Problem 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 or RTT_1, RTT_2, ..., RTT_n. Further suppose that the Web page associated with the link has a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the HTML file. Assume zero transmission time. Suppose the HTML file references 11 very small objects on the same server. Neglect transmission times, how much time elapses from when the client clicks on the link until the client receives all objects with:

(a) Non-persistent HTTP with no parallel TCP connections?

(b) Non-persistent HTTP with the browser configured for 5 parallel connections?

(c) Persistent HTTP with no parallel TCP connections?

(d) Persistent HTTP with the browser configured for arbitrarily many parallel connections?

Write your solution to Problem 1 in this box

total amount of time to acquire the IP address is

$RTT_1 + RTT_2 + \quad + RTT_n$

it takes RTT_0 to set up TCP connection

(a) non-persistent, no parallel

$2RTT_0 + 11 \times 2RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

$= 24RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

(b) non-persistent, 5× parallel

$2RTT_0 + 3 \times 2RTT_0 + RTT_1 + RTT_2 + \quad + RTT_n$

$= 8RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

(c) persistent, no parallel

$2RTT_0 + 11 \times RTT_0 + RTT_1 + \dots + RTT_n$

$= 13RTT_0 + RTT_1 + \dots + RTT_n$

(d) persistent, N× parallel

$2RTT_0 + \left(\left[\frac{11}{N}\right] + 1\right) RTT_0 + RTT_1 + \dots + RTT_n$

↓

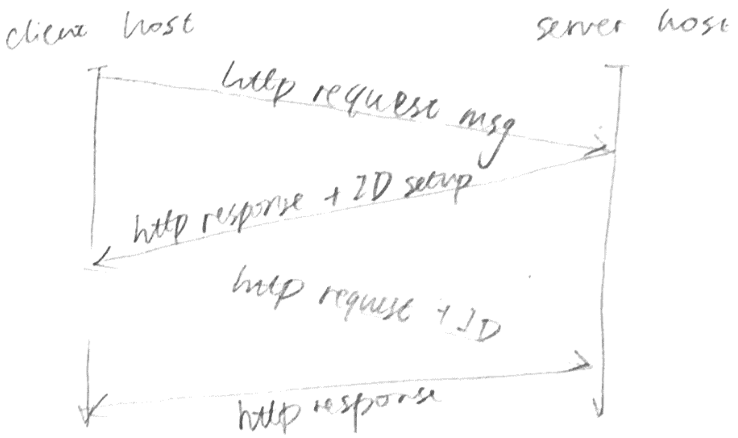
round down, plus 1

Problem 2

How does the web server (e.g., eBay) identify users when you do the Internet shopping? Briefly explain how it works.

Write your solution to Problem 2 in this box

The identification is achieved through cookies. After the first-time login/visit, the server would issue an unique ID for the user/client. The server also stores user's data. Therefore, the next time client sends a http request message, the message is followed by his ID (cookie). In this way, the server would be able to identify the user



Problem 3

A Web browser running on the client host is requesting a webpage from the server. We make the following assumptions:

- TCP window is large once the TCP handshake is complete (i.e. ignore flow control). TCP header size is h bits, and the maximum payload size is p bits.
- The bandwidth is b bps, and the propagation delay is d seconds.
- Ignore DNS related delays, and ignore the payload in three-way handshake packets, ACK packets, and HTTP request packets. In other words, those packets consist of header only.
- The client requests a webpage consisting of an HTML file that indexes 5 binary files on the same server. Each of the file is $2p$ bits long. In other words, each of the file can be sent in exactly 2 TCP packets. Piggybacking is used whenever possible.
- Each HTTP request is sent in one TCP packet.

Please answer the following questions:

(a) Suppose pipelining of HTTP requests is allowed and no parallel TCP connections are used. Calculate the minimal time it takes the browser to receive all the files.

(b) Suppose the non-persistent, non-pipelining mode with parallel TCP connections is used, repeat the calculation.

(c) Which mode gives the smaller latency? Briefly justify your answer.

Problem 3 continued on next page...

Write your solution to Problem 3 in this box

1a) persistent pipelining HTTP:

- handshaking msgs: $(\frac{h}{b}+d)_{send} + (\frac{h}{b}+d)_{resp} = 2(\frac{h}{b}+d)$
- html file request: $(\frac{h}{b}+d) + (\frac{h}{b}+d) = 2(\frac{h}{b}+d)$

5 pipelining file = 10 pipelining packets

k-th request arrive at server: $(n+1)\,h/b + d$ $\therefore$ diff $= \frac{h}{b}$

k-th request issued $n \cdot \frac{h}{b} + d$

response transmittion $2(\frac{h+p}{b}) > \frac{h}{b} \Rightarrow$ all response msgs after the first one wait

For 10th packet: send delay $= 5 \cdot \frac{h}{b} + d$

send response $= 5 \cdot \frac{h}{b} + d - (\frac{h}{b}+d) = 4 \cdot \frac{h}{b}$

remaining transmittion delay $= 10\frac{h+p}{b} - 4\frac{h}{b} = \frac{6h+10p}{b}$

arrive at $\frac{6h+10p}{b} + b$

in total: $2(\frac{h}{b}+d) + 2(\frac{h}{b}+d) + 5\frac{h}{b} + d + \frac{6h+10p}{b} + d$

$= \boxed{\frac{15h+10p}{b} + 6d}$

(b) handshaking: $2(h/b+d)$ html request: $2(h/b+d)$

10 packets: $(\frac{2h}{b/5} + 2d) + (\frac{h}{b/5} + d + \frac{2h+2p}{b/5}) = \frac{25h+10p}{b} + 4d$

$\therefore$ in total: $\boxed{\frac{29h+10p}{b} + 8d}$

1c) **persistent + pipelining gives the smaller latency**

because $\frac{29h+10p}{b} + 8d > \frac{15h+10p}{b} + 6d$

Problem 4

What is the difference between MAIL FROM: in SMTP and From: in the mail message itself?

Write your solution to Problem 4 in this box

MAIL FROM: SMTP is part of the SMTP handshaking process. It's a verification that enables the server to identify the sender of the mail message.

FROM: is not SMTP message. It's part of the email body and it's for human to read

Problem 5

Suppose your department has a local DNS server for all computers in the department.

(a) Suppose you are an ordinary user (i.e., not a network/system administrator). Can you determine if an external Web site was likely accessed from a computer in your department a couple of seconds ago? Explain.

(b) Now suppose you are a system administrator and can access the caches in the local DNS servers of your department. Can you propose a way to roughly determine the Web servers (outside your department) that are most popular among the users in your department? Explain.

Write your solution to Problem 5 in this box

(a) Yes, I can, by using the "dig" command on Unix/Linux. it shows the hierarchy of the DNS server. Since the website was just accessed, an entry indicating this website would be cached into the local DNS server. Thus the query time should be 0 sec. Otherwise, the query time is larger.

(b) Yes, I can, by measuring the frequency of Web servers appearing in the DNS cache. The more popular a web server is, the more user would be visiting this website (requests). Thus it appears in the local DNS cache more often