

## TD : Les sockets Linux

### Objectifs :

#### *Mise en œuvre des sockets en langage C sous Linux.*

---

Ce TD a pour but de développer un client et un serveur TCP/IP. Le serveur écoute sur un port donné et accepte des connexions clientes. Le programme client se connecte au serveur, puis lui envoie un message. Le message est analysé par le serveur, qui lui répond.

### **Partie 1 : création d'un client TCP**

1. Écrire le programme principal *ClientTCP.c* correspondant à un client TCP. Le programme se contentera d'envoyer une chaîne de caractères au serveur et attendra une chaîne de caractères en retour qu'il affichera. Le programme recevra en paramètres et dans cet ordre, l'adresse IP du serveur, le numéro du port d'écoute du serveur, puis la chaîne de caractères à émettre.
2. Testez votre client à l'aide de l'outil réseau **netcat** en mode client sur le port d'écoute 6000 : `nc -l -p 6000`

### **Partie 2 : création d'un serveur TCP mono client :**

3. Écrire le programme principal *ServeurTCP.c* correspondant à un serveur TCP. Le programme recevra une chaîne de caractères qu'il affichera. Il convertira ensuite en majuscule cette chaîne et la retournera au client. Le port d'écoute du serveur sera passé en paramètre au programme.
4. Testez votre serveur à l'aide du client de la partie 1.
5. Modifiez le client précédent afin de lui faire afficher son adresse d'émission, son numéro de port et l'adresse du serveur qui lui répond.
6. Testez votre nouveau client.

### **Partie 3 : création d'un serveur TCP concurrent :**

7. Écrire le programme principal de votre serveur TCP/IP. Le serveur écouterait dans une tâche sur toutes les cartes réseaux disponibles et sur un numéro de port spécifié en paramètre. Lors d'une connexion client une tâche spécifique de prise en compte de la connexion client, dite "tâche de service" sera créée afin :
  - a. De lire la chaîne de caractères envoyée par le client,
  - b. De l'afficher sur le terminal,
  - c. De répondre au client par un message d'accusé de réception qui sera également passé en paramètre au programme.
8. Testez votre nouveau serveur afin de mettre en évidence l'aspect concurrent en activant plusieurs clients par la méthode de votre choix.

9. Modifiez le serveur afin qu'il attende également un 3<sup>e</sup> paramètre signalant la chaîne de caractères qui, lorsqu'il la recevra, devra "tuer" le serveur après avoir fermé tout ce qui est nécessaire.
- 10. Testez votre serveur en réalisant des échanges à partir de plusieurs clients.*
11. Compléter votre serveur afin de lui faire afficher l'adresse et le numéro de port du client pour lequel il a accepté une connexion avant d'en afficher le message.
- 12. Testez votre nouveau serveur*