Développement d’applications - Client/Serveur (Socket TCP et UDP)

TROTTIER Arthur

SEBILLE Florian

L2 SPI

ServeurEcho :

**1. Principe et organisation**

L’application a pour but de calculer le temps d’aller/retour entre un client et un serveur. Pour cela, l’heure sera calculée avant l’envoi de la requête puis après la réponse du serveur au client. Ensuite il suffira de faire une soustraction entre le deuxième temps et le premier. Ci-dessous l’organisation :

**SERVEUR CLIENT**

Création de la socket

Initialisation adresse IP client

Création de la socket

Initialisation adresse IP serveur

Bind

Bind

Connexion au serveur

Listen

Calcule de l’heure

Attente de données

Accept

Envoie de la requête Echo

Réception de la réponse du serveur

Réception de la requête

Calcule de la deuxième heure

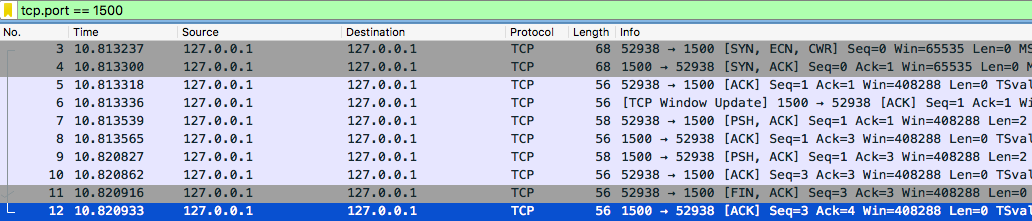
Envoie de la réponse « OK »

Soustraction puis affichage du temps

Déconnexion

**2. Wireshark et visualisation**

Pour visualiser les échanges client/serveur ainsi que les trames, nous pouvons utiliser le logiciel Wireshark. Ainsi on obtient ceci :

****

1

2

3

On impose au logiciel un filtre qui permettra d’afficher uniquement les trames où le port TCP 1500 entre en jeu (source ou destination).

Partie 1 (N°3 à 6) : Il s’agit du protocole de connexion entre le client et le serveur, on observe que la longueur de données des trames est nul (Len = 0). Il n’y a que le n° d’ACK et de SEQ qui change et qui passe à 1.

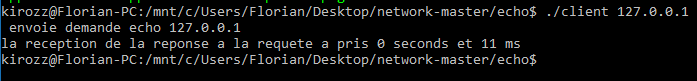
Partie 2 (N°7 à 10) : La connexion effectuée, les échanges de données peuvent désormais commencer. Pour chaque envoi, on a l’ACK venant du receveur qui va avec qui correspond au numéro de séquence de la trame reçu + LEN + 1.

Dans cet exemple on observe 2 échanges de données (1 requête du client et une réponse du serveur).

Partie 3 (N°11 à 12) : L’échange terminé, le client demande une déconnexion au serveur qui répond avec ACK semblable au SEQ de la demande + 1 (les données étant nulles).

**3. Code et visualisation du programme**

Figure 1/2 : Exemple de requête Echo

****

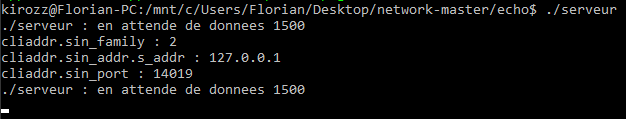
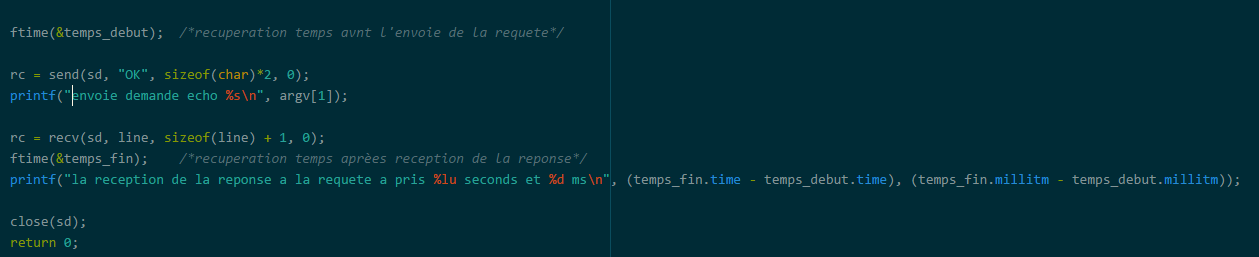


Figure 3 : Partie du code client

****

La première fonction ftime récupère l’heure avant l’envoi de la requête du client au serveur. Une fois la requête émise et la réponse renvoyée, on récupère une nouvelle fois l’heure. Ensuite pour l’affichage nous faisons la soustraction entre la deuxième heure et la première.

Pierre/Feuille/Ciseaux :

**1. Principe et organisation**

L’application a pour but de simuler une partie de pierre/feuille/ciseaux entre un client (Joueur A) et un serveur (Joueur B). Pour cela, après chaque tour du joueur A, le serveur renvoi le résultat aléatoire de la manche. La partie se joue en 3 manches, le joueur qui en remporte le plus gagne. Ci-dessous l’organisation :

**SERVEUR CLIENT**

Création de la socket

Initialisation adresse IP client

Création de la socket

Initialisation adresse IP serveur

Bind

Bind

Connexion au serveur

Listen

Si Cpt <= 3

Accept

Si Cpt <= 3

Demande choix joueur A + Envoi au serveur

Réception choix joueur B + affichage résultat

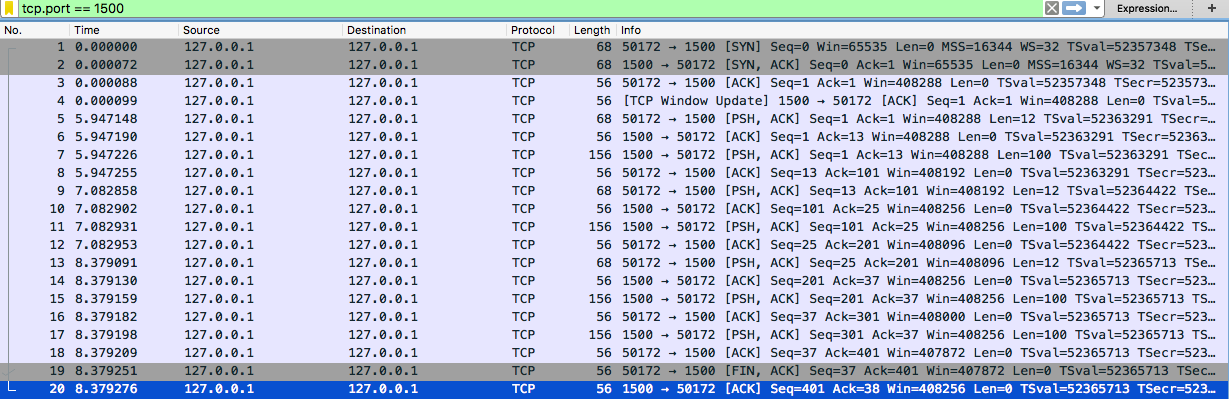
Réception choix joueur A

Déconnexion

Choix aléatoire joueur B + Envoi résultat

**2. Wireshark et visualisation**

Pour visualiser les échanges client/serveur ainsi que les trames, nous pouvons utiliser le logiciel Wireshark. Ainsi on obtient ceci :



1

2

3

On impose au logiciel un filtre qui permettra d’afficher uniquement les trames où le port TCP 1500 entre en jeu (source ou destination).

Partie 1 (N°1 à 4) : Il s’agit du protocole de connexion entre le client et le serveur, on observe que la longueur de données des trames est nul (Len = 0). Il n’y a que le n° d’ACK et de SEQ qui change et qui passe à 1.

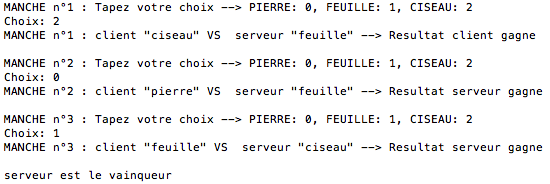
Partie 2 (N°5 à 18) : La connexion effectuée, les échanges de données peuvent désormais commencer. Pour chaque envoi, on a l’ACK venant du receveur qui va avec qui correspond au numéro de séquence de la trame reçu + LEN + 1.

Dans cet exemple on observe 7 échanges de données (3 envois du choix de A, 3 envois du choix de B + Envoi du résultat final).

Partie 3 (N°19 à 20) : L’échange terminé, le client demande une déconnexion au serveur qui répond avec ACK semblable au SEQ de la demande + 1 (les données étant nulles).

**3. Exemples de trames et visualisation du programme**

Figure 1 : Exemple d’une partie (Affichage coté client)

****

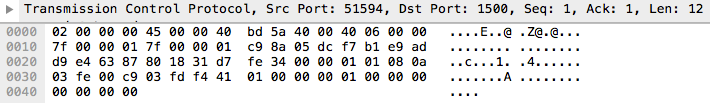
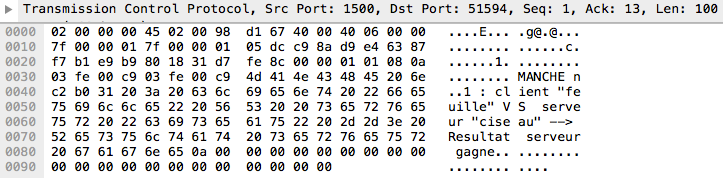
****

Figure 3 : Exemple d’une trame de données (Serveur vers Client)

Figure 2 : Exemple d’une trame de données (Client vers Serveur)