TP5 Système d'exploitation

Antoine Maendly

Architecture

- 1 fichier client.c pour le client
- 1 fichier server.c pour le server
- et une seul fonction additionnel pour le server

```
// Fonction qui renvoie un nombre aléatoire dans l'interval donné en argument
int random_number(int min, int max)
```

Makefile

```
CC = gcc

all: server client

clean:
    rm server client

server: server.c
    $(CC) server.c -o server

client: client.c
    $(CC) client.c -o client

run_client:
    ./client 127.0.0.1 8082

run_server:
    ./server 8082
```

Message client/server

Dans le pdf du tp, il est dit qu'on envoie un message de deux octets, avec le premier octet etant la commande et le deuxième la proposition. Je ne voyais pas comment cela était possible étant donné que un short fait 2 octets. Si on envoie un char qui fait bien 1 octet, on ne pourrait pas donner un nombre plus grand que 9. J'ai donc fais une array de 2 int qui représente notre message.

Test du programme avec 2 machine sur un réseau local

Après avoir fini, le programme, j'ai testé mon programme en local sur la même machine ainsi que avec deux machines différentes (windows et mac). Les programmes client et server réponde comme attendu, a condition que le pare feu ne restreint pas la connection entre les deux machines.

Exemple du programme

Côté Server

```
Client 5 connecté avec l'ip 127.0.0.1
La valeur 37 est choisie pour le client 5

--------- For 127.0.0.1 : 37 ---------
Client 5 propose 10
Attempts left : 9

--------- For 127.0.0.1 : 37 ---------
Client 5 propose 23
Attempts left : 8

Client 5 won :)
Closing socket for Client 5 with ip 127.0.0.1.
```

Côté Client

```
-------- Game --------
Try to find the exact number.
The number is between 10 and 50.

Enter your number : 10

Proposition envoyée : 10
La valeur réelle est: 37
Too low !

Enter your number : 23

Proposition envoyée : 23
La valeur réelle est: 37
Too low !

Enter your number : 37
```

```
Proposition envoyée : 37
La valeur réelle est: 37

You Won !
```