Kate Moreland

kem0149

COMP 4320 – Assignment (2)

**Problem 1.**

Imagine you are responsible for improving the efficiency and performance of a large-scale web service that serves millions of users globally. Using your understanding of web cache, HTTP versions, email protocols, and the Domain Name Service (DNS), answer the following questions:

1. **Explain how web caching (proxy servers) can be used to improve website performance and reduce network congestion? Discuss the trade-offs between increasing access link speed and deploying caching solutions. Which approach would you prioritize and why?**

Web caching involves storing frequently accessed content closer to the user, which speeds up access by reducing the need to retrieve it from the original server. This helps minimize latency since cached content can be delivered almost instantly, rather than traveling long distances from the source. It also reduces the load on the origin server, allowing it to handle other requests more efficiently. While increasing access link speed improves direct user-server communication, it doesn’t address redundant data transfer or lighten the server’s workload. Caching is used to reduce the number of requests that are traveling across the network. Due to all of these factors, caching is the better option for scalability and efficiency.

1. **Compare HTTP/1.1 and HTTP/2 in terms of performance improvements and head-of-line (HOL) blocking mitigation. Discuss the role of SMTP, IMAP, and HTTP-based email services in email transmission and retrieval. How do these protocols differ in function and efficiency?**

HTTP/1.1 uses a single connection for each request, it also equipes head-of-line (HOL) blocking. It also uses persistent connections and pipelining. However, it still experiences inefficiencies. HTTP/2 uses a single multiplexed connection, which allows for multiple requests and responses to be handled at the same time. HTTP/2 also removes HOL blocking at the application layer though stream prioritization and header compression (HPACK), reducing overhead and latency. SMTP (Simple Mail Transfer Protocol) is mostly used for sending emails. It is pushed based and works with store-and-forward mechanisms. IMAP (Internet Message Access Protocols) is used for getting emails while they remain on the server. It is more efficient than similar protocols because it works with synchronization and folder management. HTTP-based email uses web APIs to send and get emails. This is much more accessible than IMAP and SMTP and is optimized for modern web applications. HTTP-based email services offer better scalability and integration with cloud services, whereas IMAP provides better real-time synchronization.

1. **Describe the DNS resolution process when a user accesses a website. Include how iterative and recursive queries work. What are the scalability challenges of the centralized DNS model, and how does the hierarchical DNS structure help mitigate these issues?**

The DNS resolution process starts off with a user typing a certain URL in a browser. The request is then checked in the local DNS server. It is then checked to see if the cache is there. If the cache is not there then it directs it to the TLD (Top-Level Domain) server. The Top-Level Domain server points to the authoritative DNS server for the URL. The authoritative server turns around and returns the IP address for the URL. Then the connection with the IP is sent. When an iterated query is involved the local DNS server reaches out to the next server. The server then sends the name of the next server to contact if they do not have the IP. When a recursive query is involved, the request goes down the line of servers and does not have to go back to the local DNS server. When a centralized DNS model has an issue or a failure at one it is not able to go around to another server so they are not able to successfully retrieve the IP. The latency is increased because there are only a few servers. The DNS is across multiple layers including root, TLD, and authoritative server. Using caching reduces the total number of queries that have to go all the way to the authoritative server. The request goes to the geographically closest server which improves response times.

**Problem 2.**

Imagine you are responsible for designing a high-performance transport-layer protocol for a new real-time communication application. Using your understanding of TCP, UDP, multiplexing/demultiplexing, and reliable data transfer (RDT) protocols, answer the following questions:

1. **Discuss whether TCP or UDP would be more suitable for your application. Explain the trade-offs between connection-oriented vs. connectionless communication and how your choice impacts reliability, delay, and congestion control.**

UDP would be the better choice in this situation due to it being a real-time communication application. UDP is a no guarantee, which means that packets are not guaranteed to make it or they could be out of order. It is also labeled as a connectionless communication. TCP on the other hand makes sure that all the packets arrive. TCP is known as a connection-oriented communication. The latency in UDP is lower because it has the ability to send the packets without any delays. TCP on the other hand has a higher latency due to handshaking and retransmission. UDP lacks a built-in congestion control. This may result in some packet loss but helps with cutting down delays. TCP takes advantage of mechanisms like slow start as well as congestion avoidance. These are helpful in ensuring that packets are delivered but come with delays. UDP is more suitable for this application because it puts low latency over reliability, while TCP does the opposite.

1. **Describe how multiplexing and demultiplexing work in the transport layer. Explain how connectionless demultiplexing (UDP) differs from connection-oriented demultiplexing (TCP) and why it matters for your application.**

Multiplexing lets many different applications to use the same network connection through assigning different port numbers. Demultiplexing guides packets that are being received to the certain application on the destination port. UDP solely uses destination IP as well as port number. UDP does not use session tracking so each packet is unique. This method creates simplicity but does not always provide reliability. TCP has a 4-tuple which includes source IP, source port, destination IP, and destination port. The connection is on throughout the process. This ensure that all packets arrive and in the correct order. UDP is necessary for this application because everything is happening in real-time and latency needs to be low. Also, a connection does not have to be maintained which allows for a simpler model to be implemented.

1. **Describe how stop-and-wait (rdt2.0) and automatic repeat request (ARQ) protocols help ensure reliable data transmission. Discuss how modern transport-layer protocols handle packet loss, duplicate packets, and corrupted ACKs/NAKs, and how you would integrate these features into your protocol design.**

The stop-and-wait (rdt2.0) makes sure that there is a reliable transmission through only sending a single packet at a time. It then waits for the ACK before sending the next packet. This makes sure that packets get delivered because if an ACK does not come back then they can resend the packet. Automatic repeat request (ARQ) protocol prioritizes efficiency because they send off many packets before waiting for the ACK. The Go-Back-N makes it where the sender can send off many packets but also makes them retransmit all of the packets after a single packet that didn’t make it. The selective repeat only focuses on the lost packets and only retransmits them. When packet losses occur the use of FEC (Forward Error Connection) can go and get lost packets without the use of retransmission. When there are duplicate packets there can be sequence numbers to make sure they do not get out of order and ones that don’t fit can be thrown out. Checksums can use to check to see if there are any corrupted ACKs/NAKs and then see if there can be a retransmission to fix the issue.

**Problem 3.**

Consider the figure below in which a TCP sender and receiver communicate over a connection in which the sender-to-receiver segments may be lost.

The TCP sender sends initial window of five segments at t=1,2,3,4,5, respectively. Suppose the initial value of the sender-to-receiver sequence number is 118 and the first five segments each contain 506bytes. The delay between the sender and the receiver is 7 time unites, and so the first segment arrives at the receiver at t=8.

As shown in the figure, two of the five segment(s) are lost between the sender and the receiver.

A diagram of a graph

Description automatically generated

1. **Give the sequence numbers associated with each of the five segments sent by the sender**

Seg #1 = 118

Seg #2 = 118 + 506 = 624

Seg #3 = 624 + 506 = 1,130

Seg #4 =1,130 + 506 = 1,636

Seg #5 = 1,636 + 506 = 2,142

1. **List the sequence of acknowledgments transmitted by the TP receiver in response to the receipt of the segments actually received. In particular, give the value in the acknowledgment field of each receiver-to-sender acknowledgment, and give a brief explanation as to why that particular acknowledgment number value is being used.**

**You can answer the previous questions by filling the table below:**

Time Segment Received:

Seg #1 = 1 + 7 = 8

Seg #2 = 2 + 7 = 9

Seg #3 = The packet never made it.

Seg #4 = The packet never made it.

Seg #5 = 5 + 7 = 12

Receiver-to-sender ACK field value:

Seg #1 = The packet made it and is sending back what the next packet should be. = 624

Seg #2 = The packet made it and is sending back what the next packet should be. = 1,130

Seg #3 = The packet never made it so there is no ACK value that is sent back. = N/A

Seg #4 = The packet never made it so there is no ACK value that is sent back. = N/A

Seg #5 = The packet made it and is sending back what the next packet should have been but 1,130 got missed so they send it back again to tell that it was missed. = 1,130

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sender-to-Receiver | Time segment sent | Sender-to-receiver segment sequence# field value | Time segment received, and ACK segment sent | Receiver-to-sender ACK field value |
| Seg#1 | 1 | 118 | 8 | 624 |
| Seg#2 | 2 | 624 | 9 | 1,130 |
| Seg#3 | 3 | 1,130 | N/A | N/A |
| Seg#4 | 4 | 1,636 | N/A | N/A |
| Seg#5 | 5 | 2,142 | 12 | 1,130 |

**YouTube Links**

Problem 1: <https://youtu.be/kWA7FChV2RA?si=2wC2zNhdd15pyHSU>

Problem 2: <https://youtu.be/HogD-A4zLpc?si=sfzDocBaDFm0Stp7>

Problem 3: <https://youtu.be/gbY5ZBgSdDg?si=lzkI8dDQsN_4W5If>