Travaux Pratiques Réseaux

Septembre 2007

Thème : Programmation réseau

Modèle Client/Serveur, TCP/IP, Sockets Unix.

Exemple d'application client/serveur

Une application *serveur*, lancée sur une machine, attend les requêtes des applications *client*. Lorsque qu'un *client* envoie une requête (ici c'est juste une chaîne de caractères), le *serveur* transforme les deux premiers caractères de la chaîne en "RE" et renvoie la chaîne ainsi modifiée au client. Ce dernier imprime la chaîne reçue à l'écran.

Code du serveur

```
/*-----------------------------------------------
  Serveur à lancer avant le client
-----------------------------------------------*/
#include <stdlib.h>
#include <stdio.h>
#include <linux/types.h>    /* pour les sockets */
#include <sys/socket.h>
#include <netdb.h>          /* pour hostent, servent */
#include <string.h>         /* pour bcopy, ... */

#define MAXNOM 256
/*-----------------------------------------------*/
/* Traitement effectué sur la machine 'serveur' */
renvoi (soc)
int soc;
{
  char buf[256];
  int lon;
  if ((lon = read(soc, buf, sizeof(buf))) <=0) return;
  buf[lon] = '#';
  buf[lon+1] ='\0';
  buf[0] = 'R';
  buf[1] = 'E';
  write(soc,buf,strlen(buf)+1);
  return;
}
/*-----------------------------------------------------*/
main(argc,argv)
int argc;
char **argv;
```

```
{
int sd,       /* descripteur de socket                  */
     nsd,     /* [nouveau] descripteur de socket        */
     ladcour; /* longueur d'adresse courante d'un client */

struct sockaddr_in adsock,  /* structure d'adresse locale*/
      adclcour;             /* adresse client courant    */
struct hostent *hptr;       /* les infos recuperees sur la machine hote          */
struct servent *sptr;       /* les infos recuperees sur le service de la machine */
char machine[MAXNOM+1];     /* nom de la machine locale  */
char *prog;                 /* nom du programme          */

prog=argv[0];               /* recuperation du nom du programme  */
gethostname(machine,MAXNOM);/* recuperation du nom de la machine */

/* recuperation de la structure d'adresse en utilisant le nom    */
if ((hptr=gethostbyname(machine))==NULL)
{
 perror(">> Machine inconnue");
 exit(1);
}

/* initialisation de la structure adsock avec les infos recuperees */
/* copie de hptr vers adsock */

bcopy((char *)hptr->h_addr,
      (char *)&adsock.sin_addr,
      hptr->h_length);
adsock.sin_family=hptr->h_addrtype;   /* ou bien AF_INET   */
adsock.sin_addr.s_addr=INADDR_ANY;    /* ou bien AF_INET   */

/* il reste à indiquer le numero de service pour adsock    */

/*----- Deux façons d'indiquer un numero de service        */
/* Faire le choix en commentant les lignes inutiles        */

/*---------------------------------------------------------*/
/***SOIT utiliser celui d'un service existant par ex. "irc"*/
/* recuperation des infos sur le service "irc"             */
if ((sptr=getservbyname("irc","tcp"))==NULL)
{
 perror(">> Probleme de recuperation des infos sur le service");
 exit(1);
}

/* recup. du numero de port de "irc" pour initialiser adsock */
adsock.sin_port= htons(sptr->s_port);        /* SOLUTION 1   */
/*---------------------------------------------------------*/
/***SOIT en définir un comme celui du service en définition  */
adsock.sin_port= htons(5000);                /*  SOLUTION 2  */
```

```
/* creation d'un socket */
if ((sd=socket(AF_INET,SOCK_STREAM,0))<0)
{
 perror(">> Probleme de creation du socket");
 exit(1);
}

/* association du socket sd à la structure d'adresse adsock  */
if ((bind(sd,&adsock, sizeof(adsock)))<0)
{
 perror(">> Probleme avec le binding");
 exit(1);
}
printf("%d = Num-Port\n", ntohs(adsock.sin_port)); /*juste une trace */

/* initialisation de la queue d'ecoute */
listen(sd,5);

/* attente des connexions et traitement */
for(;;)
{
 ladcour=sizeof(adclcour);
 /* adclcour sera renseigné par accept via les infos du connect */
 if ((nsd = accept(sd, &adclcour, &ladcour)) <0)
 {
  perror(">> Probleme sur l'accept ");
  exit(1);
 }
 renvoi(nsd); /*traitement*/
 close(nsd);
 }
} /* du main */
```

Code du client

```
/*-------------------------------------------------------------
   Code du Client à lancer apres le serveur
   lancer avec -->  serveur <idMachineSerevur> <chaine>
-------------------------------------------------------------*/
#include <stdlib.h>
#include <stdio.h>
#include <linux/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <string.h>

int main(argc,argv)
int argc;
char **argv;
{
int sd,                     /* descripteur de socket         */
    lon;                    /* longueur d'un buffeur utilisé */
struct sockaddr_in adsock;  /* adresse de socket local       */
struct hostent *hptr;       /* info sur une machine hote     */
struct servent *sptr;       /* info sur service              */

char buf[256];
char *prog;                 /* nom du programme              */
char *host;                 /* nom de la machine distante    */
char *mesg;                 /* message envoyé                */

prog=argv[0];
printf("Code client> %s\n",prog);

if (argc != 3)
   {
       perror(">> Il faut deux arguments au programme");
       exit(1);
   }
host = argv[1];
mesg = argv[2];

printf("Code client--> machine= %s\n",host);
printf("Code client--> message= %s\n",mesg);

if ((hptr = gethostbyname(host)) == NULL)
   {
   perror(">>Pb pour la recup des infos du host");
   exit(1);
   }

/* copie car/car des infos de hptr vers adsock */
bcopy((char *)hptr->h_addr,
      (char *)&adsock.sin_addr,
       hptr->h_length);
adsock.sin_family = AF_INET;         /*  ou hptr->h_addrtype;  */
```

```
/* 2 facons de definir le service qu'on va utiliser à distance */

/* SOLUTION 1 : numéro d'un service banalisé, par ex. "irc"    */
if ((sptr = getservbyname("irc","tcp")) == NULL)
   {
   perror(">>Pb pour la recup des infos sur le service");
   exit(1);
   }
adsock.sin_port = htons(sptr->s_port);

/* SOLUTION 2 :  un nouveau numéro connu pour le service usité */
adsock.sin_port = htons(5000);

printf("Code client--> numport = %d\n",ntohs(sptr->s_port));

if ((sd = socket(AF_INET, SOCK_STREAM, 0))<0)
   {
   perror(">>Pb de creation de la socket");
   exit(1);
   }

/* tentative de connexion au serveur  dont les infos sont dans adsock */
if ((connect(sd, &adsock, sizeof(adsock)))<0)
   {
   perror(">>Pb de connexion");
   exit(1);
   }

if ((write(sd, mesg, strlen(mesg)))<0)
   {
   perror(">>Pb sur le write");
   exit(1);
   }

printf("Code client --> Affichage apres reception ...\n");
while((lon = read(sd, buf, sizeof(buf))) >0)
    write(1,buf,lon);
printf("\nCode client --> c'est fini !!\n");
close(sd);
exit(0);
}
/*-------------------------------------fin-------------------------*/
```