CS-335a: Computer Networking

Department of Computer Science, University of Crete
Fall 2024

Deadline: 08/11/2024 **12pm (afternoon, before the lab/lecture)**

Professor: Maria Papadopouli
TA: Christina Brozi (brozi@csd.uoc.gr)

SUBMISSION GUIDELINES

Your report should be in **pdf** format Please submit your assignment via email to the TA. The subject of the email should be "**335a_assign2_csdXXXX**". You can submit your assignment as many times as needed. Only the last submission will be graded. The maximum grade you can get is 115 with 15 out of the 115 points being BONUS.

Assignment 2 - Application Layer

At the Top!

Exercise 1 - HTTP Theoretical (40 points)

i) Why do HTTP, FTP, SMTP and POP3 run on top of TCP rather than UDP? (2p)

ii) How does persistent HTTP improve web performance ? What are the potential drawbacks of persistent HTTP connections? (3p)

iii) Explain the concept of statelessness in HTTP. How does it affect the server's workload? (5p)

iv) How can web caches be used to improve user experience in remote and poorly connected areas? List one drawback of the use of Web caches. (5p)

v) Describe an innovative Internet-based application or platform, after motivating it and describing its main objectives. (10p) Then describe its requirements regarding data loss and timing constraints. (5p) Discuss the proposed transport layer protocol that you will employ. Finally, outline the key performance indicators (KPIs) and user engagement metrics for assessing its performance. (10p) To argue about its innovative aspects briefly indicate its differences from well-known related applications/platforms.

Exercise 2 - HTTP GET/RESPONSE (15 points)

A. 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 <cr><lf> are *carriage return* and *line-feed* characters. **Answer the following questions, indicating where in the HTTP GET message below you find the answer.**

i) What is the name of the file that is being retrieved in this GET message?

ii) What version of HTTP is the client running?

iii) What type of files does the client accept?

iv) What is the client's preferred version of English?

v) What is the client's least preferred version of English?

vi) Will the client accept the German language?

vii) Does the client already have a cached copy of the file?

viii) Does the browser request a non-persistent or a persistent connection?

ix) What is the IP address of the host on which the browser is running?

x) What type of browser initiates this message? Why is the browser type needed in an HTTP request message?

```
GET /kurose_ross_sandbox/interactive/quotation4.htm
HTTP/1.1<cr><lf> Host: gaia.cs.umass.edu<cr><lf> Accept:
text/plain, text/html, text/xml, image/gif, image/png, audio/mpeg,
audio/vnf.wave, video/mp4, video/mpeg<cr><lf> Accept-Language:
en-us, en-gb;q=0.6, en;q=0.4, fr, fr-ch, da, de<cr><lf>
Keep-Alive: 300<cr><lf> Connection:keep-alive<cr><lf>
If-Modified-Since: Mon, 21 Oct 2024 02:05:42 -0700<cr><lf>
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; rv:11.0)
Gecko/20100101 Firefox/11.0<cr><lf><cr><lf>
```

B. Consider now the following string of ASCII characters that were captured by Wireshark when the browser sent an **HTTP RESPONSE** message. **Answer the following questions, indicating where in the HTTP RESPONSE message below you find the answer**.

i) Was the server able to send the document successfully?

ii) What time was the document reply provided?

iii) How big is the document in bytes?

iv) Is the connection persistent or nonpersistent?

v) What is the type of file being sent by the server in response?

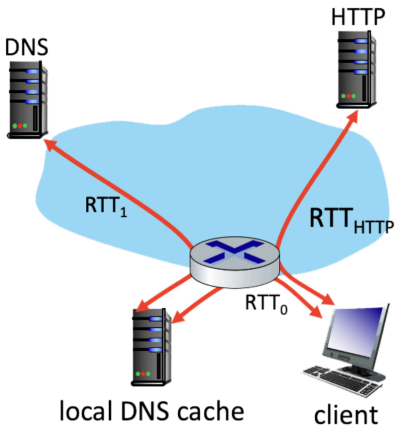
vi) What is the name of the server and its version?

vii) Will the ETag change if the resource content at this particular resource location changes?

```
HTTP/1.1 404 Not Found<cr><lf> Date: Mon, 21 Oct 2024 09:51:44
+0000<cr><lf> Server: Apache/2.2.3 (CentOS)<cr><lf> GMTETag:
”526c3-f22-a88a4c80”AcceptRanges: bytes<cr><lf> Content-Length:
59780<cr><lf> Keep-Alive: timeout=50, max=96<cr><lf> Connection:
Keep-alive<cr><lf> Content-type: image/html<cr><lf><cr><lf>
```

Exercise 3 - HTTP Response Time (20 points)

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 two 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 = 2 msecs. The second DNS server contacted has an RTT of 49 msecs. 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 RTTHTTP = 17 msecs.



i) Assuming zero transmission time for the HTML object, how much time (in msec) elapses from when the client clicks on the link until the client receives the object? (4p)

ii) Now suppose the HTML object references 3 very small objects on the same server. Neglecting transmission times, how much time (in msec) elapses from when the client clicks on the link until the base object and all 3 additional objects are received from the web server at the client, assuming non-persistent HTTP and no parallel TCP connections? (4p)

iii) Suppose the HTML object references 3 very small objects on the same server, but assume that the client is configured to support a maximum of 5 parallel TCP connections, with non-persistent HTTP. (4p)

iv) Suppose the HTML object references 3 very small objects on the same server, but assume that the client is configured to support a maximum of 5 parallel TCP connections, with persistent HTTP. (5p)

v) What are the differences between these methods: (3p)

- **a)** Non-Persistent HTTP (without parallel connections)
- **b)** Persistent HTTP without pipelining
- **c)** Persistent HTTP with pipelining

Exercise 4 - DNS Theoretical (15 points)

i) Name 4 services provided by DNS (4p)

ii) What's the role of the different DNS servers? (2p)

iii) Why isn't a single DNS server used to handle all queries? (2p)

iv) How does DNS caching work? (2p)

v) How does DNS load balancing work? (2p)

vi) Explain the role of DNS in content delivery networks (CDNs) (3p)

Exercise 5 - DNS Queries (10 points)

Suppose a web browser wants to know the IP address of www.csd.uoc.gr. Describe the name resolution process assuming:

i) an iterative query is used
ii) an iterative query is used

Make sure to list all the different DNS servers that are involved in the name resolution process. Which type of query is considered best practice: iterative or recursive? Explain your answer.

Exercise 6 - Nslookup (10 points)

For this exercise you can use the department's Linux machines. ***Make sure to provide a screenshot with the output of each command that you run.***

`Nslookup` is a command that operates by sending queries to DNS servers to retrieve information about domain names, IP addresses, and other DNS records.

Run the following command to retrieve the record for the host www.nasa.gov:

```
$ nslookup www.nasa.gov
```

i) What type of DNS record does this command return?
ii) What is the IP address of the DNS server that provides the information?
iii) What type of DNS server provides the answer?
iv) What is the IP address of the host nasa.gov? How many IP addresses are provided in the answer?

Now let's try to find some information about authoritative DNS servers. Firstly, run **`nslookup`** on your terminal (without any arguments) and then you should be able to see an angle bracket prompt (>).

Run the following:

```
> set querytype=ns
> forth.gr
```

v) What does "`querytype=ns`" do? What type of DNS records does it return ?
vi) What information is provided in the `Non-authoritative answer` section ?

Let's try to obtain the IP address of one of the authoritative DNS servers for `forth.gr`. Use the **`server`** command, followed by the name of one of the authoritative DNS server names.

```
> server [nameserver]
> set querytype=any
```

vii) What does the `server` command do?

viii) Run `> forth.gr` again. Which server provided the answer this time?

Exercise 7 - Telnet (5 points)

For this exercise we will use **`telnet`** to send an email. First, log in to one of the department's Unix machines. Before we start, we need to find the SMTP server of CSD. To do so, run the following command:

```
$ nslookup [flag] csd.uoc.gr
```

Replace `[flag]` with the appropriate `nslookup` type (e.g. A, NS, SOA, MX). Briefly answer the following questions:

i) What does the previous command do?

ii) What port does SMTP use ? Does SMTP use TCP or UDP and why?

Now let's send an email ! Start by running the following commands:

```
$ telnet [SMTP server name] [port]

$ HELO [csdXXXX@csd.uoc.gr]
```

Where `csdXXXX@csd.uoc.gr` is your email address (without the brackets "[ ]").

```
$ mail from: [csdXXXX@csd.uoc.gr]
$ rcpt to: [csdXXXX@csd.uoc.gr]
$ data
```

Now we will start typing the actual email we want to send. First we type in the subject.

```
$ subject: hy335a 2024 assignment 2
```

And now type the body of the email. You can write anything you like.

```
$ [your text goes here]
```

Now we are done with writing our email and want to send it to the recipient. How do we indicate that ? Run the appropriate command on `telnet`. After that you should receive a response of the form "`250 2.0.0 Ok: queued as 1C68D3C035F`".

iii) Provide a screenshot of your terminal where all the commands from the previous steps and their outputs are visible.

iv) Log into `webmail.csd.uoc.gr` and find the email you just sent to yourself. Find the "source" of the email and provide a screenshot. An example on how to do that is shown below: