Nathan Fox

CS 460

February 3, 2015

Applications Homework

1)

a. Parallel downloads via parallel instances of non-persistent HTTP do not make sense in this case. Retrieving 11 objects from the same sender over a non-persistent connection requires performing the handshake 11 different times, once for each TCP connection. If there were five parallel connections, each connection would get only 30 bps of the link bandwidth. The control packets of 200 bits for the handshake would require a total of 6.67 seconds of transmission delay. The handshake is in three parts, making a total of 20 seconds of transmission delay for each handshake. Five of the 11 files could be sent at the same time in parallel, but on a 30 bps link a 100 Kbit file would take about 3333 seconds (about 55.5 minutes). Three sets of five parallel requests would be required to get 11 files, resulting in 3 \* (20 + 3333) = 10060 seconds, or about 167 minutes of transmission delay.

b. Persistent HTTP would result in significant gains. Assuming a single TCP connection, the handshake would require about 1.3 seconds of transmission delay, and then sending 11 100-Kbit packets would add about 7333 seconds of transmission delay, making a total of about 122 minutes. Using persistent HTTP in this case would save about 45 minutes of transmission delay.

2)

a. Bob's parallel connections do help him to get web pages more quickly because each connection gets the same percentage of the bandwidth. If Bob makes 4 parallel connections and the other 4 users each have a single persistent connection, Bob gets half of the bandwidth while the other users each get an eighth.

b. If all five users use parallel connections, then Bob's parallel connections would no longer be beneficial. If each of the 5 users makes the same number of parallel connections, then each user gets a fifth of the bandwidth. They would all be better off if they all used persistent connections, because they would still each get a fifth of the bandwidth and they wouldn't have to deal with the overhead from performing the handshake multiple times.

3)

No. You may be able to tell if the DNS records are cached by measuring the speed of the response, but cached records are often kept for a couple of days before being discarded. If you can tell that your local DNS has cached a certain domain, then you only know that somebody accessed some site from that domain within the past couple of days. There is no way for an ordinary user to know from the DNS server exactly when the domain was cached, and there is no way for an ordinary user to know the exact page that was visited on that domain.

4)

a. Bob's claim is possible. He can receive a complete copy of the file without uploading anything, but it may take him a lot of time. He would never get on a neighbor's "top 4" list, because the "top 4" list consists of the peers that upload to you at the highest rate. But every 30 seconds, peers in the system randomly choose a peer to upload to. These random peers are said to be "optimistically unchoked." This allows a new user a chance to get file chunks that he/she can upload to other users in the hopes of getting on other users' "top 4" lists, which greatly increases the likelihood that the user can download entire files. Bob may be able to download an entire file by randomly being "optimistically unchoked" by various users who each have chunks of the file, but it might take Bob a long time if he doesn't earn any benefits by uploading to others.

b. Using multiple computers would make Bob's free-riding more efficient. Using more computers means that his computers are more likely to be optimistically unchoked, so he would be able to get more chunks of the file faster. Once he had all of the file chunks, he could transfer them all to a single machine to reassemble the file.

5)

Once peer 5 leaves, peer 4 will update its successor information, making peer 8 its first successor and peer 10 its second successor. Peer 3 could then query peer 4 for peer 4's first successor, which would become peer 3's second successor. Peer 3's first successor would remain as peer 4, and its second successor would become peer 8.

6)

1. Peer 6 sends a message to Peer 15 asking about information on Peer 6's predecessor and successor

2. Peer 15 forwards the message to Peer 1 => Peer 3 => Peer 4 => Peer 5

3. Peer 5 realizes that it will be Peer 6's predecessor, and that its current successor (Peer 8) will become 6's successor

4. Peer 5 relays the information back to Peer 6

5. Peer 6 makes Peer 8 its successor and notifies Peer 5 that it should change its immediate successor to 6

7)

Unlike unsolicited advertisements that a user must encounter to use services provided by Google, Facebook, Yahoo, etc., spam imposes a negative externality without providing any benefit to the user. A large percentage (estimated around 88%) of all sent email worldwide is spam, giving spam the potential to make email completely useless. A lot of time and money (estimated at $20 to $50 billion per year) are used to delete spam and to improve anti-spam technologies. As spam-

blocking technologies improve, so do spam-sending technologies, which results in an endless race between spam senders and spam blockers. Larger email services spend millions of dollars to improve anti-spam technologies, making it difficult for smaller email providers to compete in the market.

Due to the large amounts of time and money spent to filter spam, spam has a non-trivial effect on the economy. Spam has a very large ratio of the cost to society relative to private benefits. Spammers are estimated to generate $160-360 million per year, while an estimated $14-18 billion are spent each year in an attempt to block, filter, or delete spam, resulting in a ratio of about 100:1. Various legal policies have been adopted in the United States as an attempt to stop spammers, but such legal policies have no effect on international spammers, and even domestic spammers can be difficult to pin down. Economic proposals have been made that involve taxing those who send email or taxing those who encourage spammers by purchasing their goods. These solutions, however, may be difficult to enforce or may prevent companies or non-profit organizations from sending emails that are legitimate and useful. Another economic solution is using an attention bond, where an email sender pays the recipient for the attention. Recipients can then choose to whitelist senders so that specific senders can send to them for free. While this would disincentivize spammers without stopping legitimate mail, it would be difficult to link email accounts with payment mechanisms. Also, account hijacking could lead to theft. In essence, there is no solution elegant enough to stop spammers without creating the potential to also harm legitimate organizations or innocent users.

My preferred solution to dealing with spam is the proposal made by Randall Lewis to spam the spammers by placing fake orders on spam-advertised stores. Merchants would have a harder time filling orders, and they would have to pay bank fees to submit payment authorization requests that would end up being invalid. Spamming the spammers in this way would raise the costs for spammers and disincentivize them from using spam, and the current functionality of email could remain unchanged. The biggest downside is that legitimate merchants sending legitimate emails could be falsely identified as spammers, and they could be unjustly penalized.

8)

The problem with using incentives is that new users who have nothing to upload can't earn the incentives needed to download, leaving new users in a state where they can't upload or download anything. An ideal solution would be one that allows users to download portions of files without uploading, but makes downloading much easier (or faster) for users who do contribute by uploading. One solution would be to have a centralized database that keeps track of how much users upload and download. New users are allowed to download a small number of files without uploading anything, but then would have to upload in order to earn further downloading privileges. Free-loaders could download their first few files for free, but would then have to share if they wanted to upload more. The major drawback of this solution is that it would add a centralized component to an otherwise distributed system, creating a potential single point of failure.