ASSIGNMENT NO 4 SOLUTION

Problem 7

The total amount of time to get the IP address is
$RTT_1 + RTT_2 + \cdot + RTT_n$.
Once the IP address is known, RTT_O elapses to set up the TCP connection and another RTT_O elapses to request and receive the small object. The total response time is
$2RTT_o + RTT_1 + RTT_2 + \cdot + RTT_n$

Problem 9

a) The time to transmit an object of size *L* over a link or rate *R* is *L/R*. The average time is the average size of the object divided by *R*:

Δ = (850,000 bits)/(15,000,000 bits/sec) = .0567 sec

The traffic intensity on the link is given by $\beta\Delta$=(16 requests/sec)(.0567 sec/request) = 0.907. Thus, the average access delay is (.0567 sec)/(1 - .907) ≈ .6 seconds. The total average response time is therefore .6 sec + 3 sec = 3.6 sec.

b) The traffic intensity on the access link is reduced by 60% since the 60% of the requests are satisfied within the institutional network. Thus the average access delay is (.0567 sec)/[1 – (.4)(.907)] = .089 seconds. The response time is approximately zero if the request is satisfied by the cache (which happens with probability .6); the average response time is .089 sec + 3 sec = 3.089 sec for cache misses (which happens 40% of the time). So the average response time is (.6)(0 sec) + (.4)(3.089 sec) = 1.24 seconds. Thus the average response time is reduced from 3.6 sec to 1.24 sec.

Problem 22

For calculating the minimum distribution time for client-server distribution, we use the following formula:

$D_{cs} = max\,\{NF/u_s,\ F/d_{min}\}$

Similarly, for calculating the minimum distribution time for P2P distribution, we use the following formula:

$$D_{P2P} = max\{F/u_s,\ F/d_{min},\ NF/(u_s + \sum_{i=1}^{N} u_i)\}$$

Where, F = 15 Gbits = 15 * 1024 Mbits
u_s = 30 Mbps
$d_{min} = d_i$ = 2 Mbps

Note, 300Kbps = 300/1024 Mbps.

Client Server

		N		
		10	**100**	**1000**
	300 Kbps	7680	51200	512000
u	**700 Kbps**	7680	51200	512000

2 Mbps	7680	51200	512000

Peer to Peer

		N		
		10	**100**	**1000**
	300 Kbps	7680	25904	47559
u	**700 Kbps**	7680	15616	21525
	2 Mbps	7680	7680	7680

Problem 26

Yes. His first claim is possible, as long as there are enough peers staying in the swarm for a long enough time. Bob can always receive data through optimistic unchoking by other peers.

His second claim is also true. He can run a client on each host, let each client "free-ride," and combine the collected chunks from the different hosts into a single file. He can even write a small scheduling program to make the different hosts ask for different chunks of the file. This is actually a kind of Sybil attack in P2P networks.

Problem 28

a) If you run TCPClient first, then the client will attempt to make a TCP connection with a non-existent server process. A TCP connection will not be made.

b) UDPClient doesn't establish a TCP connection with the server. Thus, everything should work fine if you first run UDPClient, then run UDPServer, and then type some input into the keyboard.

c) If you use different port numbers, then the client will attempt to establish a TCP connection with the wrong process or a non-existent process. Errors will occur.