資工三 **109590004** 呂育瑋　　計算機網路 **Homework 2**

1. 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?

   A: 使用UDP，因為UDP協定傳輸只會需要1個RTT，而TCP協定因為需要先建立server與連線，至少會需要2RTT，因此如果只要求傳輸速度快，那麼UDP會是最佳選擇。

2. Describe how Web caching can reduce the delay in receiving a requested object.

   A: 如果請求物件在暫存中，能直接由proxy server給予請求的物件，如果暫存中不存在該請求物件，則由proxy server向original server請求物件，再由proxy server將物件傳給client，並將該物件記錄在暫存中，下次client有相同請求時，能更快收到回覆。

3. Why do HTTP, FTP, SMTP, and POP3 run on top of TCP rather than on UDP?

   A: 這些服務需要比UDP更可靠的協定來進行資料傳輸，UDP無法驗證封包是否被正確傳輸，對於這些服務，他們希望封包被完整的送達，即使那會需要較多的時間。

4. Suppose within your Web browser you click on a link to obtain a Web page. The IP address for the associated URL is not in your local host, so a DNS lookup is necessary to obtain the IP address. Suppose that *n* DNS server are visited before your host receives the IP address from DNS; the successive visits incur an RTT of RTT_1 ,..., RTT_n. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT_0 denote the RTT between the local host and the server containing the object. Assuming zero transmission time of the object, how much time elapses from when the client clicks on the link until the client receives the object?

   A: 尋找IP地址消耗的時間: $RTT_1 + RTT_2 + \dots + RTT_n$
   Client與Server建立連結時間: RTT_0
   Clinet請求與接收封包的時間: RTT_0
   總時長: $2 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

5. Assume that you click on a Web browser to obtain a web page from the web server. Assume the web page contains 3 very small additional objects on the same server. Let RTT_0 denote the Round Trip Time (RTT) between the local host and the server containing those objects. Neglecting transmission times, how much time elapses when the client clicks on the link until the client receives those objects with
   (a) Non-persistent HTTP with parallel connections?
   (b) Non-persistent HTTP with no parallel TCP connections?
   (c) Persistent HTTP with parallel connections?

   A:

   Non-persistent HTTP with parallel connections:
   $2 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n + 2 * RTT_0 = 4 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

   Non-persistent HTTP with no parallel TCP connections
   $2 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n + 3 * (2 * RTT_0) = 8 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$

   Persistent HTTP with parallel connections
   $2 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n + RTT_0 = 3 * RTT_0 + RTT_1 + RTT_2 + \dots + RTT_n$