**Computer Networking**

**FALL 2019**

**PROJECT 1**

**Part I: HTTP**

1. Load the file ‘http-trace-1.pcapng’ in Wireshark and answer the following questions. The traces were collected when a simple and very short HTML file was downloaded by the client.
2. What languages (if any) does the client browser indicate that it can accept to the server? What versions of HTTP are being run by the client and the server?

Ans: HTML version of Client: HTTP/1.1

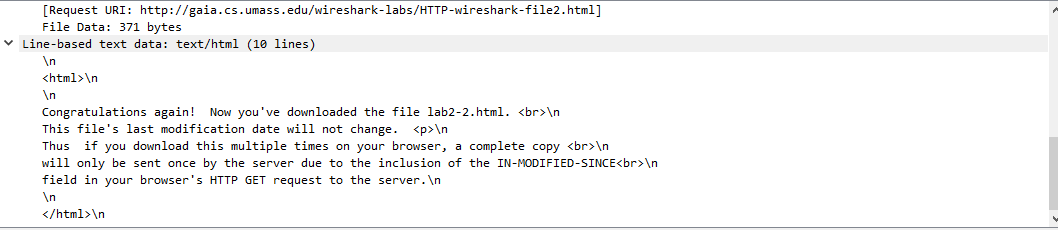
HTML version of Server:HTTP/1.1

1. What is the status code returned from the server to the client browser for the second GET request? What does the code mean?

Ans: 404 Not Found. It means that the page is not found.

1. Load the file ‘http-trace-2.pcapng’ in Wireshark and answer the following questions. The traces were collected when a particular web page is accessed twice from the browser within a short interval.
2. Inspect the contents of the first server response. Did the server explicitly return the contents of the file? If so, what was returned? (provide a screenshot)

Ans: Yes, the server explicitly returned the contents of the file.



1. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain why it did or did not.

Ans: 304 Status code Not Modified

1. Load the file ‘http-trace-3.pcapng’ in Wireshark and answer the following question.

The traces were collected when HTTP GET requests were sent for four image files (one PNG, three JPEGs). Can you tell whether the client browser downloaded the first two images (one JPEG and one PNG) serially or in parallel? Explain how you came to your conclusion.

Ans: By checking the TCP ports we can see if our files were downloaded serially or in parallel. In this case the images were transmitted over 2 TCP connections. Therefore they were downloaded parallel.

1. Enter the following URL into your browser and type the requested user name and password into the pop up box.

<http://gaia.cs.umass.edu/wireshark-labs/protected_pages/HTTP-wireshark-file5.html>

The username is “wireshark-students”, and the password is “network” (without the quotes).

1. What is the server’s response (status code and phrase) in response to the initial HTTP GET message from your browser?

Ans: 401 Not Authorized

1. When your browser sends the HTTP GET message for the second time, what new field is included in the HTTP GET message? Are the login credentials encrypted or sent as plain text?

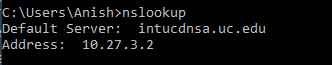
Ans: When your browser sends the HTTP GET message for the second time, the authorization basic field( d2lyZXNoYXJrLXN0dWRlbnRzOm5ldHdvcms) is included in HTTP GET message. The user credentials are shown under the column “Credentials” in Base64.

**Part II: DNS**

1. Run *nslookup* to determine the authoritative DNS server and its IP for www.uc.edu. Include screenshot in your answer.

Default Server: intucdnsa.uc.edu

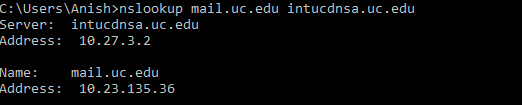
Address: 10.27.3.2



1. Run *nslookup* so that the authoritative DNS server obtained in Question 1 is queried for mail.uc.edu. Mention the IP address(es) for mail.uc.edu. Include screenshot.

Ans

:



1. Load the file ‘dns-trace-1.pcapng’ in Wireshark and answer the following questions.
2. Locate the DNS query and response messages. Are they sent over UDP or TCP? What is the destination port for the DNS query message and the source port of DNS response message?

Ans: They are sent over UDP. The destination port of the query is 53. The source port of the response is also 53.

1. Examine the DNS query message. What “Type” of DNS query is it?

Ans: The DNS Query is Type-A.

1. Examine the DNS response message. How many “answers” are provided? What does each of these answers contain?

Ans: There are two Answers for the DNS response message. They contain [www.ietf.org](http://www.ietf.org) in it.

1. Load the file ‘dns-trace-2.pcapng’ in Wireshark and answer the following questions.
2. Examine the fourth DNS response message (serial no. 19). What are the type, class, and address received for google.com?

Ans: google.com: type A, class IN, addr 216.58.217.238

1. Examine the tenth DNS response message (serial no. 39). What are the refresh interval and minimum TTL for UC mail server?

Ans: The refresh interval is :1200(20minutes). The TTL is 86400(1 day).

**Part III: Socket Programming**

1. Web Server

You will develop a web server that handles one HTTP request at a time. Your web server should accept and parse the HTTP request, get the requested file from the server’s file system, create an HTTP response message consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present in the server, the server should send an HTTP “404 Not Found” message back to the client.

Below you will find the skeleton code for the Web server. You are to complete the skeleton code. The places where you need to fill in code are marked with #Fill in start and #Fill in end. Each place may require one or more lines of code.

Put an HTML file (e.g., HelloWorld.html) in the same directory that the server is in. Run the server program. Determine the IP address of the host that is running the server (e.g., 128.238.251.26). From another host, open a browser and provide the corresponding URL. For example:

http://128.238.251.26:6789/HelloWorld.html

‘HelloWorld.html’ is the name of the file you placed in the server directory. Note also the use of the port number after the colon. You need to replace this port number with whatever port you have used in the server code. In the above example, we have used the port number 6789. The browser should then display the contents of HelloWorld.html. If you omit ":6789", the browser will assume port 80 and you will get the web page from the server only if your server is listening at port 80.

Then try to get a file that is not present at the server. You should get a “404 Not Found” message.

*You will hand in the complete server code along with the screen shots of your client browser, verifying that you actually receive the contents of the HTML file from the server.*

Ans: 

HTML Page:

<html>

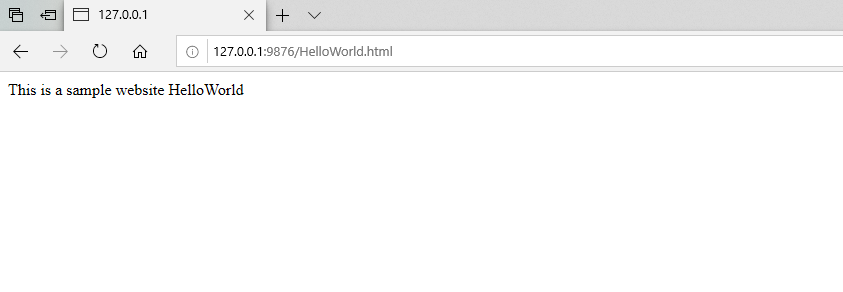
<head> This is a sample website </head>

<body> HelloWorld</body>

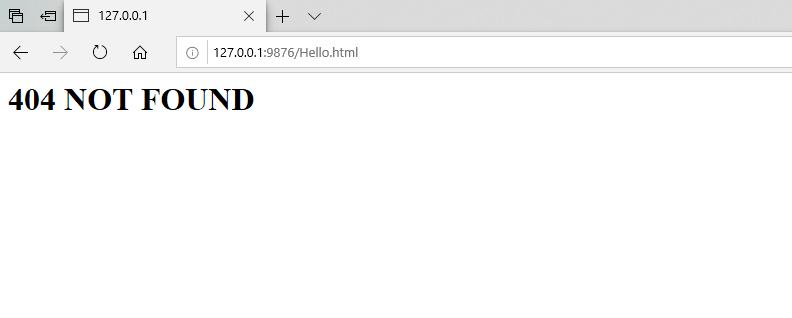
</head>

</html>

Screenshots:



Web Page Displaying the content of the page.



Error Page being displayed

2. Simple Mail Client

You will develop a simple mail client that sends email to any recipient. Your client will need to connect to a mail server, dialogue with the mail server using the SMTP protocol, and send an email message to the mail server. Python provides a module, called smtplib, which has built in methods to send mail using SMTP protocol. However, we will not be using this module in this lab, because it hide the details of SMTP and socket programming.

In order to limit spam, some mail servers do not accept TCP connection from arbitrary sources. For the experiment described below, you may want to try connecting both to your university mail server and to a popular Webmail server, such as a AOL mail server. You may also try making your connection both from your home and from your university campus.

Below you will find the skeleton code for the client. You are to complete the skeleton code. The places where you need to fill in code are marked with **#Fill in start** and **#Fill in end**. Each place may require one or more lines of code.

In some cases, the receiving mail server might classify your e-mail as junk. Make sure you check the junk/spam folder when you look for the e-mail sent from your client.

*In your submission, you are to provide the complete code for your SMTP mail client as well as a screenshot showing that you indeed receive the e-mail message.*

Skeleton Python Code for the Mail Client:



