1 Transfert de fichier

L'objectif de cet exercice est d'apprendre à réaliser des transferts de fichiers en TCP et mieux maîtriser le comportement de ce protocole. En d'autres termes, vous allez programmer un transfert FTP et comprendre les bases sur lesquelles s'appuie l'implémentation du protocole applicatif FTP.

Dans un premier temps, nous envisageons le transfert d'un fichier de petite taille. Plus précisément, nous choisirons des fichiers dont la taille ne dépasse pas la taille du buffer d'envoi de la socket de l'expéditeur, la taille du buffer de réception de la socket du récepteur et la taille du buffer de lecture d'un fichier. Ce transfert est à réaliser par une application avec :

- un programme client qui 1) lit **EN UNE SEULE FOIS** le contenu d'un fichier de petite taille et 2) envoie ce contenu **EN UNE SEULE FOIS** (un seul appel à send) au serveur. Le fichier à envoyer (comme tout autre fichier à transférer) est à ranger dans un répertoire "./emission" et son nom sera passé en paramètre de votre programme.
- un programme serveur qui tente de recevoir **EN UNE SEULE FOIS** (un seul appel à recv) le contenu d'un fichier et stocke ce dernier dans un répertoire "./reception".

1. définir un protocole d'échanges entre le client et le serveur pour réaliser ce transfert.
2. écrire les deux programmes client et serveur. Le programme serveur devra afficher le nombre d'octets reçus du contenu du fichier.
3. en utilisant deux machines différentes, tester votre application sur différents exemples de fichiers en respectant toujours une taille inférieure à celle des buffers d'envoi et de réception. Pour chaque test, le fichier envoyé et le fichier reçu sont-ils identiques ? Si oui passez à la suite, sinon, expliquer ce qui c'est produit et corriger votre code.
   Remarque : vous devez pouvoir faire des tests différents sans avoir à modifier votre programme, corriger sinon.

   A présent, tester votre application sur des fichiers dont la taille est supérieure à la taille du buffer d'envoi du client et/ou de réception du serveur.
4. Que se passe-t-il ? **Remarque** : à présent, la taille des tampons stockant le contenu d'un fichier dans la couche application peut dépasser la taille du buffer d'envoi ou de réception.
5. Expliquer le problème et corriger vos programmes. Si ce n'est pas déjà fait, le programme client devra afficher le nombre d'octets du contenu du fichier effectivement envoyés.
6. Est ce raisonnable de continuer à lire le contenu d'un fichier en une seule fois ? Expliquer pourquoi et améliorer votre programme client.
7. Résumer le comportement de TCP.

Aide

Pour la manipulation d'un fichier, vous pouvez utiliser les fonctions suivantes :

- pour l'ouverture d'un fichier :

  FILE * fopen(const char *restrict filename, const char *restrict mode) ;
- pour connaitre la taille d'un fichier :

  int stat(const char *path, struct stat *buf) ;
- pour lire dans un fichier :

  size_t fread(void *restrict ptr, size_t size, size_t nitems, FILE *restrict stream) ;
- pour écrire dans un fichier :

  size_t fwrite(const void *restrict ptr, size_t size, size_t nitems, FILE *restrict stream) ;
- pour fermer un fichier :

  int fclose(FILE *stream) ;

Enfin, pour connaitre la taille du buffer de réception d'une socket, vous pouvez :

- soit consulter le contenu de : /proc/sys/net/ipv4/tcp_rmem

– soit utiliser la fonction
int getsockopt(int socket, int level, int option_name, void *restrict option_value, socklen_t *restrict option_len) ;
en passant en paramètre SOL_SOCKET pour level, SO_SNDBUF (ou SO_RCVBUF) pour option_name, l'adresse d'un entier pour option_value et la taille d'un entier pour option_len. option_value permet de récupérer la taille du buffer d'envoi (ou de réception).
– soit une autre solution de votre choix.