Homework 2 – 50 points

ATM Application-Level Protocol (10 points)
Design and describe an application-level protocol to be used between an automatic teller machine and a bank’s centralized computer. Your protocol should allow a user’s card and password to be verified, the account balance (which is maintained at the centralized computer) to be queried, and an account withdrawal to be made (that is, money disbursed to the user). Your protocol entities should be able to handle the all-too-common case in which there is not enough money in the account to cover the withdrawal. Specify your protocol by listing the messages exchanged and the action taken by the automatic teller machine or the bank’s centralized computer on transmission and receipt of messages. Sketch the operation of your protocol for the case of a simple withdrawal with no errors, using a diagram similar to that in Figure 1.2. Explicitly state the assumptions made by your protocol about the underlying end-to-end transport service.

The HTTP GET message (10 Points)
Consider the figure below, where a client is sending an HTTP GET message to a web server, gaia.cs.umass.edu.

Suppose the client-to-server HTTP GET message is the following:

GET /kurose_ross/interactive/quotation1.htm HTTP/1.1
Host: gaia.cs.umass.edu
Accept: text/plain, text/html, image/gif, image/jpeg, audio/basic, audio/vnf.wave, video/mp4, video/wmv, application/*, */*
Accept-Language: en-us, en-gb;q=0.5, en;q=0.1, fr, fr-ch, zh, cs
If-Modified-Since: Wed, 10 Jan 2018 13:13:03 -0800
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/535.11 (KHTML, like Gecko) Chrome/17.0.963.56 Safari/535.11

Answer the following questions:

1. What is the name of the file that is being retrieved in this GET message?
2. What version of HTTP is the client running?
3. What formats of text, images, audio, and video does the client browser prefer to receive? [Note: for this and the following questions on browser media and language preferences, you will need to do a bit of additional reading on the Web. Here is a good place to start.]
4. What do the strings "application/*" and "*/*" signify in the Accept: header?
5. What languages is the browser indicating that it is willing to accept? [Note: you can look at your own browser preferences to get a listing of language codes.]

6. What is the meaning of the "relative quality factor," q, associated with the various version of English? [Note: Here is a good place to start. See also [RFC 2616].]

7. What is the client’s preferred version of English? What is the browser's least preferred version of English?

8. Does the browser sending the HTTP message prefer Swiss French over traditional French? Explain.

9. Does the client already have a (possibly out-of-date) copy of the requested file? Explain. If so, approximately how long ago did the client receive the file, assuming the GET request has just been issued?

10. What is the type of client browser and the client’s operating system? [Note: To answer this, you’ll need to understand the User Agent: header field. Here is a good place to start.]

The HTTP RESPONSE message (10 Points)
Consider the figure below, where the server is sending a HTTP RESPONSE message back the client.

Suppose the server-to-client HTTP RESPONSE message is the following:

HTTP/1.1 200 OK
Date: Wed, 10 Jan 2018 21:23:35 +0000
Server: Apache/2.2.3 (CentOS)
Last-Modified: Wed, 10 Jan 2018 21:35:35 +0000
ETag:17dc6-a5c-bf716880.
Content-Length: 77385
Keep-Alive: timeout=41, max=92
Connection: Keep-alive
Content-type: image/html

Answer the following questions:

1. Is the response message using HTTP 1.0 or HTTP 1.1? Explain.
2. Was the server able to send the document successfully? Explain
3. At what date and time was this response sent?
4. When was the file last modified on the server?
5. How many bytes are there in the document being returned by the server?

6. What is the default mode of connection for HTTP protocol? Is the connection in the reply persistent or non-persistent? Explain.

7. What is the type of file being sent by the server in response?

8. Does the response message use separate keep-alive messages?

9. What is the name of the server and its version? List the advantages of the server used.

10. What is the timeout value for the response message?

**DNS and HTTP delays (10 points)**

Before doing this question, you might want to review sections 2.2.1 and 2.2.2 on HTTP (in particular the text surrounding Figure 2.7) and the operation of the DNS (in particular the text surrounding Figure 2.19).

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 four DNS servers are visited before your host receives the IP address from DNS. The first DNS server visited is the local DNS cache, with an RTT delay of RTT₀ = 1 msecs. The second, third and fourth DNS servers contacted have RTTs of 31, 47, and 25 msecs, respectively. Initially, let's suppose that the Web page associated with the link contains exactly one object, consisting of a small amount of HTML text. Suppose the RTT between the local host and the Web server containing the object is RTT_HTTP = 14 msecs.

1. Assuming zero transmission time for the HTML object, how much time elapses from when the client clicks on the link until the client receives the object?

2. Now suppose the HTML object references 7 very small objects on the same web server. Neglecting transmission times, how much time elapses from when the client clicks on the link until the base object and all 7 additional objects are received from web server at the client, assuming non-persistent HTTP and no parallel TCP connections?

3. Repeat 2. above but assume that the client is configured to support a maximum of 5 parallel TCP connections, with non-persistent HTTP.
4. Repeat 2. above but assume that the client is configured to support a maximum of 5 parallel TCP connections, with persistent HTTP.

5. What do you notice about the overall delays (taking into account both DNS and HTTP delays) that you computed in cases 2., 3. and 4. above?

Why is it that voice and video traffic is often sent over TCP rather than UDP in today’s Internet? (Hint: the answer we are looking for has nothing to do with TCP’s congestion-control mechanism.)

**Question: Reliable UDP data transfer (5 points)**

Is it possible for an application to enjoy reliable data transfer even when the applications run over UDP? If so, how?

**Question: NAK-only vs ACK-only (5 points)**

Consider a reliable data transfer protocol that uses only negative acknowledgments. Suppose the sender sends data only infrequently. Would a NAK-only protocol be preferable to a protocol that uses ACKs? Why? Now suppose the sender has a lot of data to send and the end-to-end connection experiences few losses. In this case, would a NAK-only protocol be preferable to a protocol that uses ACKs? Why?