Test #2

CSCI-3400 - Spring-2023

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Instruction: Write clearly and give full justification to each question. Show all your MATHEMAICAL working.**

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**Name: \_\_\_\_\_\_\_\_\_\_\_Tyler Burleson\_\_\_\_\_\_\_\_\_\_**

**E#:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_E00672866\_\_\_\_\_\_\_\_\_\_\_\_**

|  |  |  |
| --- | --- | --- |
| Questions | Max Points | Earned Points |
| 1 | 10 |  |
| 2 | 10 |  |
| 3 | 10 |  |
| 4 | 10 |  |
| 5 | 10 |  |
| 6 | 10 |  |
| 7 | 10 |  |
| 8 | 10 |  |
| 9 | 10 |  |
| 10 | 10 |  |
| Total | **100** |  |

**IMPORTANT**

* You can use online references, but **write answers in your own words**!  **Cite any references used**. Answer the question in your own words, no credit will be given for answers copied from any source*.*
* No collaborating with other people; a **0** will be given if any collaboration evidence is found.
* I am looking for very specific, detailed, correct, and complete answers.
* Most answers found on the Internet (especially Wikipedia) are generic answers for people without any networking background and are not any of the above! Research each problem completely.
* Turn in a Word document or pdf into the Dropbox.

1. What is a network service model; discuss each of the services from network performance point-of-view.

The network service model helps define the end to end delivery of packets between hosts. Some of the services that can be included are: Guaranteed delivery, Guaranteed delivery with bounded delay, In-order packet delivery, Guaranteed minimal bandwidth, and Security.

Let's delve into what each service includes. Guaranteed delivery is the service that guarantees the source host’s packet arrives at the destination. The “Guaranteed delivery with bounded delay” not only guarantees that the packet will reach its destination, but will be delivered within a specified host to host delay bound. In-order packet delivery is the service that guarantees a packets arrival the destination in the set order they were set in. Guaranteed minimal bandwidth is the service that emulates the behavior of a transmission link at a specified bit rate for sending and receiving. As long as the host transmission rate for the bits are lower than that specified rate, they are guaranteed to be delivered. Finally is security, this service provides confidentiality at the source of the datagram by encrypting and decrypting them.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. Explain in detail how NAT works. Support your explanation with an example.

Here is an example of a NAT.  
Diagram, funnel chart

Description automatically generated

In this figure we can see the operation of a NAT-enabled router. On the right side we can see an interface that is part of the home network. In the figure above addressing is done with all devices having the same subnet address.

If the private address only has meaning inside the home network, how is it handled when the data is sent outside of the network? This is solved with NAT. The NAT-enabled router doesn’t look like a router to the outside world, instead it appears to the outside as a single device. Since it appears as a single device it only needs one IP address to talk to the outside world. In the example all the data that exits the home has an IP listed as 138.76.29.7 and all the traffic coming in has an IP of 138.76.29.7. Essentially the NAT hides the details of the home network from the outside networks.

One question is how does the router know which internal host to forward to if all the datagrams that enter have the same IP address? This is solved by the NAT translation table at the router, which includes port numbers and IP addresses in the table entries. This is an example of how that table may look using our figure. Suppose the internal user requests a webpage from an external server. The host assigns an arbitrary source port number and sends the datagram to the LAN. The NAT receives it and generates a new source port number(as long as it is not currently being used) 5001 and replaces the source IP with the WAN facing IP. Once the request comes back the router rewrites the destination IP and destination port number and forwards the datagram into the network to it’s desired host.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. Explain in detail the overall functionality of Network Layer and discuss two main functions of network- layer?

Diagram

Description automatically generated

The generalized form of this layer is that it is responsible for moving datagrams from one host to another. This layer provides the segments from the transport-layer and a destination similar to a letter and delivers the segment to the destination host. This layer’s protocols are the IP protocol and other types of routing protocols to ensure the segments reach their destination.

Now that we have a general idea of the network layer let’s discuss the two main functions of this layer in depth. These two methods are known as forwarding and routing. Starting with forwarding in the data plane, when data enters the router it needs to have a method of moving that data to the proper link. This is the job of forwarding! We can use an example figure to help further explain. A packet has arrived from the host H1 to the router R1. This packet must be forwarded to the next router in the path to H2.

Next we are going to look in the control plane where we handle routing. While working alongside forwarding the network layer must also determine a path for a packet to travel as it flows from sender to receiver, this is the job of the routing algorithm. While these two are similar, forwarding is the local action of transferring packets to the correct output link while routing is the network wide process that determines the packets to travel across end-to-end paths. Another note is that forwarding is commonly implemented on the hardware level whilst routing is usually held on the software level. By using both these methods together we can have them complement one another. We can see this in the example below, as packets move across the network, we use routing algorithms to help configure the local tables in the data plane for forwarding.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. State three types of scheduling policies and discuss each of them with a help of figure.

Diagram

Description automatically generated

The first scheduling policy I will talk about is a method called First-in-First-Out (FIFO). FIFO works by having packets that arrive wait in a queue before they are transmitted if the link is busy. This means that there is no sorting or priority for certain packets. Due to this factor if the queue becomes full then new incoming packets will be dropped from the buffer. This figure shows the operation without dropped packets. New arrivals (Blue) go to the back of the line and “wait their turn” in the buffer and are then pushed out of the link.  
Diagram

Description automatically generated  
The next chosen operation is “Priority Queuing.” Similar to FIFO , we are placing packets into queues based on when they arrived, but now we have ways to differentiate them based on what is more important. These categories can be configured by a network administrator to fit their needs. An example is that admin may configure the network to prioritize data that has network management information over user traffic. Now that the network is configured the system will transmit packets from the queue with the highest priority first then to the next highest, etc. We can see this in the figure, new arriving packets are classified into one of two groups(High and Low). The four packets in the high queue will be transmitted first then it will move to the two in the low queue.  
Diagram, schematic

Description automatically generated

The final policy I have chosen is called “Weighted Fair Queueing,” this discipline is a generalized version of the “round robin queueing.” Each packet that enters is classified into a special queue depending on their data. Once the packets have reached the queue the scheduler will server the queues in a circular motion. Using the example we can see our network has 3 distinct queues. Each type of queue has the same fair chance to push their information, but packets in those queues have to wait until the scheduler has reached them again. The WFC will start with packet 1 from queue 1 then transmit packet 1 from queue 2, then packet 1 from queue 3. After the last queue has transmitted it returns to the top queue and repeats the cycle until the buffer is empty. The difference between WFC and round robin though is that each class “may receive a differential amount of service in any interval of time.” Each class is denoted as “i” and is assigned a weight “w.” This guarantees each class to then receive a fraction of the service equal to wi/(∑wj) of the bandwidth. The worst case is that the denominator is over all classes, this means the class will at least have a throughput of R⋅wi/(∑wj) where “R” is equal to the transmission rate.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. Explain the concept of flow control and how it is implemented in TCP.

During a TCP connection both the receiver and sender set aside a receive buffer. When the connection receives bytes that are in the correct sequence they are then placed into the buffer, but it’s not instantaneously. If the buffer is too slow at reading the data the buffer may overflow. This is why TCP provides flow-control to its applications to assist in neutralizing this issue. This makes flow control a speed-matching service, which matches what the sender is pushing and the receiver is reading. Another note is that the sender can be throttled due to congestion in the IP Network; this form is mentioned as congestion control. Even though flow control and congestion control are similar, they take action based on different reasons. TCP controls flow by having the sender have a maintained variable also known as the “receive window.” This window is used to give the sender a general status of how much free space there is in the buffer at the receiver. Since TCP is full duplex each side of the connection has a set receive window.

To help understand receive window we will use an example where host A and host B are transferring a file over a TCP connection. Host B establishes a receive buffer with a size denoted as “RcvBuffer.” Periodically the host reads from the buffer denoted with the following variables: “LastByteRead” as the number of the last byte read from the data stream read from the buffer and “LastByteRcvd” as the number of the last byte in the data stream that was placed in the buffer. To prevent the buffer from overflowing we use the formula “LastByteRcvd-LastByte≤RcvBuffer.” The spare room for the receive window, “rwnd,” is listed as “rwnd=RcvBuffer-[LastByteRcvd-LastByteRead],” since the window is dynamic. Host B then tels host A how much spare room is left in the buffer. In response host A tracks two variables, “LastByteSent” and “LastByteAcked.” Then it calculates the difference of the two to know the amount of data that hasn’t been acknowledged. By keeping the value of unacknowledged data lower than the value of rwnd, host A ensures the buffer will not overflow host B’s buffer.

An issue to these plan is “what if the value of rwnd=0 since the buffer has become so full?” This then tells host A that the buffer is empty while it is not. Another assumption is what if host B has nothing else to send to A. Since the buffer doesn’t tell A the rwnd unless it needs to respond to a request there is no way to tell A the buffer is free. To combat this issue TCP requires A to continue to send 1 byte data to the buffer, so that when the buffer has finally emptied then A will be told.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. Explain in detail how additive increase and multiplicative decrease in TCP congestion control works.

Chart, line chart

Description automatically generated

When talking about TCP congestion control it is common to hear it referred to as AIMD. AIMD stands for additive-increase, multiplicative-decrease which brought the rise of the “saw tooth behavior” in the figure.This behavior is a nice representation of TCP probing the bandwidth to increase the congestion control window size until it has received a “triple duplicate-ACK” event. The window size decreases by a factor of two and proceeds to probe the bandwidth again to find available bandwidth. This algorithm was developed and serves as an “asynchronous-optimization algorithm” that optimizes several important pieces of the user and network performance.

Let’s take a deeper look into this algorithm. When we have a window size of “w” and a round-trip time of “RTT” the transmission rate is roughly w/RTT. TCP proceeds to probe the bandwidth for additional space by increasing “w” by 1 mss each RTT. This will continue until the TCP has encountered a packet loss event. We now denote the value of “w” when this loss occurred as “W” and continue to the next part of the equation. Assuming both factors of RTT and W were constant for the connection the transmission rate range will change from W/(2\*RTT) to W/RTT. This hen means the average throughput of a connection is .75\*WRTT.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

1. Suppose that TCP's current estimated values for the round trip time (estimatedRTT) and deviation in the RTT (DevRTT) are 250 msec and 15 msec, respectively. Suppose that the next three measured values of the RTT are 280, 260, and 300 respectively.

Compute TCP's new value of estimatedRTT, DevRTT, and the TCP timeout value after each of these three measured RTT values is obtained.Use the values of α = 0.125 and β = 0.25.

First ERTT-250

First DevRTT-15

RTT samples-> 280, 260, 300

α = 0.125 and β = 0.25.

Estimated RTT = (1- α) \* Estimated RTT + α \* Sample RTT

DevRTT = (1- β) \* DevRTT + β \* |Sample RTT - Estimated RTT|

Time-out Interval = 4 \* DevRTT + Estimated RTT

RTT=280

EstimatedRTT = (1 - .125) \* 250 + .125 \* 280 = 253.75ms

DevRTT = (1 - .25) \* 15 + .25 \* | 280- 253.75 | = 17.8125ms

Time-out = 4 \* 17.8125 + 253.75 = 325ms

RTT=260

EstimatedRTT = (1 - .125) \* 253.75 + .125 \* 260 = 254.53125ms

DevRTT = (1 - .25) \* 17.8125 + .25 \* | 260- 254.53125 |= 14.7265625ms

Time-out = 4 \* 14.7265625 + 254.53125 = 313.4375ms

RTT=300

EstimatedRTT = (1 - .125) \* 254.53125 + .125 \* 300 = 260.21484375ms

DevRTT = (1 - .25) \* 14.7265625 + .25 \* | 300 - 260.21484375 | = 20.9912109375ms

Time-out = 4 \* 20.9912109375 + 260.21484375 = 344.1796875ms

“How to Compute Devrtt, Estimated RTT, &amp; Time-out Interval in CCN.” Educative, https://www.educative.io/answers/how-to-compute-devrtt-estimated-rtt-time-out-interval-in-ccn.

1. Explain the concept of pipeline, and how GBN protocol uses the concept pipeline.

To understand the concept of pipeline we need to look at the history before it. In protocol rdt3.0 was functionally sound, but would not keep up in today’s networks. The main flaw was the transfer of packets and their acknowledgements. Rdt3.0 used a method called “stop-and-wait,” this protocol would send packets over the network one by one and wait for the acknowledgement to arrive before the next packet would send. This caused long delays and slowed down the transmission speed of the network. That is why the pipeline protocol was created. The main idea of the pipeline is to send each packet, but not wait on their acknowledgement.

A picture containing diagram

Description automatically generated

Diagram

Description automatically generated

Figure A is an example of the stop-and-wait operation, while figure B is the pipeline operation. The key feature is the amount of data being pushed out in pipeline compared to stop-and-wait, but this comes with a few issues. The range of sequence numbers must be increased and each packet needs their own unique sequence number. The consequence to this is that there may be multiple in-transit unacknowledged packets. Next is that the sender and the receiver sides may have to buffer more than one packet. Third is the manner of how packets that are lost, corrupted, or delayed are handled and recovered. One of those approaches can be identified as “Go-Back-N.”

In the “Go-Back-N” protocol the sender is allowed to transmit multiple packets without waiting for an acknowledgement. The constraint to that is that the sender has a set maximum of unacknowledged packets known as “N” or the window size, this is why the GBN is also referred to as the “sliding window protocol.” In this protocol we define “base” to be the sequence number of the oldest unacknowledged packet and “nextseqnum” to be the smallest unused number. These then create a range of four numbers that can be identified. [0, base-1] is the interval of corresponding packets that have already been transmitted and acknowledged. [base, nextseqnum-1] is the interval of packets that have been transmitted, but haven’t received an acknowledgement. [nextseqnum, base+N-1] is the interval of packets that can be sent immediately. Finally numbers greater than or equal to base + N cannot be used until an unacknowledged packet is fulfilled.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

9. Explain in detail, how smapleRTT and EstimatedRTT are used to calculate RTT?

When sending packets in the networking world we need to be able to control how long the packet is allowed to live for an acknowledgement before the packet is resent. To help decide this we need to look at historic data in the router and these include sampleRTT and estimatedRTT. These two factors can then help calculate the devRTT and the time-out period the packet is allowed to have.

Starting with the sampleRTT and the estimatedRTT this is how they correlate to calculating RTT. The sampleRTT is a segment amount of time from when the segment is sent and when the acknowledgement is received. This information fluctuates with congestion, so we use the data to help calculate an estimated RTT. The algorithm for this is listed as EstimatedRTT=(1−α)⋅EstimatedRTT+α⋅SampleRTT. Every new sampleRTT then updates the estimatedRTT. From there we can then start calculating deviations from our estimatedRTT.

The devRTT is the deviation from the estimatedRTT. By calculating this information we can add the deviation time to our estimate to give a reasonable period of time before we time-out the segment. The algorithm for devRTT is listed as DevRTT=(1−β)⋅DevRTT+β⋅|SampleRTT−EstimatedRTT|.

Now that we have our devRTT and our estimatedRTT we can now calculate the time-out. This time out is listed in the algorithm TimeoutInterval=EstimatedRTT+4⋅DevRTT. By adding 4 times the deviation to our estimate we create a reasonable net for our segment to travel through congestion and for us to receive the acknowledgement. All these factors all come together to help calculate RTT and create padding for future segments that come into the system.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.

10. Explain, how destination base forwarding gets impacted by input port forwarding.

Diagram

Description automatically generated

To help explain this problem let’s start with an analogy to help represent what destination-based forwarding is. We will assume packet forwarding is similar to cars entering and leaving a roundabout. Suppose when the car that enters the roundabout states its final destination outside of the interchange. The attendant will then look up the final destination and relay that back to the driver on which exit to take to reach their destination.

Now that we understand the idea of destination forwarding we can look at how input port processing affects it. With input processing in effect, when data enters the router the data is analyzed and the router looks up on the forwarding table which output port the packet should be used to push to the switch fabric. We can see a generalized version of the operation in the figure above. Essentially input port processing acts as the attendant for the packet to communicate in the roundabout and sets the exit route for the car.

Graphical user interface, application

Description automatically generated

We can continue to look at this process with the following figure. When the router finds a prefix using the longest prefix matching rule that matches the packet's destination in the table, it then will forward the packet to the associated link.

Kurose, J. F., &amp; Ross, K. W. (2017). In Computer networking: A top-down approach (7th ed.). essay, Pearson.