About HTTP Client

Gabriele Biagini

September 2020

Contents

1 Context

The Hypertext Transfer Protocol is the fundamental application protocol for data communication in the World Wide Web with a request-response pattern. It presumes an underlying and reliable Transport Layer protocol: TCP is usually the preferred one.

2 Overview

This project purpose is to build a simple HTTP Client to perform requests through an **interactive CLI**.

The program will ask the user to enter parameters to build the HTTP request following the **RFC-7230**. It will then perform the request and print out the HTTP response.

3 Socket Library

The Operative System makes possible to access the Transport Layer via the Socket Library.

Sockets are descriptors that identify the communication channel depending on the role of the software (client or server).

They can be created via the function:

```
int socket (int domain, int type, int protocol);
```

In this project the created socket is a network socket that relies on a TCP transport (set by the AF_INET and SOCK_STREAM parameters).

The connection to the server is made via the function:

```
int connect (int sockfd, const struct sockaddr *addr, socklen_t
 addrlen);
```

4 The Client

It is possible to compile the client with a single command:

```
gcc http.c -o http -Wall -Wextra
```

And launch it:

```
./http
```

4.1 Sending a Request

The client will ask the user to enter the parameters in order to build the HTTP request. Because of the nature of the HTTP request line, it is virtually possible to build and send any kind of requests.

An example of a simple POST request:

```
Enter method (default GET): POST
Enter host (default httpbin.org):
Enter port (default 80):
Enter path (default /): /post
Enter general header (leave blank to finish):
Enter request header (leave blank to finish): Connection: close
Enter request header (leave blank to finish):
Enter body (leave blank to finish):
```

The parameters inserted will generate the following request line:

```
POST /post HTTP/1.1
Host: httpbin.org:80
Connection: close
```

And after a short interval to establish the connection:

```
Socket successfully created..
Host resolution done..
Address built..
Connected to the server..
```

The request will be fired and the client will read the response:

```
Response:
HTTP/1.1 200 OK
Date: Tue, 08 Sep 2020 14:48:42 GMT
Content-Type: application/json
Content-Length: 263
Connection: close
Server: gunicorn/19.9.0
Access-Control-Allow-Origin: *
Access-Control-Allow-Credentials: true

{
  "args": {},
  "data": "",
  "files": {},
  "form": {},
  "headers": {
    "Host": "httpbin.org",
    "X-Amzn-Trace-Id": "Root=1-5f5799ca-4d7608b8149114f2c63a59ea"
  },
  "json": null,
  "origin": "88.217.181.150",
  "url": "http://httpbin.org/post"
}
```