UNIVERSITÉ DE ROUEN	ANNÉE
FACULTÉ DES SCIENCES	RÉSEAUX
LICENCE 3 INFORMATIQUE	FICHE DE TD/TP N° 5

Primitives de base pour la manipulation des sockets internet en mode connecté (TCP)

Nous allons créer une bibliothèque pour la programmation des sockets Internet pour le protocole TCP. Cette bibliothèque sera utilisée pour écrire des protocoles réseaux en mode connecté dans le domaine Internet. Pour réaliser ce travail, vous utiliserez les fonctions définies dans `<sys/socket.h>`
Le type `SocketTCP` est une structure permettant d'accéder aux informations suivantes :

- un descripteur pour manipuler une socket BSD (vaut -1 si la socket n'est pas encore créée) ;
- l'adresse de la machine locale (lorsque la socket est attachée) ;
- l'adresse de la machine distante (lorsque la socket est connectée) ;

Exercice 1

1) Rappeler le schéma général d'un serveur (d'un client) internet en mode connecté.

2) Rappeler les paramètres à passer à l'appel système `socket` pour créer une socket Internet en mode connecté.

3) Expliquer la différence entre les sockets en mode connecté et en mode non connecté. Qu'est-ce que cela implique sur les structures de données et les appels système utilisés ?

4) Définir un type

`SocketTCP`

permettant de manipuler une socket BSD en mode connecté.

Exercice 2

1) Décrire les appels systèmes qui seront utilisés pour implanter les fonctions demandées en TP.

Exercice 3

1) Définissez la fonction utilitaire

```
int initSocketTCP(SocketTCP *psocket);
```

qui initialise la socket `psocket` (préalablement alouée).

Valeur de retour : 0 en cas de succès, -1 sinon.

2) Définissez la fonction

```
int connectSocketTCP(SocketTCP *osocket, const char *adresse,
          uint16_t port);
```

qui se connecte sur une machine distante dont l'adresse et le port sont donnés en paramètre.

La fonction retourne 0 si la connexion est établie, -1 sinon. Cette fonction devra mettre à jour la structure `osocket` de type `SocketTCP` passée en paramètre.

3) Définissez la fonction

```
int creerSocketEcouteTCP(SocketTCP *isocket, const char *adresse,
          uint16_t port);
```

qui permet de créer une socket d'écoute et de l'attacher à l'adresse et au port donnés en paramètres. Si `adresse` est `NULL`, la socket `isocket` écoute sur toutes les interfaces.

De plus, cette fonction devra créer une file de connexions en entrée de taille `SIZE_QUEUE` sur `SocketTCP`. La constante `SIZE_QUEUE` est défini dans le fichier d'entête `SocketTCP.h`.

La fonction retourne -1 si la socket n'est pas créée ou si l'attachement n'a pu avoir lieu, 0 si tout s'est bien passé.

4) Définissez la fonction

```
int acceptSocketTCP(const SocketTCP *secoute, SocketTCP *sservice);
```

qui attend une connexion sur la socket d'écoute `secoute`, et met à jour la socket de service `sservice` lorsque la connexion est établie. Cette fonction est bloquante jusqu'à ce qu'une connexion soit établie. Elle retourne -1 en cas d'erreur, 0 si tout s'est bien passé.

Exercice 4

1) Définissez la fonction

```
int writeSocketTCP(const SocketTCP *osocket, const void *buffer,
          int length);
```

qui écrit sur `osocket` un bloc d'octets `buffer` de taille `length` et retourne la taille des données écrites.

2) Définissez la fonction

```
int readSocketTCP(const SocketTCP *nsocket, void *buffer, int length);
```

qui lit sur `nsocket` un bloc d'octets de taille au plus `length` dans `buffer` et retourne la taille des données réellement lues.

3) Définissez la fonction

```
int closeSocketTCP(SocketTCP *socket);
```

qui ferme la connexion dans les deux sens et libère l'espace éventuellement alloué par la `SocketTCP`.