Jonathan Woong
804205763
CS 118 – DIS 1A

Homework 3

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 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?

$$time\ to\ get\ IP = RTT_1 + RTT_2 + RTT_3 + \cdots + RTT_n$$

$$TCP\ setup + get\ object = RTT_0 + RTT_0$$

$$total\ time = time\ to\ get\ IP + TCP\ setup + get\ object = 2RTT_0 + RTT_1 + RTT_2 + \cdots + RTT_n$$

Suppose the HTML file references eight very small objects on the same server. Neglecting transmission times, how much time elapses with:

a. Non-persistent HTTP with no parallel TCP connections?

$$RTT_1 + RTT_2 + RTT_3 + \cdots + RTT_n + 2RTT_0 + 8 * 2(RTT_0) = 18RTT_0 + RTT_1 + \cdots + RTT_n$$

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

$$RTT_1 + RTT_2 + RTT_3 + \cdots + RTT_n + 2RTT_0 + 2 * 2(RTT_0) = 6RTT_0 + RTT_1 + \cdots + RTT_n$$

c. Persistent HTTP?

$$RTT_1 + RTT_2 + RTT_3 + \cdots + RTT_n + 2RTT_0 + RTT_0 = 3RTT_0 + RTT_1 + \cdots + RTT_n$$

2. Does there exist the notion of client and server sides of a communication session for a peer-to-peer (P2P) application? Why or why not? List any four applications that use P2P architecture.

Yes. Any form of communication between two entities has a client side and server side. For P2P, the peer that receives the file can be thought of as a client and the peer that sends the file can be thought of as the server.

1. File distribution
2. Video conferencing
3. Content streaming
4. Collective computing

3. Consider distributing a file of F = 15 Gbits to N peers. The server has an upload rate of us = 30 Mbps, and each peer has a download rate of di = 2 Mbps and an upload rate of u. For N = {10, 1000} and u = {300, 2000} Kbps, prepare a chart giving the minimum distribution time for each of the combinations of N and u for both client-server distribution and P2P distribution.

$$D_{Client-Server} = max\left\{\frac{F}{u}, \frac{F}{d_{min}}\right\}$$

$$D_{P2P} = max\left\{\frac{F}{u}, \frac{F}{d_{min}}, \frac{NF}{u + \sum_{i=1}^{N} u_i}\right\}$$

$$F = 15\ Gbits = 15 * 1024\ Mbits$$

Jonathan Woong
804205763
CS 118 – DIS 1A

Homework 3

$$u = 30\ Mbps$$
$$d_{min} = d_i = 2\ Mbps$$

Client-Server		N	
		10	**1000**
u	**300 Kbps**	7680	512000
	2000 Kbps	7680	512000

P2P		N	
		10	**1000**
u	**300 Kbps**	7680	47559
	2000 Kbps	7680	7680

4. List at least four types services that a transport protocol can provide. For each of the service classes, indicate if UDP or TCP (or neither or both) provides such a service.

1. Reliable data transfer (TCP)
2. Sustained throughput (neither)
3. Guaranteed delivery time (neither)
4. Security (neither)

5. UDP and TCP use 1s complement for their checksums. Suppose you have the following three 8-bit bytes: 01010011, 01100110, 01110100. What is the 1s complement of the sum of these 8-bit bytes? (Note that although UDP and TCP use 16-bit words in computing the checksum, for this problem you are being asked to consider 8-bit sums.)

Why is it that UDP takes the 1s complement of the sum; that is, why not just use the sum? With the 1s complement scheme, how does the receiver detect errors? Is it possible that a 1-bit error will go undetected? How about a 2-bit error? Page 1 of 1

$$\begin{array}{r} 0\,1\,0\,1\,0\,0\,1\,1 \\ \underline{+\,0\,1\,1\,0\,0\,1\,1\,0} \\ 1\,0\,1\,1\,1\,0\,0\,1 \end{array}$$

$$\begin{array}{r} 1\,0\,1\,1\,1\,0\,0\,1 \\ \underline{+\,0\,1\,1\,1\,0\,1\,0\,0} \\ 0\,0\,1\,0\,1\,1\,1\,0 \end{array}$$

$$one's\ complement = 1101001$$

UDP takes the 1s complement of the sum in order to utilize error detection. It does this by adding the three original 8-bit bytes with the checksum and checking whether the final sum has a 0 bit. If it does have a 0 bit, then a 1 bit error has been detected.