CPE-490 Midterm

Joshua Schmidt

I pledge my honor that I have abided by the Stevens Honor System. - Joshua Schmidt, 10/16/2019

Theory based questions

(answer the following in a paragraph or two, in less than 20 lines of text) [10 points per questions]

1. Briefly explain the five layers of the Internet, including: Application, Transport, Network, Data-link and Physical. How does the simple act of "checking email" make use of these five layers?

The application layer is the top layer, which is directly visible to the user and is the layer the user interacts with. Some examples of this include Spotify, YouTube, and Google - any user-facing application that accesses the internet. In the case of checking email, the email message would be queried through an api typically with an http get request, which would then go through all 5 of the layers of the internet, route to the server, and pop back up through the 5 layers in reverse order to the application on the server side. The response will then be sent in the same fashion to the client, but this time the message will contain the email data.

The second layer is the transport layer, which communicates with the network layer and implements what is needed for end-to-end communication. Data is split and sent in packets over the transport layer, with the two most common protocols being Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). These protocols handle error checking and the connection between the client and the server. TCP is typical for reliable, guaranteed transmission, while UDP is used when speed is valued over having some error. It is in the transport layer that the TCP port is specified.

The next layer down is the network layer, which manages routing packets between different nodes in the network. The network layer encapsulates the packets from the transport layer in another layer of packets. It determines the best path to deliver the packets, assigning Internet Protocol (IP) packets to the source and destination.

Next is the data-link layer, which encapsulates the packets from the network layer in another packet. The data-link layer assigns a MAC address to the new data packets, which are essentially unique identifiers for devices within a network. There is a source and destination MAC address so the physical layer knows where the packets need to go.

The physical layer handles transmitting the raw bits over a physical connection and communication link. The data sent would be the raw bits from the packet created in previous layers, ultimately including the email, encoded and compressed in the body of the packet. This layer includes the physical hardware used to send the data.

2. Which layers are "hop-by-hop" and which are "end-to-end"? What's the difference between these two modes?

Hop-by-hop layers happen at every node in the network. These include physical and data-link layers. The end-to-end layers happen only at the start and end nodes in the network - the source and the destination. These include the application, transport, and network layers.

In the end-to-end network framework, the features specific to the application are handled only at the end nodes of a network. The nodes in the network between the two ends do not do much computation and instead just pass the data along from the sender to the receiver. The main advantage of this framework is that it allows for intermediary nodes to go offline or be swapped without noticeable impact to the end user. However, if one of the ends disconnects from the network, the application will not work.

In the Hop-by-hop transport network framework, it is reliant on intermediary nodes to send data from the source to the destination. The main advantage of hop-by-hop is that only the current and next node have to be connected in the network. So if the destination node disconnects from the network, the data can still flow though the network and if the destination connects again, it will be able to receive the packets.

3. What are the major differences between TCP and UDP? Give an example application each that is best served by TCP and UDP.

The main difference between TCP and UDP is TCP guarantees the delivery of data to the destination router. UDP cannot guarantee this delivery. Additionally, TCP is connection-oriented, meaning the communicating devices establish a connection before they start to transmit data, and close the connection after they finish transmitting. UDP on the other hand does not do anything for starting, maintaining, or closing a connection. UDP is simpler and faster than TCP, TCP retransmits packets if they are lost, TCP has a much longer header than UDP (20-80 bytes vs 8 bytes), and UDP supports broadcasting whereas TCP does not. Live video and audio streaming would be best served with UDP because it is more important to have a continuous, live stream instead of a choppy, delayed high-quality stream. Web pages are better served with TCP because it is more important that the page comes in as one piece, with all of the formatting intact, so that the browser can properly render the html and javascript.

4. How does TCP estimate round-trip-times (RTT) despite it being impossible to measure one way latency on the Internet? Expand on the idea of EWMA and it's application to RTT estimation.

It is impossible to measure one way latency on the Internet because once the packet is sent and received at the destination, it takes time for the node at the destination to send a packet to the receiver saying that it received the original packet. To remedy this, TCP estimates the round-trip-times using an exponential weighted moving average (EWMA), essentially using a history of delays and the most recent delay to estimate the current actual delay. When the TCP connection is established, the sender node sends a packet to the receiving node and starts a timer, which will timeout when it reaches an interval value. Each packet has a sequence number, and when the sender receives an `ACK` (acknowledgement) for a packet, it stops that specific timer (sample_rtt). The original estimated_rtt is equal to this sample, but subsequent estimations are calculated with $estimated\_rtt = \alpha \cdot estimated\_rtt + (1-\alpha) \cdot sample\_rtt$. The value of α can vary but the most common is $\frac{7}{8}$, meaning 87.5% of the weighted average value comes from the estimated rtt and 12.5% comes from the current rtt.

5. How does TCP ensure reliable data delivery? Expand on the following TCP concepts: (i) Stop-and-wait, (ii) Go-back-N, and (iii) TCP-SACK.

TCP uses several methods to ensure the data is delivered, with sufficient error checking such that the probability of all the methods failing together is very low. The first method is stop and wait, where the sender sends the packet and waits for an acknowledgement (`ACK`) for the packet. When the sender receives the `ACK`, it can then send the next packet. If the sender does not receive the `ACK`, it sends the same packet again. The second method is Go Back N. In this process, the sender sends N packets (N is the window size), and when these packets are sent, the sender waits for a cumulative acknowledgement (`ACK`) before sending more packets. The receiver receives only in-order packets and ignores packets that are out of order. If there is any packet loss, the whole window is transmitted again. The last method is Selective Acknowledgements (TCP-SACK). In this process, the receiver acknowledges all packets received, regardless of if they were received in-order or not. For any packets not received, the sender will send those again. TCP also uses checksums to ensure that data is not corrupted in the packet itself. With these methods, TCP is able to ensure near-perfect data delivery.

6. Describe the AIMD congestion control algorithm employed by TCP. What is the specific need to increase "additively" and decrease "multiplicatively"?

The Additive Increase / Multiplicative Decrease (AIMD) congestion control algorithm is used to limit the amount of data being transferred at a given time, to leave bandwidth for others on the network, and use more bandwidth when the network has less congestion. To calculate the current "max window", or the amount of bandwidth that the sender can use without saturating the network, the sender takes the minimum of the advertised window - the window for the receiving side, and the congestion window - the window for the sending side. The receiving side provides the advertising window to the sender, and the sender can get the congestion window based on the amount of congestion it perceives in the network, increasing the window when congestion goes down and decreasing it when congestion goes up. The sender determines the congestion window based on timeouts, so whenever a packet is lost the congestion window halves (decreasing multiplicatively) and whenever the sender receives an `ACK`, the congestion window increases by one packet in size (increasing addatively). The decrease is multiplicative because the sender is more willing to reduce its congestion window than to increase it, since the consequences of having a window that is too large are much higher than of having a smaller window. The resulting bandwidth vs time graph looks similar to a sawtooth wave.

7. Describe the "three-way-handshake" used by TCP to establish a connection.

A three-way handshake is the algorithm used by TCP to create and end a connection, and like the name implies, involves exchanging three messages between two parties. For TCP, it is used so that the two participants can agree on the starting sequence numbers for their byte streams, but the handshake can really be used for agreeing upon any set of parameters for communication. In the first step, the client sends a message to the server stating the initial sequence number that it will be using (`SYN = a`). The server then responds with a segment that acknowledges it received the client's sequence number (`ACK = a + 1`) and adds it's own sequence number (`SYN = b`). Then the client responds with acknowledging the server's sequence number (`ACK = b + 1`). Timeouts are used to ensure that if the segments are transmitted and the acknowledgements are not received within a given amount of time, the sender will retry. The acknowledgement includes the increment of the given synchronization number because this implies the receiver understood the message.

8. What is the difference between medium access that is time-based (TDMA) and carrier-sense based (CSMA/CD)?

TDMA and CSMA/CD are two different methods for multiple devices to share a single transmission medium and share bandwidth. With Time Division Multiple Access (TDMA), every node in the network is set a specific amount of time to send and receive data. Everyone in the network is equal, with the high bandwidth nodes having the exact same amount of time to send and receive data as the nodes requiring little bandwidth. Carrier Sense Multiple Access with Collision Detection (CSMA/CD), on the other hand, takes a more active approach that allows for greater efficiency. Every node in the network can listen to the link and determine if it is in use or is idle, and can check if two nodes are transmitting on the link simultaneously (collision detection). There are several algorithms to avoid collisions and find the right time for each node to transmit data. One example is the back-off algorithm, which is used in ethernet to reschedule transmissions after collisions. Both TDMA and CSMA/CD are focused on avoiding collisions and allowing each node in the network to have a turn. TDMA focuses on a simple time scheduling, while CSMA/CD focuses on measuring the amount of bandwidth that every node uses and creating a schedule based off of that. CSMA/CD therefore has on average a higher throughput than TDMA.

Plot this on a map

A traceroute to a remote destination in Osaka, Japan yielded the following router codes. Describe the cities that were traversed to reach Osaka from Hoboken, NJ. Do you think this is a direct path that a flight would take from here to there?

mco.454a.lightpath.net	Orlando International Airport
atl.as32112.level3.net	Atlanta International Airport
stl.as39887.level3.net	St. Louis International Airport
sat.as889.telegrid.net	San Antonio International Airport
pdx.as399.level3.net	Portland International Airport
Itm.7689.bigglobe.jp	Osaka International Airport

You can often determine the cities that the packet passed through based off of the first letters of the router codes, which correspond to the nearest airport. So the packet started in Hoboken, then went to Orlando, then to Atlanta, then to St. Louis, then to San Antonio, then to Portland, and then travelled across the Pacific to finally arrive in Osaka. This is not a direct path, as the packet took many intermediate steps to get across the continental US before going to Osaka. But it is relatively direct, as there are relatively few hops (5) to get across the US, spanning a large distance (over 3000 miles). A more direct path that I may take on a plane would be to go from JFK to Portland and then to Osaka.