

Pente : Echo Server

D.Moreaux

21 septembre 2024

1 Code à réaliser

On commencera par mettre en place la connexion réseau et la boucle principale qui permet de gérer cette dernière ainsi qu'un client python de base.

A cet effet, le code ouvrira le port 55555 en TCP en écoute et puis, une boucle basée sur la fonction `select()` se chargera d'accepter les connexion, de lire les données reçue et de retourner les messages reçus à l'expéditeur.

Ce code pourra être testé à l'aide d'un client telnet comme PuTTY ou d'un programme python basique qui effectuera la connexion et utilisera la fonction `input()` pour demander un texte et `print()` pour afficher ce qui sera reçu du serveur (pas d'interface graphique à ce niveau).

2 Le serveur

Le code pourra être effectué en plusieurs étapes.

2.1 Ecoute du port TCP/55555

La première version du code se contentera d'ouvrir le port, accepter une connection, lire un message de cette connexion et réécrire le dit message sur le socket avant de fermer les sockets (connexion et socket de réception) et quitter.

Le serveur affichera éventuellement l'adresse IP de la personne se connectant.

2.2 Boucle de réception

L'étape suivante sera de mettre en place un tableau¹ ou une liste liée de connexions.

Une fois la connexion lancée, une boucle sans fin fera les étapes suivantes :

- Préparer la liste des sockets en lecture et exception pour le socket de contrôle et les sockets des connexions en cours
- Effectuer un `select()` pour détecter les connexions ayant de l'activité
- Si une nouvelle connexion est faite, l'ajouter dans la table ou liste de connexions.
- Si des données peuvent être lues, les écrire en retour sur le même socket²
- Si une exception a lieu, fermer le socket correspondant et retirer la connexion de la liste ou du tableau.

3 Le client

Le client utilisera l'interface socket. Il commencera par créer la connexion puis fera une boucle sans fin dans laquelle il lira une chaîne. Si cette chaîne est "quit", le programme fermera la connexion et s'arrêtera.

Sinon, il enverra la chaîne au serveur, lira le retour et affichera ce dernier avant de boucler.

1. on peut utiliser TEMPORAIREMENT un tableau mais au final, il faudra utiliser une liste liée pour permettre une allocation dynamique

2. Plus tard, on ajoutera des buffers d'entrée et de sortie