Support du cours UE1-04 N°1
Création de sockets

1 Fonctions de base

1.1 Adresse générique d'une machine

```
struct in_addr{
u_long s_addr;
};
```

1.2 Caractéristiques d'une machine

```
struct  hostent {
        char    *h_name;        /* Nom officiel de la machine */
        char    **h_aliases;    /* liste d'alias */
        int     h_addrtype;     /* type d'adresse : AF_INET */
        int     h_length;       /* longueur de l'adresse */
        char    **h_addr_list;  /* liste d'adresses de la machine */
#define h_addr  h_addr_list[0]  /* la 1ere adresse de la liste */
};
```

Comme on le voit, une adresse apparaît comme une suite de `h_length` octets.
Il est commode pour recopier une adresse d'utiliser la fonction de recopie d'octets.

```
void bcopy( const void *s1,   /* chaine d'origine */
            void *s2,         /* chaine destination */
            size_t n);        /* nombre d'octets a recopier */
```

La fonction `bzero` permet d'initialiser à 0 une zone mémoire d'adresse et de taille données.

```
void bzero( void *s,        /* adresse de la zone a initialiser */
            size_t n);      /* nombre d'octets a initialiser    */
```

1.3 Capture des caractéristiques d'une machine

```
struct hostent *gethostbyname(const char *name);
```

1.4 Capture du nom d'une machine locale

```
int gethostname( char *name,    /* Nom du systeme local */
                 int namelen); /* longueur espace reserve au nom */
```

1.5 Problème de la représentation des entiers

Comme les machines échangent des adresses ou des numéros de ports, il est prudent de mettre ceux-ci sous leur forme "réseau". En effet, certaines machines utilisent la notation *litte-endian* (bit de poids forts à droite) et d'autres la notation *big-endian* (bits de poids forts à gauche) pour représenter leurs données, leurs types, leurs adresses...Les fonctions suivantes permettent le passage de la représentation locale à la représentation réseau.

```
/* entier long locale --> reseau */
uint32_t htonl(unint32_t hostlong);
/* entier court locale --> reseau */
uint16_t htons(uint16_t hostshort);
/* entier long reseau --> locale */
uint32_t ntohl(uint32_t netlong);
/* entier court reseau --> locale */
uint16_t ntohs(uint16_t netshort);
```

2 Fonctions sur les sockets

2.1 Domaine d'une socket

Structure générique :

```
struct sockaddr {
  sa_family_t       sa_family;      /* famille de l'adresse */
  char              sa_data[14];    /* filler de 14 octets */
  };
```

Pour une utilisation particulière, on remplacera cette structure par une structure correspondant au domaine de communication utilisé.

2.1.1 Domaine UNIX

```
struct  sockaddr_un {
        sa_family_t     sun_family;       /* domaine UNIX : AF_UNIX */
        char            sun_path[108];    /* reference UNIX */
};
```

2.1.2 Domaine INTERNET

```
struct sockaddr_in {
  sa_family_t    sin_family;  /* la famille de l'adresse : AF_INET */
```

```
  in_port_t      sin_port;          /* le numero de port */
  struct         in_addr sin_addr; /* l'adresse internet */
  char           sin_zero[8];       /* champ de 8 zeros */
};
```

2.2 Création/suppression d'une socket

- Création :

```
int socket( int domain,     /* AF_INET, AF_UNIX ...*/
            int type,       /* SOCK_DGRAM, SOCK_STREAM ... */
            int protocol); /* 0 : protocol par defaut */
```

- Suppression :

```
int close(int socketdes);
```

2.3 Attachement d'une socket à une adresse

```
int bind( int sock,  /* descripteur de la socket */
          struct sockaddr *p_adresse,  /* pointeur sur l'adresse */
          socklen_t *lg); /* longueur de l'adresse */
```

2.3.1 Exemple dans le domaine UNIX

```
/*
 * Commande de creation et de "nommage"
 * d'une socket dans le domaine UNIX
 * Le nom de la socket est donnee en parametre
 */

#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <sys/un.h>

int
main(int argc, char * argv[])
{
     int sock ; /* descripteur de la socket */
     struct sockaddr_un adr ; /* adresse  UNIX de la socket  */

     if( argc != 2 )
     {
          fprintf( stderr, "Usage : %s <reference fichier socket>\n",
                   argv[0] );
          exit(-1);
     }
```

```
    /* Creation point d'entree */
    if( (sock = socket(AF_UNIX, SOCK_DGRAM, 0 )) == -1 )
    {
        perror("Pb socket");
        exit(-2);
    }

    /* Initialisation de l'adresse */
    adr.sun_family = AF_UNIX ;
    bcopy(argv[1] , adr.sun_path, strlen(argv[1])) ;

    /* Attachement du point d'entree a l'adresse */
    if( bind(sock,(struct  sockaddr *)&adr,sizeof(adr)) == -1 )
    {
        perror("Pb bind");
        exit(-3);
    }
    printf("Attachement reussi\n");

    /* Boucle sans fin pour test */
    while(1);
}
```

```
prompt>sockunix ref_socket &
[2] 13808
Attachement reussi

prompt>sockunix ref_socket &
[3] 13982
Pb bind: Address already in use

prompt>ls -l ref_socket
srwxr-xr-x   1 jacob    parole         0 Dec  6 19:49 ref_socket

prompt>netstat
...
...
...
Active UNIX domain sockets
Address     Type  Vnode       Conn     Local Addr Remote Addr
30001cc5988 dgram 30006f92c38 00000000 ref_socket
...
```

```
...
...

prompt>kill %2

prompt>sockunix ref_socket &
[2] 13983
Pb bind: Address already in use

prompt>rm ref_socket

prompt>sockunix ref_socket &
[2] 13986
Attachement reussi

prompt>kill %2 ; rm ref_socket
```

2.3.2 Exemple dans le domaine INTERNET

Comme informations supplémentaire il faut
- connaitre l'adresse de la machine locale. Pour cela 2 moyens :
  - affecter le champ `sin_addr` avec `gethostname` + `gethostbyname`
  - ou avec la valeur d'adresse symbolique `INADDR_ANY` si la machine a plusieurs adresses (passerelle)
- choisir un n^o de port libre. Pour cela 2 moyens :
  - mettre 0 dans le champ `sin_port` $\rightarrow$ au choix du système
  - en choisir un quelconque mais $\leqslant$ `IPPORT_RESERVED`

Exemple

```
/*
 * Commande de creation et de "nommage" d'une socket
 * choisie par l'utilisateur dans le domaine INTERNET
 * Le parametre est le numero du port souhaite
 */

#include <stdio.h>
#include <stdlib.h>
#include <strings.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
```

```c
#include <netdb.h>
#include <unistd.h>

int
main(int argc, char * argv[])
{
     int sock ;                /* Descripteur de la socket */
     struct sockaddr_in adr ; /* Adresse  INTERNET de la socket  */
     int port = 0 ;            /* Numero de port */
     char nom_machine[126] ;  /* Nom de la machine locale */
     struct hostent  *ptr_machine ;/*Pointeur sur ses caracteristiques*/

     if( argc != 2 )
     {
          fprintf( stderr, "Usage : %s <numero de port souhaite>\n",
                    argv[0] );
          exit(-1);
     }
     sscanf( argv[1] , "%d" , &port );

     /* Creation point d'entree */
     if( (sock = socket(AF_INET, SOCK_DGRAM, 0 )) == -1 )
     {
          perror("Pb socket");
          exit(-2);
     }

     /* Initialisation de l'adresse */
     adr.sin_port = port ;
     adr.sin_family = AF_INET ;
     gethostname(nom_machine,126);
     ptr_machine = gethostbyname(nom_machine);
     bcopy(ptr_machine->h_addr, &adr.sin_addr, ptr_machine->h_length);
     bzero(adr.sin_zero, sizeof(adr.sin_zero) );

     /* Attachement du point d'entree a l'adresse */
     if( bind(sock,(struct  sockaddr *)&adr,sizeof(adr)) == -1 )
     {
          perror("Pb bind");
          exit(-3);
     }

     printf("Attachement sur numero de port %d reussi\n", port);
```

```
    /* Boucle sans fin pour test */
    while(1) ;
    exit(0);
}
```

2.3.3 Récupération de l'adresse d'une socket

```
int getsockname(int  sock,  /* descripteur de la socket */
                struct  sockaddr *p_adr,  /* pointeur sur l'adresse */
                socklen_t *p_lg); /* pointeur longueur de l'adresse */
```

Exemple

```
/*
 * Commande de creation et de "nommage" d'une socket
 * choisie par le systeme dans le domaine INTERNET
 * avec recuperation de l'adresse de la socket
 */

#include <stdio.h>
#include <strings.h>
#include <sys/types.h>
#include <sys/socket.h>
#include <netinet/in.h>
#include <netdb.h>
#include <unistd.h>

int
main(int argc, char * argv[])
{
    int sock ;                  /* Descripteur de la socket */
    struct sockaddr_in adr ;    /* Adresse  INTERNET de la socket  */
    int longueur;               /* Longueur de l'adresse */
    int port = 0 ;              /* Numero de port */

    if( argc != 1 )
    {
        fprintf( stderr, "Usage : %s\n", argv[0] );
        exit(-1);
    }

    /* Creation point d'entree */
    if( (sock = socket(AF_INET, SOCK_DGRAM, 0 )) == -1 )
    {
```

```
        perror("Pb socket");
        exit(-2);
    }

    /* Initialisation de l'adresse */
    adr.sin_port = 0 ;
    adr.sin_family = AF_INET ;
    adr.sin_addr.s_addr= INADDR_ANY;
    bzero( adr.sin_zero , sizeof(adr.sin_zero) ) ;

    /* Attachement du point d'entree a l'adresse */
    if( bind(sock,(struct  sockaddr *)&adr,sizeof(adr)) == -1 )
    {
        perror("Pb bind");
        exit(-3);
    }

    /* Recuperation de l'adresse de la socket */
    longueur = sizeof(adr);
    if(getsockname(sock,
                   (struct sockaddr *)&adr ,
                   (socklen_t *)&longueur))
    {
        perror("Pb getsockname");
        exit(-4);
    }
    port = ntohs(adr.sin_port);
    printf("Numero de port choisi-->%d\n" , port);
    exit(0);
}
```