L2 informatique : Travaux Dirigés UE ASR5 Système d'exploitation
13/03/2017
F.Rico

0. Nom et prénom :

Lorsque du code est demandé pour ces exercices, vous devez le faire en C/C++. Vous êtes autorisés à utiliser les fonctions vues en cours (`send`, `recv`, `read`, `fork`, `signal` ...) en TD ou en TP (`socklib, ...`).

Question1 ()

Programmation - 10 minutes Donnez le code d'une fonction :

```
int enregistre(const string &nomfichier, int taille, int sock);
```

Cette fonction doit ouvrir le fichier `nomfichier` et sauvegarder dans ce fichier le contenu des `taille` prochains octets reçus sur le socket. À la fin, la fonction doit fermer le fichier et retourner le nombre d'octets reçus si tout c'est bien passé ou -1 s'il y a eu un problème.

Solution:

```
int enregistre(const string &nomfichier, int taille, int sock) {
  ofstream f(nomfichier.str());
  if (!f) { // traitement erreur retour -1}
  int deja = 0;
  while (deja < taille) {
    char buff[100];
    int r = read(sock, buff, 100);
    if (r == -1) {//traitement erreur retour -1}
    if (r == 0) {break;}
    deja += r;
  }
  return deja;
}
```

Question2 ()

Mail - 3 minutes Le type MIME (Multipurpose Internet Mail Extensions) a été inventé pour les mail. Expliquez à quoi il sert dans les mails.

Solution: Le type MIME est une information ajoutée à un fichier que l'on transmet pour décrire ce qu'il contient. C'est le remplaçant de l'extention utilisée pour associer un logiciel à un fichier.
Dans le cas des mails il sert pour les pièces jointes. Si un mail contient une image jpeg, l'image est copiée dans le contenu du mail et précédée par une petite description qui contient le type MIME pour permettre au logiciel de lecture de mail savoir qu'il faut utiliser la librairie jpeg afin de l'afficher.

Question3 ()

Tunnel - 12 minutes

Faire un programme tunnel, c'est à dire un programme qui se lance avec 3 arguments :

```
$> ./tunnel.exx PORT_IN IP_DEST PORT_DEST
```

Il doit se connecter sur le serveur IP_DEST:PORT_DEST, écouter sur le port PORT_IN et attendre un seul client. Il doit alors transmettre tout ce que le client lui envoie au serveur IP_DEST:PORT_DEST.

Solution: J'utilise les primitives de socklib dans cet exemple, en peut de temps il est difficile de ne pas utiliser ces primitives pour les créations de socket. Mais pour tout le reste, vous pouvez aussi utiliser `recv/send` ou `read/write`

```
int main(int argc, char *argv[]) {
  if (argc != 4) {
    cerr << "Usage \" << argv[0] << " PORT_IN IP_DEST PORT_DEST"
         << endl;
  }
  int s_at = CreeSocketServeur(argv[1]);
  int s_in = AcceptConnexion(sat);
  int s_out = CreeSocketClient(argv[2], argv[3]);

  BufferedReaderWriter in(s_in);
  BufferedReaderWriter out(s_out);

  while (1) {
    vector<char> data = in.read();
    if (data.size() == 0) {
      cerr << "Fin du tunnel" << endl;
      break;
    }
    out.write(data);
  }

  close(sat);
}
```