PR6 – Programmation réseaux
TP n° 7 : Programmation réseaux en C (suite)

I) Encore un service

Exercice 1 : Connexion au service disponible sur le port 2628

1. Déterminer le nom du service qui se trouve sur le port 2628 et devinez ce qu'il fait.
2. La documentation de ce service est disponible sur Internet (RFC 2229). Parcourir cette documentation pour avoir une idée de ce que l'on peut faire avec ce service.
3. Se connecter à ce service sur la machine *lucien* avec la commande `nc` (qui fonctionne comme `telnet`) et tester quelques requêtes (une requête très utile : `HELP`).
4. Écrire un programme C qui prend en argument un mot et qui retourne sa définition si il la trouve. Pour cela utiliser le service disponible sur le port 2628.

II) Votre propre programme dictionnaire

Exercice 2 : Un client et un serveur pour un service de dictionnaire

Écrire un programme client et un programme serveur correspondant à un service de dictionnaire. Pour cela le client demande la définition d'un mot au serveur et celui-ci lui renvoie le texte obtenu par l'appel à l'utilitaire `dict` (voir `man dict` sur son terminal). Si `dict` ne trouve pas de définition, le serveur renvoie un code d'erreur au client. Le serveur ferme alors la connection avec le client. Le serveur devra pouvoir accepter plusieurs connexions simultanées. Pour cela, on vous demande de programmer cela d'abord en créant de nouveaux processus avec `fork()`, puis en créant des processus légers.

Indications : Pour récupérer le résultat de la commande `dict`, on pourra s'inspirer du programme suivant :

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(int charc,char * *argv) {
  char tab[50];
  char *res="";
  int i=1;
  int taille_entree=strlen(argv[1]);
  char * appel=malloc((6+taille_entree)*sizeof(char));
  strcpy(appel,"dict ");
  strcat(appel,argv[1]);
  FILE * f=popen(appel,"r");
```

```
  while(fgets(tab,50,f)!=NULL){
    char *temp=res;
    res=malloc(i*50*sizeof(char));
    strcpy(res,temp);
    strcat(res,tab);
    i++;
  }
  printf("Resultat %s\n",res);
  free(appel);
  free(res);
}
```

III) IPv6

Exercice 3 :

Reprendre l'exercice 2 du TP6 mais cette fois-ci l'entité 1 reçoit de l'entité 2 une adresse IPv6.