Week-end Challenge
03

Subject:

For this challenge, you will learn the basics of network programming by creating a small TCP client in C.

Rules:

- Only 1 submission per student, feel free to work in teams but do not submit the same code
- Your code has to be written in C and compile with the following flags: `-Wall -Wextra -Werror`
- Try to respect the norm as much as you can to make it readable, but also feel free to take some liberties if you want to: longer functions (more than 25 lines), the use of `switch` and `for`, etc

Mandatory part:

program name	mini-tcp
Turn-in files	client.c
Arguments	the domain and port you use to send the request
Authorized libraries	anything from the **libc**
Description	Your program will create a connection with a remote server.

General description

You will create a program using the `Makefile`, `wec03.h` and `hash_file.c` file, provided in the resources section. The only file you will write yourself is `client.c`.

Your project will have to realise a few simple tasks:

- You create a connection with a remote TCP server whose domain is `challenge.cln.ac` on port `1942`
- You send the following string: `? [login] [hash]` where login is your intra [login] and [hash] is the hashcode of your `client.c` file (see next section for more info)
- The server responds with the following string: `= [code]` where [code] is a random integer between 0 and 46340

- You return the following string: `! [code2]` where [code2] is the code the server gave you power 2.
- If all the informations where correct, the server will send `success` and close the connection.

The objective of the project is to send a valid input to the remote server as fast as possible. Every valid requests will be saved to a database.

Note: even if you sent a valid request, you have to send a new one to the remote server if you changed your code. It will update your hash, but also the submission time. The client.c file you submit has to be the exact one you used to make the request.

Hashing

In this project, you will be asked to send the result of your client.c file through a hashing function. Don't worry, we already wrote this function (see Ressources section), and here is how you can use it:

`unsigned long hash = hash_file(“./client.c”);`

The resulting number will have to be sent along with your login.
This is done to prevent cheating, because we will hash the file you turn-in, and we'll check if the hash number matches the one you sent to the server.

Note: the hashing function gives a different output even if you only changed one character from your file. Therefore, we heavily recommend you not including a 42header to your client.c file because the 'Updated: ' line will change if you save your file again.

Simulation

The remote server automatically ends the connection after 3 seconds to prevent you from manually enter the input. To make it easier to understand how to contact the remote server, you can use the simulation server whose domain is `challenge.cln.ac` on port `1943`. Your entries on this server will not be sent to the database. Here is an example of a valid command-line use (yellow lines are entered by the user):

```
>  telnet challenge.cln.ac 1943
Trying 51.178.40.137...
Connected to 137.ip-51-178-40.eu.
Escape character is '^]'.
? ancoulon 7894307453
= 3758
! 14122564
success
```

```
Connection closed by foreign host.
>
```
Obviously, you will have to do this in C, without using pre-made programs like telnet.

Ressources:

- Provided files
- WEC GitHub repository
- Network programming guide
- TCP client-server implementation in C (also a guide)

Turn-in:

The first step in the submission of your project executing with the actual remote server(port 1942). Once this is done, do not change your client.c file in any way, and send it by email to `tutors@s19.be` with the following subject: `wec03 - [login]`. The deadline is Sunday at 10:19pm.

Results:

The results will be announced during the following Monday's AMA. Points will be distributed according to the order in which the remote server received valid entries. Consider this challenge a race!