *NumLSA number of LSA's, each having:*
    - *4 byte for IP of node 1*
    - *4 byte for IP for node 2*
    - *1 byte for cost*
    - *1 byte for status (again 1 bit is to be given full marks given that status is only up and down, but having a byte is better for alignment)*

Alignment is usually on word boundary rather than byte boundary, but we will ignore that here. Also, name of the fields can be anything as long as purpose is ok. Additional fields are ok, but not marked.



Interface names & IP addresses

W0	203.110.120. 1	G0	203.110.120.3
W1	203.110.130. 1	G1	203.110.130.4
W2	203.110.150.1	G2	203.110.150.3
W3	203.110.160.1	G3	203.110.160.2
W4	203.110.170.1	G4	203.110.170.5
W5	203.110.180.1	G5	203.110.180.3
W6	203.120.200.1	G10	203.200.100.2
W7	203.120.200.4	G11	203.200.200.3
W8	203.150.150.1	W10	203.200.100.1
W9	203.150.150.2	W11	203.200.200.1

3. **(a)** Explain with an example how NAT (Network Address Translation) works. Assume that you have 2 public IP addresses 200.100.20.10 and 200.100.20.11, and 4 machines with private IP addresses 10.1.2.3, 10.1.2.4, 10.1.2.5, and 10.1.2.6, with the first 2 connecting to google.com (say with IP address 100.200.100.5) and the last 2 connecting to amazon.com (say with IP address 150.100.100.50) at the same time. (6)

(b) Consider the routers R1, R2, R3 that route packets between the subnets N1, N2,…,N8 as shown (with their network addresses) in the figure. Each subnet has one gateway connected to a router. The gateways in the subnets are at distance one hop from the routers, and hop count is used as the metric. The names of the interfaces at the routers (W's) and the interfaces at the gateways (G's) that they connect to, and their IP addresses, are as shown. Show the main fields of the entries in the IP routing tables at R1, R2, and R3. The number of routes at each router should be minimized. (12)

The key thing to note here is that for R2 and R3, they must have one entry for each subnet they are connected to, with next hop as the gateway IP, ***plus they must have at least a default entry pointing to R1*** *for going to the other networks (for R2 to go to R3's network and vice-versa). R1 on the other hand can do route aggregation wherever possible. In this case, the supernets that can be formed are 195.25.4.0/22 (from 195.25.4.0/24 to 195.25.7.0/2) and 195.25.2.0/23 (from 195.25.2.0/24 and 195.25.3.0/24). Note that though the first supernet includes 195.25.4.0/24 which is not connected to R2, a more specific entry for the same at R1 can take care of this, by longest-prefix match. So the entries are (shown as triplets <destination network, nexthop IP, cost>; not showing interface name):*

At R3: <195.25.4.0/24 203.200.100.2 1>, <195.30.4.64/28 203.200.200.3 1>, (default, 203.150.150.1 1>
At R2: <195.25.2.0/24 203.110.120.3 1>, <195.25.3.0/24 203.110.130.4 1>, <195.28.4.0/24 203.1150.3 1>, <195.25.5.0/24 203.110.160.2 1>, <195.25.6.0/24 203.110.170.5 1>, <195.25.7.0/4 203.110.180.3 1>, <default, 203.120.200.4 1>

At R1: <195.25.4.0/24 203.150.150.2 2>, <195.30.4.64/28 203.150.150.2 2>, <195.28.4.0/24 203.120.200.1 2>, <195.25.4.0/22 203.120.200.1 2>, <195.25.2.0/23 203.120.200.1 2>

Cost values can be +1 depending on how you count. No marks where deducted for cost values as long as you have them reasonably, deduced only if you did not have any cost at all.
Also did not include entries to reach the routers and gateways as individual machines, almost all of you did not do it either probably as they are usually not source and destinations of data, though there are other needs to reach them, was not expected here and ignored here.

Other variations are also possible, all answers have been given due consideration and credit depending on the number of final route entries, correctness etc.

(c) Consider a long path from a source X to a destination Y with different MTU's for different intermediate links. You wish to design a protocol that will be run by the IP layer of X before sending any data to Y so that the chance of any fragmentation occurring in the intermediate routers is reduced when the data is actually sent by X (need not be guaranteed). Design the protocol for this (first think **what** the protocol needs to compute to do what it is supposed to do, before thinking **how** to do it). (6)

Start from the requirement. You do not want the intermediate routers to fragment. So why will any intermediate router fragment? Because the packet to be sent has size > MTU of the link. So to stop fragmentation, packet size sent must be less than MTU of link. Since the MTU's are different on outgoing links from different intermediate routers, you need to get the minimum MTU of any link on the path, so that packets can be sent with size less than that so no intermediate routers will need to fragment.

So how to estimate the minimum MTU? You know how to get feedback using ICMP. Use the DF flag in IP header. Send an IP packet with DF flag set to 1. If a router needs to fragment, it cannot because DF=1. So it will drop it, and send a Destination Unreachable ICMP back with "Fragmentation needed but cannot fragment" code. So now keep sending packets with smaller and smaller size, until you stop getting this ICMP (can do binary search also). The point where you do not get the ICMP is the size that is lower than any MTU. It is just an estimate, and best effort, just like everything else with ICMP. You can also go the other way by sending the smallest packet first and keep sending larger packets until you get an ICMP

(2 marks for use of DF flag, 2 marks for sending packets of different sizes, 2 marks for using ICMP)

Many of you have built your own protocols, some correct, some incorrect. This protocol then needs to run on all machines on the internet, since the source and destination can be anywhere. Asking the internet to change for you is not acceptable, when a solution using mechanisms already there on all machines already exists. All such answers have got 2 out of 6 (mostly).

4. **(a)** Explain how congestion control works in TCP Reno. (8)

(2 marks for basic slow start, 2 marks for basic congestion avoidance, 1.5 for timeout handling, 2.5 for 3 duplicate ack handling)

(b) Consider a path with bandwidth 10 Mbps and round-trip delay 200 milliseconds. Is slow start/congestion avoidance (ignore fast retransmit and fast recovery) a good scheme for congestion control on such a link? Justify your answer briefly. (5)

No. During congestion avoidance phase, it increases the congestion window by only 1 MSS every RTT. In one RTT, the number of bytes that could have been sent is 256 KB (10 Mbps x 200 milliseconds). Instead, the increase is only 1 MSS which is very small in comparison to what could have been sent. So if there is no congestion, it will take too long for the window to reach a size that can utilize the link fully.

Note that other problems are also there like slow-start starting at congestion window = 1 and slow start, even though fast, will need some RTTs to reach ss_tthres causing the same inefficiency problem. Any reasonably explained answer has been given due consideration.

(c) List the steps taken and events that occur when a TCP packet is received at the TCP layer. Assume that it is not a packet for connection establishment/termination/reset, URG/PSH flags are not used, and there are no options used. (9)

The answer can be broken down in three parts: general things that happen always, things that happen when data is received, and things that happen when acknowledgements are received. Some packets may carry both data and ack, they will contribute to all parts.

General steps and events: (3 marks)

- *Pseudo header is formed and checked with checksum in TCP header (1 mark)*
- *Demultiplex using the 4-tuple <source IP, source port, destination IP, destination port> to identify the connection this TCP segment belongs to. Note that IP is important as the same TCP software will demultiplex packets coming into say two different NICs. (1 mark)*
- *Receive window is updated as per the value received in the window field of the segment (1 mark)*
- *If the segment carries data (3 marks, any 3 of the following are fine)*
    - *If it is a segment with sequence no. equal to next sequence no. expected, send an ack if any previous segment is un-ack'ed, or delay 500 milliseconds and then send ack*
    - *If it is an out-of-order segment having a higher than expected sequence number, send an ack with next sequence number expected and store the bytes*
    - *If it is a missing segment which extends the longest contiguous byte stream from the start that it has received, send an ack with the next sequence number to expect*
    - *If it is a duplicate segment, discard the segment but send an ack with the next sequence number expected*
    - *If it is a segment with sequence no. equal to next sequence no. expected, or if it is a missing segment which extends the longest contiguous byte stream from the start that it has received, put the new data received or the data that it completes in the connection's receive buffer*
- *If the segment carries acknowledgement (either only ack or piggybacked ack) (3 marks, any 3 of the following are fine)*
    - *If it is a non-duplicate ack, stop the timer associated with the segment*
    - *If it is a duplicate ack, retransmit segment expected*

- *If it is ack for a non-retransmitted segment, update RTT estimate by Karn's algorithm*
- *Update sender window in sliding window flow control if needed*
- *Update congestion window as per congestion control algorithm followed*
- *Send new segment if permitted by Nagle's algorithm and window sizes*

(Answers that are written reasonably to have enough number of points, or some other points that are not included above but are correct, have been given due credit. However, it is surprising to see very few of you even mention the effect of an ack on timeout computation, or on congestion control, or on sending new packets, all of which are clearly affected by receive of acks. The question was not just on how header fields are processed, it asked for all steps taken and events that occur.)

(d) Suppose a send call puts X bytes of data to be sent in a TCP send buffer. Explain how TCP decides how many bytes of the X bytes of data will be sent in the next segment? (4)

First assume that min(congestion window, receiver window) is $\geq X$ bytes. Then there are two cases:

- *If $X < MSS$, then next segment sent will have X bytes (sent immediately if no ACK pending or after a pending ACK is received)*
- *If $X \geq MSS$, then next segment size will be MSS size (sent immediately)*

Now if min(congestion window, receiver window) is < X bytes, then the next segment will have size = min(congestion window, receiver window) subject to MSS

(2 marks for Nagle's, 1 mark each for receive window and congestion window)

5. **(a)** List the steps taken by a node X in obtaining a dynamic IP address from a DHCP server not in X's subnet using a DHCP relay agent. For each message sent, list the source IP, source port, destination IP, destination port in the IP packet carrying the message, and values put in other important fields of the DHCP packets sent (exact message format not needed). Does the DHCP server always have to run in a computer in practice? Justify in one sentence. (8)

(b) What is the difference between recursive and iterative query in DNS? (4)