2023-2024

TP 5 – applications avec un seul client (et un seul serveur)

Pour simplifier, on suppose que les blocs échangés ont tous une taille < 128 octets.

Exercice 1 : améliorations de l'application « écho » UDP

a) Reprendre les codes client et serveur UDP vus en cours et insérer des affichages pour suivre le déroulement de l'application. Tester l'application.

b) Créer un module regroupant toutes les fonctions suivantes :

- ✔ preparer_serveur() : inclus socket() et bind()
- ✔ preparer_client() : inclus socket()
- ✔ analyser() : pas d'analyse pour « echo » => affichage de la requête reçue
- ✔ utiliser() : pas d'utilisation particulière de la réponse reçue pour « echo » => affichage de la réponse
- ✔ construire_requete() : « echo » => requête = simple chaîne à saisir
- ✔ construire_reponse() : « écho » => réponse = requête
- ✔ envoyer() : inclus recvfrom()
- ✔ recevoir() : inclus sendto()
- ✔ arreter() : inclus close()

Attention : les fonctions peuvent avoir plusieurs arguments non précisés ici ainsi que retourner plusieurs valeurs également non précisées ici.

c) Modifier les codes de a) grâce aux fonctions du module créé en b). Tester l'application.

d) Modifier les fonctions du module pour y inclure tous les tests concernant les fonctions « socket ».

e) Modifier les codes de d) pour rendre cette application itérative. Pour cela, choisir un bloc (saisi par le client) qui servira de message d'arrêt des échanges puis des programmes. Les programmes doivent se terminer le plus proprement possible.

Exercice 2 : application « comptage et tri de requêtes »

Sur la base d'un C / S UDP, créer une application itérative qui :

- ✔ demande à l'utilisateur de choisir un pseudo au lancement du client
- ✔ chaque requête est constituée de : pseudo + séparateur + message
- ✔ chaque réponse est constituée de : message + séparateur + code + séparateur + compteur
  avec :
  - « message » : celui que le serveur a reçu dans la requête (comme « echo » en somme)
  - « compteur » : compte toutes les requêtes reçues par le serveur, indépendamment du client et du pseudo
  - « code » = 'c' si message < 20 octets ou = 'l' dans les cas contraires
- ✔ Le client doit afficher chaque réponse reçue et séparément le message, le « code » et la valeur de « compteur »

Le choix du séparateur est libre. De même, le choix du « message » permettant d'arrêter l'application est libre.

Conseil : commencer par une version non itérative.

Exercice 3 : améliorations de l'application « écho » TCP

Effectuer la même démarche que dans l'exercice 1. Créer un module spécifique. Attention, certaines fonctions sont relativement différentes par rapport à UDP ; et avec TCP, il en faut 2 de plus …

Pour ce qui est de rendre l'application itérative, ne pas oublier que TCP offre 2 possibilités : tous les échanges dans une seule connexion ou une connexion par échange. Programmer les 2 cas.