Name: CENSORED

Date: 7/24/19

Assignment: HW2

Class: COMP-4320

1. Consider an HTTP client that wants to retrieve a Web document at a given URL. The IP address of the HTTP server is initially unknown. What transport and application-layer protocols besides HTTP are needed in this scenario?

**Answer:** The only other application-layer protocol that is necessary in this scenario is DNS, so that the client can perform a DNS lookup on the server to receive the IP address. The transport layer protocols that are needed are UDP for DNS (so that DNS lookups can be fast and efficient, since data loss is not an issue in this case) and TCP for HTTP (so that the web document can be sent back to the client with minimal to no data loss).

1. Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not cached in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that *N* DNS servers are visited before your host receives the IP address from DNS; the successive visits incur an RTT of *RTT1, …, RTTN*. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let *RTTs* denote the *RTT* between the local host and the server containing the object. Assume that the transmission time of the object is *Ti*. How much time elapses from when the client clicks on the link until the client receives the object.

**Answer:** The total amount of time it takes to receive the IP address from the multiple DNS servers is RTT1 + RTT2 + … + RTTn. After getting the IP address, two RTTs’s are needed to receive the file; this is because one RTTs is used to establish the TCP connection and another RTTs is used to request/receive the file. Therefore, the total response time is 2RTTS + RTT1 + RTT2 + … + RTTn. Since there is only one object to download, and this one object has transmission time Ti, the final amount of time it takes for the client to receive the object is Ti + 2RTTs + RTT1 + RTT2 + … + RTTn.

1. Referring to Problem 2 above, suppose the HTML file references four very small objects on the same server. Neglecting transmission time, how much time elapses with
2. Non-persistent HTTP with no parallel TCP connections?

**Answer:** RTT1 + … + RTTn + 2RTTs + (4 \* 2RTTs) = 10RTTs + RTT1 + … + RTTn

1. Non-persistent HTTP with the browser configured for 6 parallel connections?

**Answer:** RTT1 + … + RTTn + 2RTTs + 2RTTs = 4RTTs + RTT1 + … + RTTn

1. Persistent HTTP? (Assume that pipelining is used.)

**Answer:** RTT1 + … + RTTn + 2RTTs + RTTs = 3RTTs + RTT1 + … + RTTn

1. Consider a short, 90-meter link, over which a sender can transmit at a rate of 420 bits/sec in both directions. Suppose that packets containing data are 320,000 bits long, and packets containing only control (e.g. ACK or handshaking) are 240 bits long. Assume that *N* parallel connections each get 1/*N* of the link bandwidth. Now consider the HTTP protocol, and assume that each downloaded object is 300 Kbit long, and the initial downloaded object contains 6 referenced objects from the same sender. Would parallel download via parallel instances of nonpersistent HTTP make sense in this case? Now consider persistent HTTP. Do you expect significant gains over the non-persistent case? Justify and explain your answer.

**Answer:** Let Tprop denote the propagation delay between the client and the server. Lets first consider parallel instances of non-persistent HTTP. The total time needed to receive all objects is given by (240/420 + Tprop + 240/420 + Tprop + 240/420 + Tprop + 300000/420 + Tprop) + (240/(420/6) + Tprop + 240/(420/6) + Tprop + 240/(420/6) + Tprop + 300000/(420/6) + Tprop) = 5012 + (8 \* Tprop) seconds. Now let’s consider a persistent HTTP connection. The total time needed to receive all objects is given by (240/420 + Tprop + 240/420 + Tprop + 240/420 + Tprop + 300000/420 + Tprop) + 6 \* (300000/420 + Tprop) = 5001.714 + (10 \* Tprop) seconds. Tprop is completely negligible in comparison with the large transmission delay, so we will only pay attention to the resulting numbers in the comparisons. Thus, we see that persistent HTTP is slightly faster (~10 seconds faster) than the non-persistent case with parallel download.

1. Consider the scenario introduced in Question (4) above. Now suppose that the link is shared by Tom with seven other users. Tom uses parallel instances of non-persistent HTTP, and the other seven users use non-persistent HTTP without parallel downloads.
2. Do Tom’s parallel connections help him get Web pages more quickly? Why or why not?

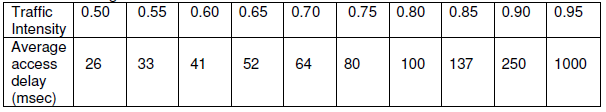
**Answer:** Yes, since Tom’s parallel connections improve the response time that Tom gets from the web pages, since he can receive multiple objects from the server at once. The other users are forced to share a single serial connection to download their objects (one at a time), and therefore have smaller response times than Tom.

1. If all eight users open parallel instances of non-persistent HTTP, then would Tom’s parallel connections still be beneficial? Why or why not?

**Answer:** It is still beneficial for Tom to have parallel connections regardless if all users have parallel connections. Even if all users use parallel connections, this does not impede the performance times for any of the users.

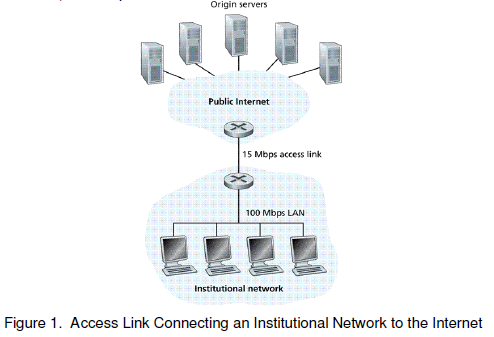
1. Consider Figure 1 in which there is an institutional network connected to the Internet. Suppose that the average object size is 675,000 bits and that the average request rate from the institution’s browser to the origin server is 20 requests per second. Also suppose that the amount of time it takes from when the router on the Internet side of the access link forwards an HTTP request until it receives the response is 2.0 seconds on average. Model the total average response time as the sum of the average access delay (that is, the delay from Internet router to institution router) and the average Internet delay.

The average access delay is related to the traffic intensity as given in the following table.



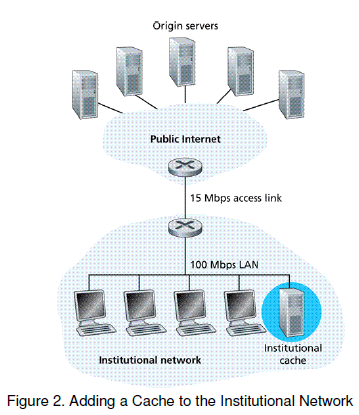
Traffic intensity is calculated as follows: Traffic intensity = *aL/R,* where *a* is the

arrival rate, *L* is the packet size and *R* is the transmission rate.



1. Find the total average response time.

**Answer:** The traffic intensity is aL/R, where a is the arrival rate, L is the packet size and R is the transmission rate. We get L/R = 675000/15000000 = 0.045, and multiplying this by the average number of requests per second (20) gives us 0.9 as the traffic intensity on the link. Based on the table above, the average access delay is going to be 250 msec (0.25 seconds), since the traffic intensity is (on average) 0.9. Adding this to the average internet delay (2 sec) gives us a total average response time of 2 sec + 0.25 sec = 2.25 sec.



1. Now suppose a cache is installed in the institutional LAN as shown in Fig. 2. Suppose the hit rate is 0.333. Find the total average response time.

**Answer:** The traffic intensity on the access link is reduced by 33.3%, since 33.3% of the requests are satisfied at the cache. The average access delay is also 250 msec, or 0.25 seconds (given by the table above, since the traffic intensity is 0.9). Suppose the response time is zero if the request is satisfied by the cache (33.3% of the time). The average response time is 0.25 sec + 2 sec = 2.25 sec (determined in part a) for cache misses (66.6% of the time). This means that the total response time is (0.333 \* 0) + (0.666 \* 2.25) = 1.4985 sec when a cache is installed.