P113

Problem 1

If you visit a certain webpage and your browser has a cached version of that page saved on your previous visit.

(a) How does the browser know if its cached page is the current version on the server?

(b) What will the web server do if the browser's cached page is the current version? What about the opposite case?

Answer:

a)

HTTP has a mechanism that allows a cache to verify that its objects are up to date. This mechanism is called the **conditional GET . (P113 of textbook)**

When visit the page, to check if the object has been modified, the cache performs an up-to-date check by issuing a conditional GET. The request message includes an If-Modified-Since: header line.

When the web server send the requested object to the cache, there would be a value called Last-modified. The cache forwards the object to the requesting browser but also caches the object locally. The cache would also store the last-modified date along with the object, which equal to the value of if-modified-since.

In short, according to condition GET, which is telling the server to send the object if the object has been modified since the time that stored in if-modified-since. Then the server would respond with status code to tell browser that if its cached page is the current version of the server.

B) If browser's cached page is the current version. Web server sends reponses message but does not include the request object in the response message. It returns status code 304 - Not Modified in the status line, which tells the cache that it can go ahead and forward it.

In opposite case, the web server returns status code 200 - OK, and send the request object to the cache, and update the value of Last-Modified as well.

Problem 2

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 (round-trip time) of 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 nine very small objects on the same server. 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 4 parallel connections?

(c) Persistent HTTP with no parallel TCP connections?

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

Answer:

Total time to get IP address : $RTT_1 + \cdots + RTT_n = \sum_{i=1}^{n} RTT_i$

Non-persistent HTTP requires $2RTT_0$ per object:
To establish connection, the time is 1 RTT_0,
and 1 RTT_0 to fetch the object. So each object need 2 RTT_0

a) $(2 \times \underbrace{9}_{\text{9 objects}} RTT_0 + \sum_{i=1}^{n} RTT_i) + \underbrace{2RTT_0}_{\text{fetch root html file}}$

$= 20RTT_0 + \sum_{i=1}^{n} RTT_i$

b) $2RTT_0 + 2RTT_0 + 2RTT_0 + \sum_{i=1}^{n} RTT_i$

(get HTML file) (1~5 objects) (6~9 objects)

$= 6RTT_0 + \sum_{i=1}^{n} RTT_i$

c) 1 RTT_0 for each object in persistent HTTP.

$9RTT_0 + 2RTT_0 + \sum_{i=1}^{n} RTT_i = 11RTT_0 + \sum_{i=1}^{n} RTT_i$

d) $RTT_0 + 2RTT_0 + \sum_{i=1}^{n} RTT_i$

$= 3RTT_0 + \sum_{i=1}^{n} RTT_i$

Problem 3

How does SMTP marks the end of a message body? How about HTTP? Can HTTP use the same method as SMTP to mark the end of the message body?

Answer:

SMTP: it marks the end of a message body with the client sending a single period on a new line. SMTP message body should be in a 7-bit ASCII text.
HTTP: use “content-length header field” to indicate the length of a message body.

No. HTTP cannot use the same method as SMTP to make the end of the message body.
As said before, SMTP message body should be in a 7-bit ASCII text.
However, HTTP message has no restrictions, it could be binary data.

Problem 4

(a) Why do HTTP, SMTP, and SSH run on top of TCP rather than on UDP?

(b) Suppose you wanted to do a transaction from a remote client to a server as fast as possible. Would you use UDP or TCP? Why?

Answer:

a)

Because TCP is more reliable than UDP.

TCP is connected-oriented protocol, but UDP is not. TCP provide reliable message delivery. It ensures that data is not damaged, lost, duplicated or delivered out of order to a receiving process.

HTTP, SMTP, and SSH cannot be affordable using UDP, UDP cannot guarantee that transmit package in the correct order. It could not provide reliable delivery.

b)

Use UDP, because of the requirement “as fast as possible”.

UDP is faster than using TCP.

Problem 5

Consider the recursive DNS query procedure.

(a) What are the root DNS servers used for?

(b) If a local DNS server needs to query root DNS servers, how does it know their IP addresses?

(c) If you go to `https://www.iana.org/domains/root/servers`, you will find that there are only 13 root server addresses. Will this be a problem? Why or why not?

Answer:

DNS - Domain Name System.

a) Root DNS servers are the first step in the name resolution of any domain name, meaning they translate domain names into IP address.

Room name servers provide the IP address of the TLD servers.

b)

In the procedure of recursive DNS query, if the local DNS servers did not have the information of the requested IP address, the local DNS servers will send a query to the root server. Since the root DNS IP address is generally fixed, local DNS servers are usually built with the IP address of root DNS server.

Or more precisely,"Operators who manage a DNS recursive resolver typically need to configure a "root hints file". This file contains the names and IP addresses of the root servers, so the software can bootstrap the DNS resolution process. For many pieces of software, this list comes built into the software." (From https://www.iana.org/domains/root/servers)

Root servers will return the list of TLD servers.

c) Not a problem.

First, there are more than 1000 root servers instances that are copies of 13 different root servers. It means that there is a server cluster for each of the 13 root server address. They use anycast routing, which helps balance their decentralization and ensure reliability. The would be not a huge influence if one of them is unavailable. In a word, it is 13 clusters of servers, not literally 13 root servers.

Furthermore, the reason for choosing 13 root name servers was to fit all the IP addresses in a single 512 byte packet. UDP is used for queries, however the maximum packet that can be guaranteed to work in any UDP implementation is 512 bytes. For all the root server data to be contained in a 512-byte UDP packet,

The number of root servers is limited to 13, and each server is named using a single letter of the alphabet.