**컴퓨터네트워킹 과제**

**HW#2**

**강좌 명: 컴퓨터네트워킹**

**교수님: 안종석 교수님**

**학과: 컴퓨터공학과**

**학번: 2017112138**

**이름: 정여준**

**Q4. Consider the following string of ASCII characters that were captured by Wireshark when the browser sent an HTTP GET message (i.e., this is the actual content of an HTTP GET message). The characters are carriage return and line-feed characters (that is, the italized character string in the text below represents the single carriage-return character that was contained at that point in the HTTP header). Answer the following questions, indicating where in the HTTP GET message below you find the answer.**

GET /cs453/index.html HTTP/1.1Host: gaia.cs.umass.edu User-Agent: Mozilla/5.0 ( Windows;U; Windows NT 5.1; en-US; rv:1.7.2) Gec ko/20040804 Netscape/7.2 (ax) Accept:ex t/xml, application/xml, application/xhtml+xml, text /html;q=0.9, text/plain;q=0.8,image/png,\*/\*;q=0.5 Accept-Language: en-us,en;q=0.5AcceptEncoding: zip,deflateAccept-Charset: ISO -8859-1,utf-8;q=0.7,\*;q=0.7Keep-Alive: 300 Connection:keep-alive

**a. What is the URL of the document requested by the browser?**

**b. What version of HTTP is the browser running?**

**c. Does the browser request a non-persistent or a persistent connection?**

**d. What is the IP address of the host on which the browser is running?**

**e. What type of browser initiates this message? Why is the browser type needed in an HTTP request message?**

a) Host: gaia.cs.umass.edu는 서버 이름을 나타내고 /cs453/index.html은 파일 이름을 나타내므로 browser에 의해 요청된 document는 <http://gaia.cs.umass.edu/cs453/index.html>이다.

b) browser에서 적용되고 있는 HTTP의 version은 1,1이다.

c) browser가 Connection: keep-alive를 통해서 영구적인 연결 상태인 것을 알 수 있다.

d) 호스트의 IP 주소는 HTTP 메시지에 포함되지 않는다. 따라서 HTTP 메시지 교환만으로는 host의 IP 주소를 알 수 없다. 정보를 알기 위해서는 IP datagram이 필요하다.

e) request message는 User-Agent가 만들기 때문에 이 message를 만든 browser의 type Mozilla/5.0이다. browser 유형 정보는 서버가 동일한 개체의 다른 버전을 다른 유형의 browser로 전송하기 위해 필요합니다.

**Q7. 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 RTT1, . . . , RTTn. Further suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Let RTT0 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?**

IP 주소의 총 시간: RTT1 + RTT2 + … + RTTn이다.

IP 주소가 알려지면 TCP 연결을 설정하는데 RTT0 흘러가고 다른 RTT0이 작은 개체를 요청하고 수신하는 데 흘러간다.

따라서 클라이언트가 링크를 클릭할 때부터 클라이언트가 개체를 수신할 때까지의 걸린 시간은

2RTT0 + RTT1 + RTT2 + … + RTTn이다.

**Q10. Consider a short, 10-meter link, over which a sender can transmit at a rate of 150 bits/sec in both directions. Suppose that packets containing data are 100,000 bits long, and packets containing only control (e.g., ACK or handshaking) are 200 bits long. Assume that N parallel connections each get 1/N of the link bandwidth.**

**Now consider the HTTP protocol, and suppose that each downloaded object is 100 Kbits long, and that the initial downloaded object contains 10 referenced objects from the same sender. Would parallel downloads via parallel instances of non-persistent HTTP make sense in this case? Now consider persistent HTTP. Do you expect significant gains over the non-persistent case? Justify and explain your answer.**

다운로드한 각 객체를 하나의 데이터 패킷에 완전히 넣을 수 있다. Tp는 클라이언트와 서버 간의 단방향 전파 지연을 나타낸다.

먼저 non-persistent 연결을 통한 병렬 다운로드를 생각해본다. 병렬 다운로드는 10개의 연결이 150 bits/sec의 대역폭을 공유하도록 허용하므로 각 연결은 단지 15 bits/sec의 대역폭을 얻을 수 있다.

Transmission rate(R) = 150 bits/sec

Packet Length(L) = 100,000 bits

Control data = 200 bits, Object data = 100Kbits

Distance(d) = 10meter, N = 10, Bandwidth = 150/10 = 15 bits/sec

Non-persistent HTTP를 통해 모든 객체를 수신하는 데 필요한 총 시간은 다음과 같다.

T = (200/150 + Tp + 200/150 + Tp + 200/150 + Tp + 100000/150 + Tp) + (200/15 + Tp + 200/15 + Tp + 200/15 + Tp + 100000/15 + Tp)

T = 7377 + 8Tp

Persistent HTTP 연결에 필요한 총 시간은 다음과 같다.

T = (200/150 + Tp + 200/150 + Tp + 200/150 + Tp + 100000/150 + Tp ) + 10\*{(200+100000)/15 + 2Tp}

T = 7350 + 24Tp

Tp는 빛의 속도로 0.03 \* 10^-6이고 이는 무시할 수 있다.

결론적으로 Persistent HTTP를 통한 병렬 다운로드가 Non-persistent HTTP과 1% 미만의 차이밖에 없기 때문에 큰 이득이 되지 않는 다는 것을 추론할 수 있다.

**Q19. In this problem, we use the useful dig tool available on Unix and Linux hosts to explore the hierarchy of DNS servers. Recall that in Figure 2.19, a DNS server in the DNS hierarchy delegates a DNS query to a DNS server lower in the hierarchy, by sending back to the DNS client the name of that lower-level DNS server. First read the man page for dig, and then answer the following questions.**

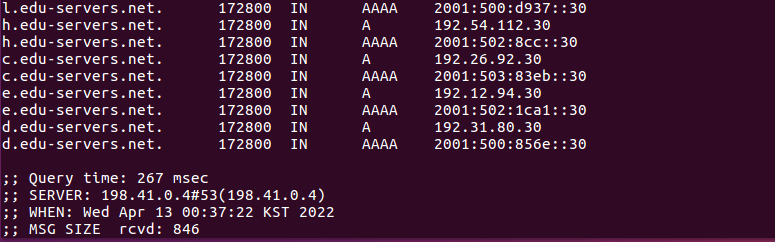
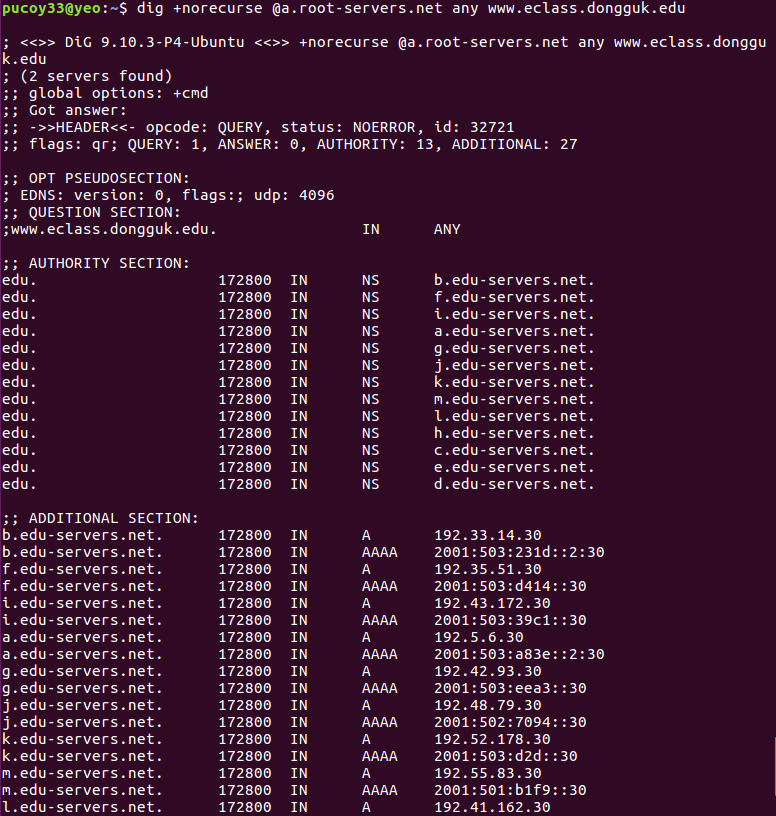
**a. Starting with a root DNS server (from one of the root servers [a-m]. root-servers.net), initiate a sequence of queries for the IP address for your department’s Web server by using dig. Show the list of the names of DNS servers in the delegation chain in answering your query.**

**b. Repeat part (a) for several popular Web sites, such as google.com, yahoo .com, or amazon.com.**

a) 연쇄적으로 호출할 주소를 eclass.dongguk.edu로 한다.

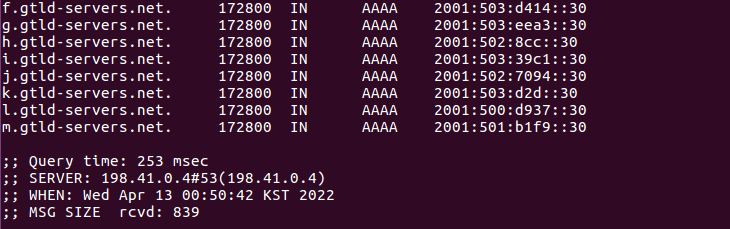
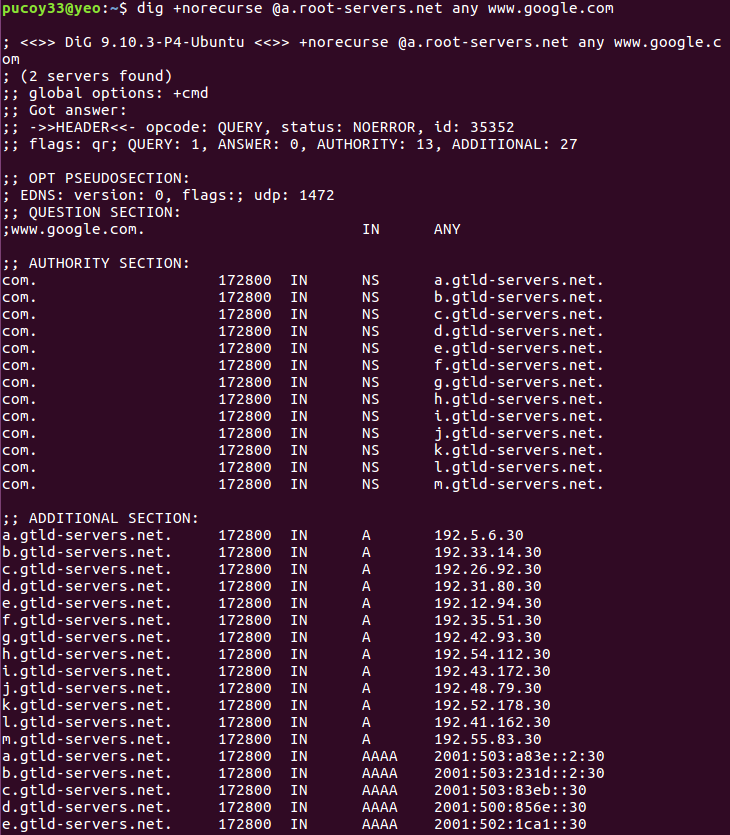
명령어: dig +norecurse @a.root-servers.net any [www.eclass.dongguk.edu](http://www.eclass.dongguk.edu)

실행 결과



b) 명령어: dig +norecurse @a.root-servers.net any [www.google.com](http://www.google.com)

실행 결과



**Q23. Consider distributing a file of F bits to N peers using a client-server architecture. Assume a fluid model where the server can simultaneously transmit to multiple peers, transmitting to each peer at different rates, as long as the combined rate does not exceed us.**

**a. Suppose that us/N <= dmin. Specify a distribution scheme that has a distribution time of NF/us.**

**b. Suppose that us/N >= dmin. Specify a distribution scheme that has a distribution time of F/dmin.**

**c. Conclude that the minimum distribution time is in general given by max {NF/us, F/dmin}.**

a) 클라이언트의 다운로드 속도가 높을수록 서버는 us/N의 속도로 최대 전송이 가능하다. 그래서 클라이언트는 us/N의 비율로 파일을 받을 수 있다.

클라이언트가 전체 파일을 받기 위해 필요한 시간은 F/(us/N) = NF/us이다.

따라서, 서버가 모든 클라이언트에 파일을 전송하는 데 필요한 시간은 NF/us이다.

b) 클라이언트가 파일을 받기 위해서 필요한 시간은 dmin이다. 모든 파일을 받기 위해 필요한 시간은 F/dmin이다.

c) client-server에서 파일을 distribution하는 실질적인 시간은 Dcs >= max {NF/us, F/dmin}이다.

Dcs >= NF/us라는 것을 알 수 있고 (a)에서 우리는 Dcs <= NF/us라는 것을 알 수 있다.

위의 두 식을 조합하면 Dcs = NF/us(us/N <= dmin일 때)라는 것을 알 수 있다.