CS118 Homework 2

Wenxuan Liu

Problem 1

The web server uses cookie to identify users and maintain the state. Assume that the TCP connection port 80 has already been set. First, when user access internet (such as login) on the client, the client sends a HTTP request massage to the server under the TCP protocol on port 80. Then, the server creates a cookie for the user(client) with a unique ID corresponding to its user account, web browser, device, etc., and save the cookie into the server's backend database. Then, the server sends a http response with the cookie ID back to the client, and the client set the cookie ID in its cookie file. Hence the user state is saved both in the client and server.

When accessing the server the next time, the client sends the http request with its cookie header line(saved in client's cookie file) to the server, then the server verify the received cookie with its backend database, and therefore server can identify the user and send the usual http response massage back to the client.

Problem 2

(a) Non-persistent with no parallel connection is consists of RTTs visiting each of the n DNS server and 1+9 times of requests, each takes 2 RTT0:

$$RTT_1 + RTT_2 + \cdots + RTT_n + 2RTT_0 + 9 * 2RTT_0$$
$$= 20RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$$

(b) Non-persistent with 5 parallel connections:
With 5 parallel connection, 9 files need 9/5≈2 times of requests, each takes 2 RTT0.
The connection for the first time also needs 2 RTT0.

$$RTT_1 + RTT_2 + \cdots + RTT_n + 2RTT_0 + 2 * 2RTT_0$$
$$= 6RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$$

(c) Persistent with no parallel connection is consists of RTTs visiting each of the n DNS server, 2RTT0 for getting connection and HTML, and 9 times of requests each taking 1 RTT0:

$$RTT_1 + RTT_2 + \cdots + RTT_n + 2RTT_0 + 9 * 1RTT_0$$
$$= 11RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$$

(d) Persistent with arbitrarily many parallel connections is consists of RTTs visiting each of the n DNS server, 2RTT0 for getting connection and HTML, and 1 requests for all objects taking 1 RTT0:

$$RTT_1 + RTT_2 + \cdots + RTT_n + 2RTT_0 + 1RTT_0$$
$$= 3RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$$

Problem 3

SMTP uses a line with only a period ('CRLF.CRLF') to mark the end of a message body.
HTTP uses "Content-Length header field" to mark the length of a message body, and then use it to find the end of a message.
No, HTTP cannot use the same method as SMTP to mark the end of the message body, because HTTP is in binary data format, and SMTP is in 7-bit ASCII format. Also, there is no special format for message body in HTTP, so special mark(like a line with only a period) cannot be effectively detected in HTTP message body, and using such method could cause misinterpretation when such mark is actually used for message itself instead of marking the end of message body.

Problem 4

(a) Yes. We can access the local DNS server's cache to check whether or not the external web site has been accessed by a computer in the department a couple of seconds ago. It is because if a computer in the department access the website as a user, it will leave the record of the website's domain name in the local DNS server's cache that is available to all users. Then, we can use command like 'dig' to check the query time for this website. If the query time is about 0 msec, it means the domain name has just been accessed and is in the local DNS server's cache. Otherwise the query time will be larger since we have to contact the root DNS server for the query.

(b) Yes. We can record the local DNS server's caches periodically and choose the external web servers that appear most frequently to be the most popular Web servers. It is because if a web server is popular among the users in the department, it would be frequently requested by users and all consequently stored in the local DNS server's caches for a lot of times. Periodically recording the website domains in the local DNS caches shared by all users can give us the most frequently requested web servers, and hence find the most popular ones.

(c)

Problem 5

Client-Server Distribution

u\N	10	100
300Kbps	7500s	50000s
2Mbps	7500s	50000s

P2P Distribution

u\N	10	100
300Kbps	7500s	25000s
2Mbps	7500s	7500s

Explanation:
The statistics we have is:
F = 15Gbits = 15000 Mbits
u_s = 30 Mbps
d_{min} = 2 Mbps
u_i = 300Kbps = 0.3Mbps, or u_i = 2Mbps
For Client-Server distribution, the minimum distribution time is:

$$D_{cs} = \max\left\{\frac{NF}{u_s}, \frac{F}{d_{min}}\right\}$$

For P2P distribution, the mimimum distribution time is:

$$D_{p2p} = \max\left\{\frac{F}{u_s}, \frac{F}{d_{min}}, \frac{NF}{u_s + \sum_{i=1}^{N} u_i}\right\}$$

So, for N=10 and u_i=300Kbps,

$$D_{cs} = \max\left\{\frac{10 * 15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}\right\} = \max\{5000, 7500\}\,s = 7500s$$

$$D_{p2p} = \max\left\{\frac{15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}, \frac{10 * 15000Mbits}{30Mbps + 10 * 0.3Mbps}\right\}$$

$$= \max\{500, 7500, 4545\}\,s = 7500s$$

For N=10 and u_i=2Mbps,

$$D_{cs} = \max\left\{\frac{10 * 15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}\right\} = \max\{5000, 7500\}\,s = 7500s$$

$$D_{p2p} = \max\left\{\frac{15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}, \frac{10 * 15000Mbits}{30Mbps + 10 * 2Mbps}\right\}$$

$$= \max\{500, 7500, 3000\}\,s = 7500s$$

For N=100 and u_i=300Kbps,

$$D_{cs} = \max\left\{\frac{100 * 15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}\right\} = \max\{50000, 7500\}\,s = 50000s$$

$$D_{p2p} = \max\left\{\frac{15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}, \frac{100 * 15000Mbits}{30Mbps + 100 * 0.3Mbps}\right\}$$

$$= \max\{500, 7500, 25000\}\,s = 25000s$$

For N=100 and u_i=2Mbps,

$$D_{cs} = \max\left\{\frac{100 * 15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}\right\} = \max\{50000, 7500\}\, s = 50000s$$

$$D_{p2p} = \max\left\{\frac{15000Mbits}{30Mbps}, \frac{15000Mbits}{2Mbps}, \frac{100 * 15000Mbits}{30Mbps + 100 * 2Mbps}\right\}$$

$$= \max\{500, 7500, 6522\}\, s = 7500s$$