IN423 - PW 03

HTTP POST and Network Analysis

1. Objectives and instructions

In this third practical work we will continue our previous work on TCP and HTTP. Follow following instructions.

2. HTTP

In this part we will see how to enhance our HTTP server to make it able to send different kind of data (encoding) and receive POST requests with data.

HTTP POST:

We are going to analyze the post requests made by the client. A post request is composed as follow: an http header ending with an empty line, followed by the data.

1 - Adapt your http server so it displays the entire requests send by the client.

2- Start your http server

3 - Use your web browser to open the control.html page and click on one button.

4 – Take a look at the POST request (print by the server)

2.1 What is the value (data) associated with the post? Where it is located on the request body (i.e. the received message) ?

```
 POST /control.html HTTP/1.1
Host: 127.0.0.1:55000
Connection: keep-alive
Content-Length: 13
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="112", "Google Chrome";v="112",
"Not:A-Brand";v="99"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://127.0.0.1:55000
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/112.0.0.0
Safari/537.36
```

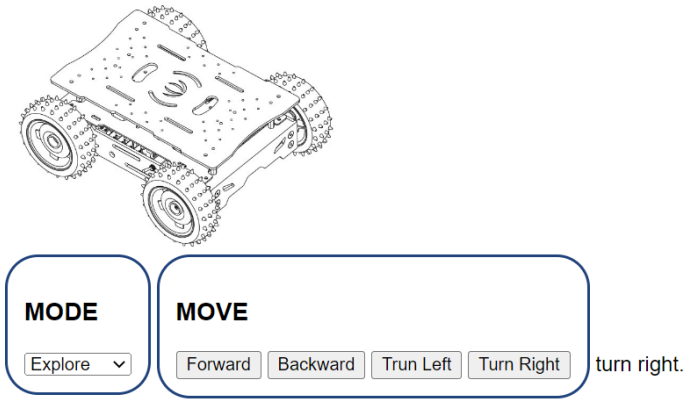
```
Accept: text/html,application/xhtml+xml,application/
xml;q=0.9,image/avif,image/webp,image/apng,*/
*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://127.0.0.1:55000/control.html
Accept-Encoding: gzip, deflate, br
Accept-Language: fr-FR,fr;q=0.9,en-US;q=0.8,en;q=0.7

move=leftturn
sending data...
```

The value associated with the post here is “lefturn” because it’s the button I clicked on. It is located after the blank line, after the header.

2.2 Adapt the server so that it prints “move forward”, “move backward” etc. depending on the post request.

2.3 Adapt your server so that is replies to the post request with an HTTP 200 message and the control.html page as data.

```python
if method != "GET":
    if method == "POST":
        print(data_decoded)
        click = data_request[-1][5:]

        if click =="forward" :
            response = "move forward."

        if click =="backward" :
                response = "move backward."

        if click =="leftturn" :
                    response = "turn left."

        if click =="rightturn" :
                    response = "turn right."

        r = ("HTTP/1.1 200 OK\r\n").encode('ascii')
        reply = "Date: Tuesday, 20 Apr 2023 10:48:13 GMT\r\n"
        reply += "Expires: -1\r\n"
        reply += "Cache-Control: private, max-age=0\r\n"
        reply += "Content-Type: control.html\r\n"
        reply += "charset=ISO-8859-1\r\n\r\n"
        f = open(Path+data_ressource,'rb')
        dataf= f.read()
        f.close()
        http_response = r+reply.encode("utf8")+dataf + response.encode('utf8')
```



HTTP 304 Not Modified:

/!\ For now this part is only working with google chrome /!

Here is an example of an HTTP 304 message that tell the client that the requested resource is not modified and that it should not refresh the page.

```
HTTP/1.1 304 Not Modifed\r\n
Date : Mon, Apr 26 07:43:53 2021 GMT\r\n
Expire : Mon, Apr 26 08:43:53 2021 GMT\r\n
Etag : ipsav1
Cache-control : max-age=3600\r\n
\r\n
```

2.4 **Use this example, but adapt values of Date and Expires options to respond to the image request when it has already been send.**

/!\ For now this part is only working with google chrome /!

The date should be current date: `time.asctime(time.gmtime())`

The Expires field should be filled with :

`time.asctime(time.gmtime(time.time()+3600)) + " GMT"`

explanation:

time.time() gives current time in seconds

time.time()+3600 gives current time in seconds + 3600 seconds

time.gmtime() translate time to GMT

time.asctime() make it a string

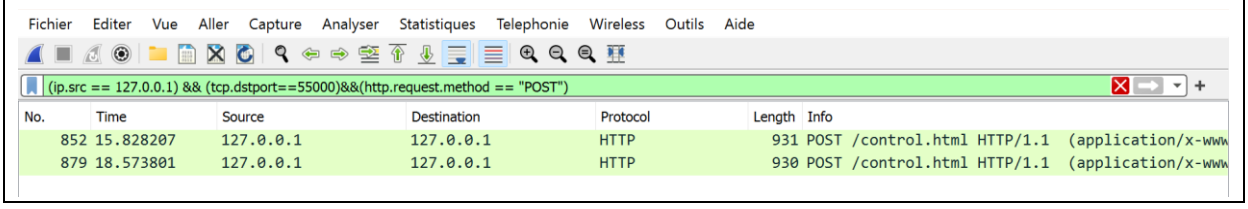
3. Network analysis

Use wireshark to capture POST requests and responses of your server.

3.1 What is the filter you have used

```
(ip.src == 127.0.0.1) && (tcp.dstport==55000)&&(http.request.method == "POST")
```

3.2 Copy your results here:



Use wireshark to capture your HTTP 304

3.3 What is the filter you have used ?

1. Give the display filter used to filter the HTTP response code 404 not found messages

http.response.code == 304

3.4 Copy your results here:
