**CS 4313 & 5313 Computer Networks (Spring 2022)**

**Homework 6**

* **Assignment Date: April 20 (Wednesday), 2022**
* **Due Date: April 26 (Tuesday), 11:59 pm**
  + **10% penalty for late submission per day**
  + **Will NOT accept HW after April 27**
* **Submission:** 
  + **Upload your file on Black Board (Please, use MS word format or pdf, not pic format)**
  + **One single file, not multiple files**
* **Problem: Some questions are from Computer Networks: A top-down approach (the 7th edition)**
* **Total: 180 points (10 points each question)**

**Chapter 3:**

18. (R3) Consider a TCP connection between Host A and Host B. Suppose that the

TCP segments traveling from Host A to Host B have source port number x and destination port number y . What are the source and destination port numbers

for the segments traveling from Host B to Host A?

Source: y

Destination: x

19. (R4) Describe why an application developer might choose to run an application over UDP rather than TCP.

When the application needs speed rather than guaranteed connection or security for example Video.

20. (R14) True or False?

1. Host A is sending Host B a large file over a TCP connection. Assume

Host B has no data to send Host A. Host B will not send acknowledgments

to Host A because Host B cannot piggyback the acknowledgments on data.

* False

1. The size of the TCP rwnd never changes throughout the duration of the connection.

* false

1. Suppose Host A is sending Host B a large file over a TCP connection. The number of unacknowledged bytes that A sends cannot exceed the size of the receive buffer.

* True, except when sending 1 byte with size 0 in avoiding deadlock

1. Suppose Host A is sending a large file to Host B over a TCP connection. If the sequence number for a segment of this connection is m, then the sequence number for the subsequent segment will necessarily be m + 1.

* false

1. The TCP segment has a field in its header for rwnd.

* true

1. Suppose that the last SampleRTT in a TCP connection is equal to 1 sec. The current value of TimeoutInterval for the connection will necessarily be ≥ 1 sec.

* false

1. Suppose Host A sends one segment with sequence number 38 and 4 bytes of data over a TCP connection to Host B. In this same segment the acknowledgment number is necessarily 42.

* false

21. (R17) Suppose two TCP connections are present over some bottleneck link of rate R bps. Both connections have a huge file to send (in the same direction over the bottleneck link). The transmissions of the files start at the same time. What transmission rate would TCP like to give to each of the connections?

R/2

22. (P3) UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.) Show all work. Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error?

01010011 10111001

+01100110 +01110100

------------- ---------------

10111001 00101110 **1s: 11010001**

* **In order to find errors**
* **If there is any 0 in sum, then here is error.**
* **No**
* **It is possible**

23. (P4)

1. Suppose you have the following 2 bytes: 01011100 and 01100101. What is the 1s complement of the sum of these 2 bytes?

+01100101

-------------

11000001 **1s: 00111110**

1. Suppose you have the following 2 bytes: 11011010 and 01100101. What is the 1s complement of the sum of these 2 bytes?

11011010

+01100101

--------------

01000000 **1s: 10111111**

1. For the bytes in part (a), give an example where one bit is flipped in each of the 2 bytes and yet the 1s complement doesn’t change.

**First: 01010100**

**Second: 01101101**

24. (P24) Answer true or false to the following questions and briefly justify your

answer:

1. With the SR protocol, it is possible for the sender to receive an ACK for a

packet that falls outside of its current window.  
-> True

1. With GBN, it is possible for the sender to receive an ACK for a packet that

falls outside of its current window.

-> True

1. The alternating-bit protocol is the same as the SR protocol with a sender

and receiver window size of 1.

* True

1. The alternating-bit protocol is the same as the GBN protocol with a sender

and receiver window size of 1.

* True

25. (P37) Compare GBN, SR, and TCP (no delayed ACK). Assume that the timeout values for all three protocols are sufficiently long such that 5 consecutive data segments and their corresponding ACKs can be received (if not lost in the channel) by the receiving host (Host B) and the sending host (Host A) respectively. Suppose Host A sends 5 data segments to Host B, and the 2nd segment (sent from A) is lost. In the end, all 5 data segments have been correctly received by Host B.

1. How many segments has Host A sent in total and how many ACKs has Host B sent in total? What are their sequence numbers? Answer this question for all three protocols.

**GBN: 9 segments including 12345 and 2345. 8 acks from B with seq # 1 and seq # 2345.**

**SR: 6 segments including 12345 and 2. 5 acks from B with seq # 1345 and seq # 2.**

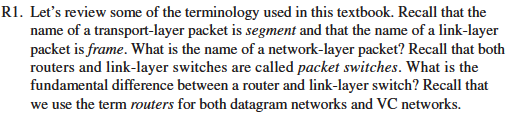
**TCP: 6 segments including 12345 and 2. 5 acks from B with seq # 2 and seq # 6.**

1. If the timeout values for all three protocol are much longer than 5 RTT, then which protocol successfully delivers all five data segments in shortest time interval?

**TCP: fast transmit**

**Chapter 4:**

Q1.



**Datagram.**

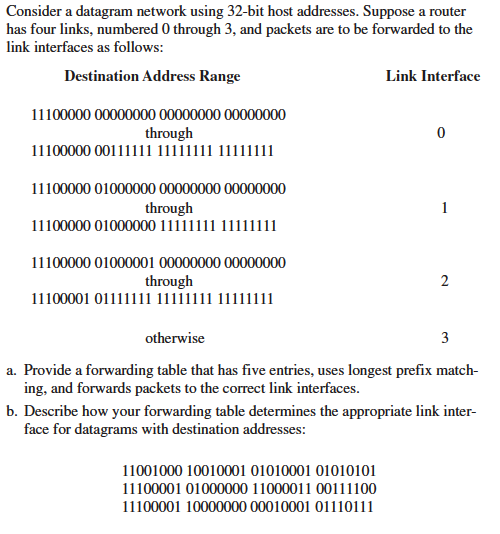
**Difference is what is the base of forwarding a packet as router uses IP and link-layer switch uses MAC.**

Q2. What is the difference between routing and forwarding?

**Forwarding: moving a pkt from input to output in router**

**Routing: determining the path taken from source to destination in routers.**

Q3. (P5)

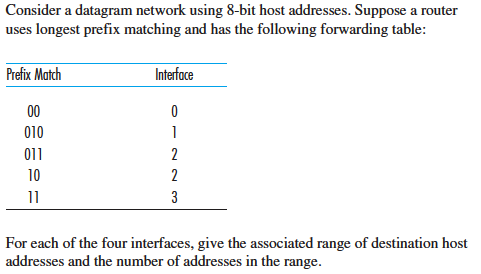


|  |  |
| --- | --- |
| **11100000 00** | **0** |
| **11100000 01000000** | **1** |
| **1110000** | **2** |
| **11100001 1** | **3** |
| **Other wise** | **3** |



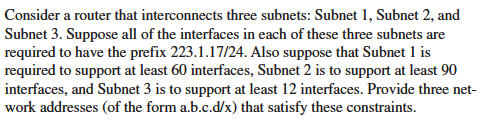
**The first one matches otherwise. The second one matches link 2. The last one matches link 3.**

Q4. (P6)



|  |  |
| --- | --- |
| **00000000**  **00111111** | **Link 0** |
| **01000000**  **01011111** | **Link 1** |
| **01100000**  **01111111** | **Link 2** |
| **10000000**  **10111111** | **Link 2** |
| **11000000**  **11111111** | **Link 3** |

Q5. (P8)



**223.1.17.0/25 – 60 interfaces**

**223.1.17.128/26 – 90 interfaces**

**223.1.17.192/26 – 12 interfaces (rest)**

**Chapter 5**

Q6. (R3) Compare and contrast the properties of a centralized and a distributed routing algorithm. Give an example of a routing protocol that takes a centralized and a decentralized approach.

**Centralized: calculates a table centrally and distributes it to each router**

**There is a remote controller taking to local control agents.**

**Distributed: calculates a table using distributed algorithm in each router.**

**Each router only knows neighbors**

Q7. (R5) What is the “count to infinity” problem in distance vector routing?

**It happens when error propagates in routers such as routers sed updates to one another.**

Q8. (R20) What two types of ICMP messages are received at the sending host executing the Traceroute program?

**Warning and destination port unreachable messages**

**Chapter 6**

Q9 (R9) How big is the MAC address space? The IPv4 address space? The IPv6 address space?

**MAC: 48bits**

**IPv4: 32bits**

**IPv6: 128bits**

Q10 (R11) Why is an ARP query sent within a broadcast frame? Why is an ARPresponse sent within a frame with a specific destination MAC address?

1. **First sender has to know both ip and mac address. To get mac address, ARP is sent via broadcast.**
2. **In response, sender knows where to send.**

Q11 (P5) Consider the 7-bit generator, G=10011, and suppose that D has the value1010101010. What is the value of R?

**10101010100000 / 10011 = 101101111…0101**

**R = 0101**