TP n°4 : Client/Serveur TCP	Durée : 8 heures
Principale compétence terminale concernée	Savoir et savoir faire associés
C4.1 Câbler et/ou intégrer un matériel C4.1 Adapter et/ou configurer un matériel C4.4 Développer un module logiciel C5.2 Mettre en œuvre une solution matérielle/logicielle en situation	SF35. Réaliser la mise en situation et interconnecter du matériel. S7.7 Programmation réseau S7.7.1 Concept Client/Serveur S7.7.2 Sockets POSIX

Niveau	2ere année BTS SN IR
Configuration du poste	Possède un système Linux avec le compilateur gcc
Données fournies	Cours Réseau API Socket
Prérequis	Maîtrise du langage C

1] Client/Serveur TCP simplifié : programme srv_tcp1 et clt_tcp1

Écrire et tester les programmes client/serveur simplifiés. Simplifié signifie tout dans le main() avec le minimum de fonctions et pas de paramètre, n°IP et n° port sont fixés dans le code. Le serveur ne se détache pas du terminal, il inclut le service qui est monoclient et monocoup.

Le service affiche les messages reçus et retourne au client : message n°x reçu, avec gestion de l'incrémentation de x. Il s'arrête en fin de connexion.

Le client invite l'utilisateur à entrer un message puis l'envoie au serveur. Il s'arrête sur l'envoi du message "fin".

Tester

a) en local

b) avec une machine distante

2] Client/Serveur TCP squelette : programme srv_sql_tcp et clt_sql_tcp

Même fonctionnalité du client et du serveur/service que précédemment mais cette fois-ci le serveur est concurrent et le service est codé en dehors du serveur. Il faut utiliser les programmes squelettes fournis et compléter le code.

Tester

a) en local

b) avec une machine distante

c) Le service affiche le n° de port et le n°IP du client connecté ainsi que son nom de machine.

3] Caractéristiques TCP (Mettre en évidence la non perte de données)

Écrire les mêmes client/service que pour le TP UDP mais cette fois-ci en TCP.

Le client permet d'envoyer des trames dont le nombre et la taille sont paramétrables.

Le service affiche la taille de chaque message reçu et comptabilise son nombre. La fin est réalisée par le mécanisme du shutdown() (voir cours). Avant de se terminer, on affichera le nombre de messages reçus et la taille totale d'octets reçus. Pour simuler une charge au niveau du service, inclure des temps de pause entre chaque lecture.

Conclusion ?

4] Client/Serveur TCP : transfert de fichier

a) Client TCP
Le programme sqclt_tcp s'exécutera de la façon suivante : sqclt_tcp nom_machine n°port nomFichier
Compléter la fonction clientService() de ce programme pour réaliser l'envoi du nom de fichier passé en paramètre, puis la totalité du contenu du fichier.

Rappel: Attention on écrit pas forcement le nombre d'octets passé à la fonction write() !!
Pensez à utiliser au moins 3 variables : int pos, nbwn, nbf ;
nbf : nombre de caractères lu à partir du fichier
nbwn : nombre de caractères envoyé (write) sur le réseau (network)
pos : position précisant la position dans le buffer des données à transmettre

b) Serveur TCP
Vous utiliserez le programme sqsrv_tcp tel que et développerez le service qui doit récupérer le nom du fichier au travers du réseau, le créer en local puis le remplir des données provenant aussi du réseau.

Au final on a réalisé le transfert d'un fichier du poste client vers le poste serveur. Vérifier le résultat.

Option serveur:

- Afficher le n° de port et le n°IP du client connecté ainsi que son nom de machine. (voir fonction getpeername())

- Écrire la fonction int extraire(char *chemin_fichier);
  Cette fonction retourne la position du nom du fichier dans un chemin
  Exemple : i = extrait("/home/user/document.txt"); // la fonction retourne la valeur: 11, c-a-d la position du 'd')

- Modifier le serveur pour le rendre mono-client.