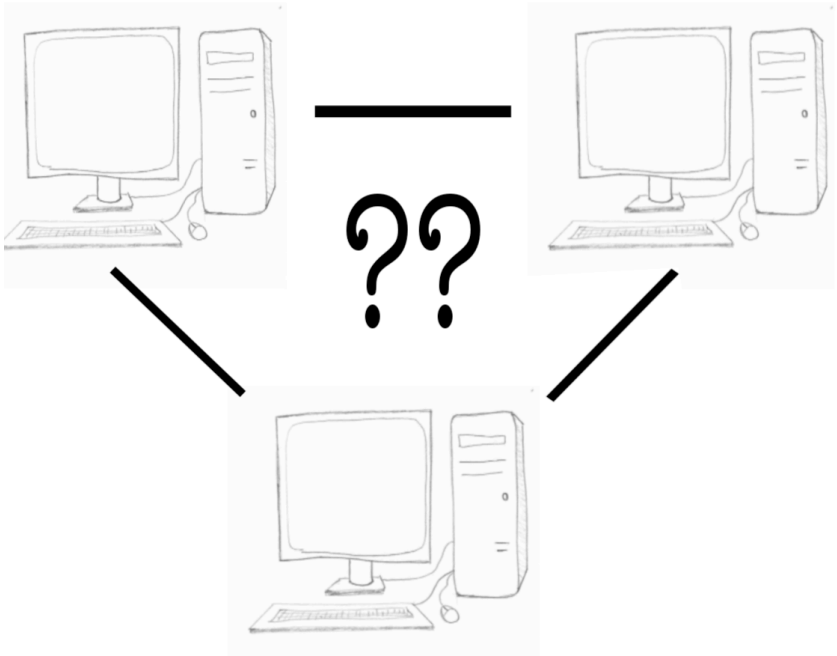
PROGRAMMATION RÉSEAU

Arnaud Sangnier
sangnier@irif.fr

UDP - Mode par paquet - II



La communication UDP en C

- Pour le C, il faut changer le type des sockets quand on appelle socket
  - **SOCK_STREAM** : socket TCP
  - **SOCK_DGRAM** : socket UDP

```
int sock=socket(PF_INET,SOCK_DGRAM,0);
```

- On n'a plus besoin d'utiliser **connect** (vu qu'on n'est pas en mode connecté)
- Si on veut écouter sur une socket, il faudra bien faire le **bind**
- On n'aura plus de **listen** ou d'**accept**
- On utilisera **sendto**, **recv**, **recvfrom** pour l'envoi et la réception des paquets

Envoi en UDP

- Comme en Java, on va préciser le destinataire au moment de l'envoi du paquet
- On va utiliser la fonction suivante :
  - **ssize_t sendto(**

    int socket, // le numéro de socket

    const void *buffer, // le message à envoyer

    size_t length, // la taille de ce message

    int flags, // pour les options

    const struct sockaddr *dest_addr, //infos destinataire

    socklen_t dest_len); //taille de la struct sockaddr

Envoi en UDP (2)

- On va donc commencer par récupérer les infos du destinaire par exemple avec **getaddrinfo**
- **int getaddrinfo(const char *hostname, const char *servname ,const struct addrinfo *hints, struct addrinfo **res);**
- Rappel :

```
struct addrinfo hints;
bzero(&hints, sizeof(struct addrinfo));
hints.ai_family = AF_INET;
hints.ai_socktype=SOCK_DGRAM;
struct addrinfo *first_info;
int r=getaddrinfo("localhost","5555",&hints,&first_info);
```

- Ici, dans les hints, on précise SOCK_DGRAM car on veut des sockets UDP
- Ensuite le struct sockaddr correspondant est dans
  - first_info->ai_addr; (de type struct sockaddr*)

Exemple d'envoi

```
int main() {
  int sock=socket(PF_INET,SOCK_DGRAM,0);
  struct addrinfo *first_info;
  struct addrinfo hints;
  memset(&hints, 0, sizeof(struct addrinfo));
  hints.ai_family = AF_INET;
  hints.ai_socktype=SOCK_DGRAM;
  int r=getaddrinfo("localhost","5555",&hints,&first_info);
  if(r==0){
    if(first_info!=NULL){
      struct sockaddr *saddr=first_info->ai_addr;
      char tampon[100];
      int i=0;
      for(i=0;i<=10;i++){
        strcpy(tampon,"MESSAGE ");
        char entier[3];
        sprintf(entier,"%d",i);
        strcat(tampon,entier);
        sendto(sock,tampon,strlen(tampon),0,saddr,
                        (socklen_t)sizeof(struct sockaddr_in));
      }
    }
  }
  return 0;
}
```

Pour la réception

- Là encore on doit préciser que l'on utilise des socket UDP
- On fait un bind pour écouter sur le bon port
- **int bind(int sockfd, struct sockaddr *my_addr, int addrlen);**
- Rappel :

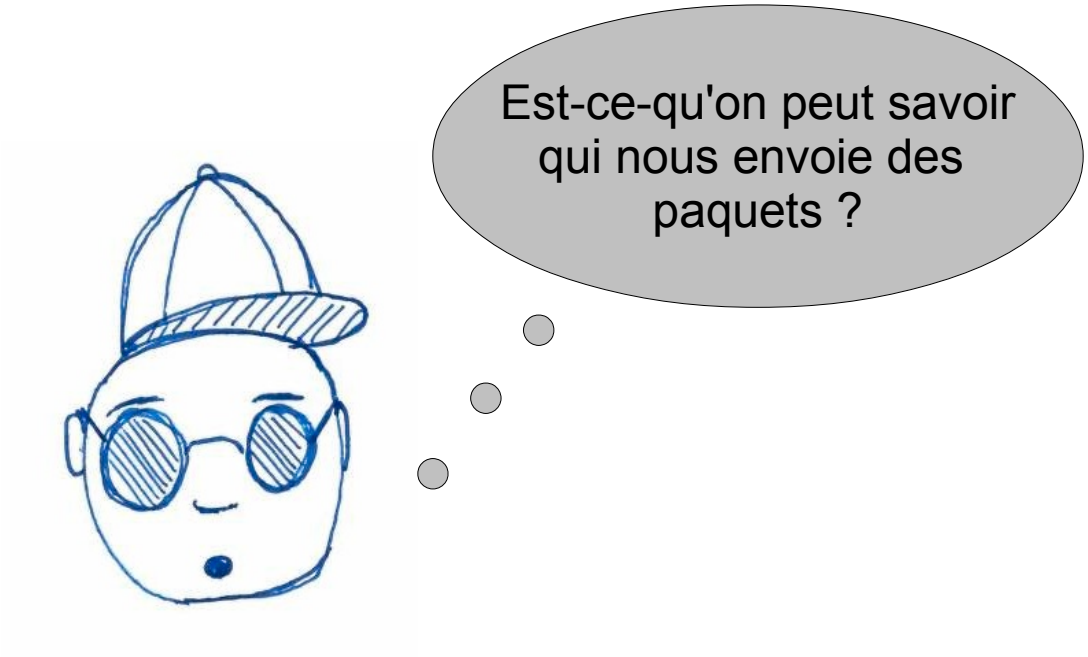
```
sock=socket(PF_INET,SOCK_DGRAM,0);
struct sockaddr_in address_sock;
address_sock.sin_family=AF_INET;
address_sock.sin_port=htons(5555);
address_sock.sin_addr.s_addr=htonl(INADDR_ANY);
int r=bind(sock,(struct sockaddr *)&address_sock,sizeof(struct
sockaddr_in);
```

- On utililse la fonction suivante pour recevoir :
  - **ssize_t recv(int socket, void *buffer, size_t length, int flags);**

Exemple

```
int main() {
  int sock=socket(PF_INET,SOCK_DGRAM,0);
  struct sockaddr_in address_sock;
  address_sock.sin_family=AF_INET;
  address_sock.sin_port=htons(5555);
  address_sock.sin_addr.s_addr=htonl(INADDR_ANY);
  int r=bind(sock,(struct sockaddr *)&address_sock,sizeof(struct
sockaddr_in));
  if(r==0){
    char tampon[100];
    while(1){
      int rec=recv(sock,tampon,100,0);
      tampon[rec]='\0';
      printf("Message recu : %s\n",tampon);
    }
  }
  return 0;
}
```

D'autres informations ?



- OUI !!!!
- Avec la méthode :
  - **ssize_t recvfrom(int socket, void *restrict buffer, size_t length, int flags, struct sockaddr *restrict address, socklen_t *restrict address_len);**

Exemple

```
int main() {
  int sock=socket(PF_INET,SOCK_DGRAM,0);
  sock=socket(PF_INET,SOCK_DGRAM,0);
  struct sockaddr_in address_sock;
  address_sock.sin_family=AF_INET;
  address_sock.sin_port=htons(5555);
  address_sock.sin_addr.s_addr=htonl(INADDR_ANY);
  int r=bind(sock,(struct sockaddr *)&address_sock,sizeof(struct
                                                    sockaddr_in));
  struct sockaddr_in emet;
  socklen_t a=sizeof(emet);
  if(r==0){
    char tampon[100];
    while(1){
      int rec=recvfrom(sock,tampon,100,0,(struct sockaddr *)&emet,&a);
      tampon[rec]='\0';
      printf("Message recu : %s\n",tampon);
      printf("Port de l'emetteur: %d\n",ntohs(emet.sin_port));
      printf("Adresse de l'emetteur: %s\n",inet_ntoa(emet.sin_addr));
    }
  }
  return 0;
}
```