Types de diffusion

* Deux façon de faire de la diffusion :
  + **Diffusion intégrale** (**broadcast**)
    - La diffusion s'effectue en direction de toutes les machines d'un réseau donné
    - Diffusion d'un message à tout le monde
    - Parallèle possible : sirène d'incendie
  + **Multi-diffusion** (**multicast**)
    - La diffusion s'effectue en direction d'un groupe de machines qui se sont **abonnées**
    - Seuls les abonnés reçoivent donc les messages
    - Parallèle possible : télévision, radio

PR - MULTICAST 4

Le broadcast

* La diffusion intégrale s'effectue en envoyant un paquet sur la « dernière » adresse possible du réseau
* L'alias pour désigner l'adresse de diffusion intégrale pour n'importe quelle réseau est donc l'adresse :
  + **255.255.255.255**
* En envoyant un message là, on envoie en théorie à toutes les machines connectés via Internet ....
* En fait, cet envoi est limité au réseau local

-> *Lorsque l'on fait du broadcast, il n'y a pas de routage*

PR - MULTICAST 7

Broadcast en Java

* En fait, le broadcast en Java se passe comme la communication point à point UDP
* La différence c'est que quand on envoie un paquet, on l'envoie sur l'adresse

**255.255.255.255**

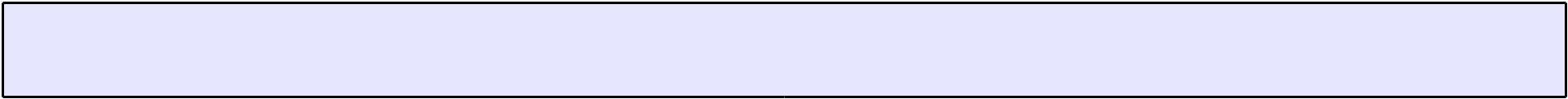
* Avant on envoyait sur l'adresse de la machine qui était connectée
* Pour la réception, on ne change rien

PR - MULTICAST 12

Broadcast en C

* En C, pour l'adresse d'envoi on utilise la macro **INADDR\_BROADCAST**
* Il faut préciser que la socket doit pouvoir envoyer en diffusion intégrale
  + On utilise la fonction setsockopt
* En pratique :

**int ok=1;**



**int r=setsockopt(sock,SOL\_SOCKET,SO\_BROADCAST,&ok,sizeof(ok));**

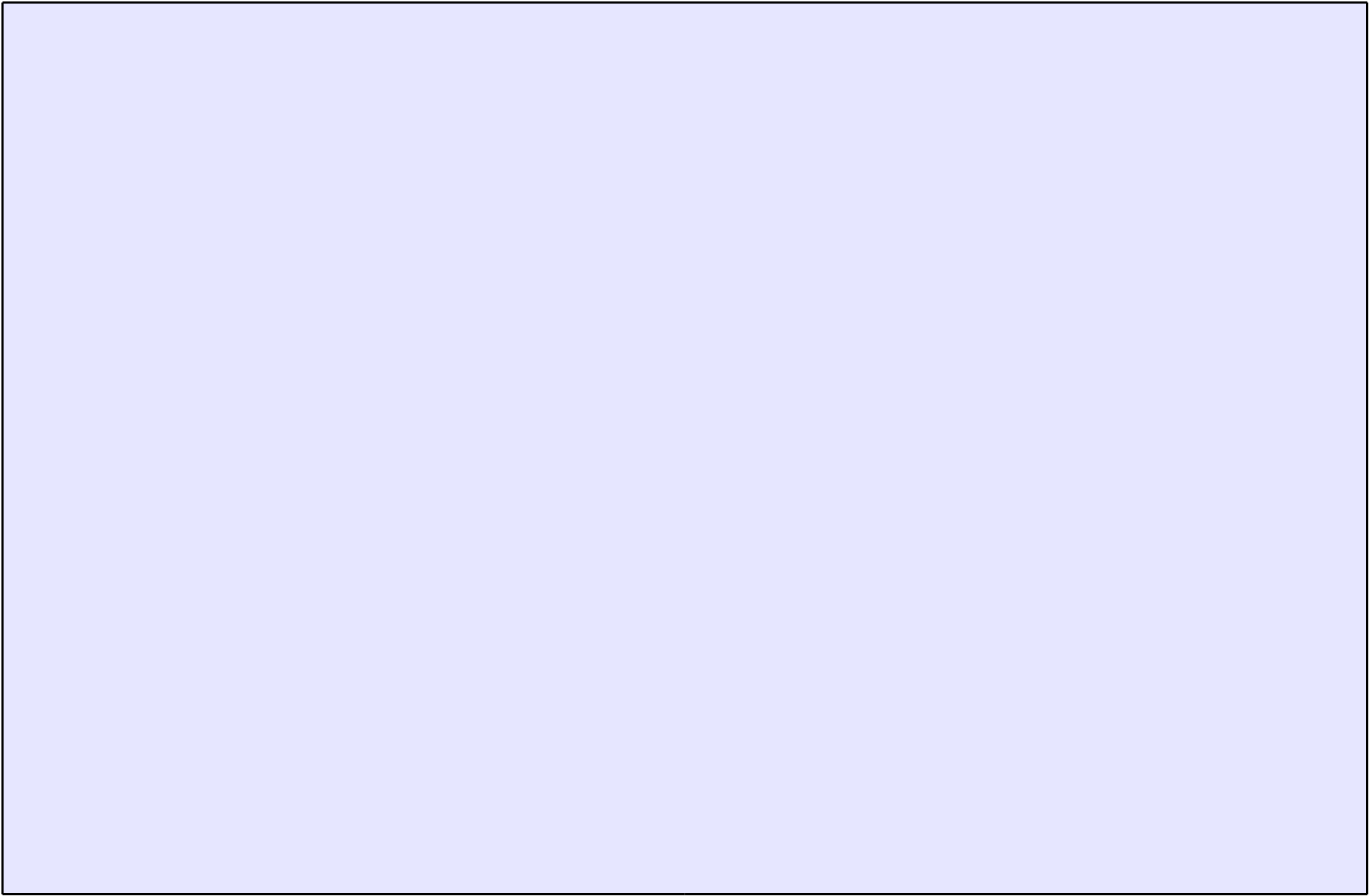
où sock est la socket UDP sur laquelle on souhaite envoyer

* setsockopt renvoie 0 si tout se passe bien
* L'option **SO\_BROADCAST** autorise la socket à émettre en broadcast
* Si on ne fait pas cela, cela ne marche pas

PR - MULTICAST 16

Exemple envoi broadcast

**int main() {**



**int sock=socket(PF\_INET,SOCK\_DGRAM,0);**

**int ok=1;**

**int r=setsockopt(sock,SOL\_SOCKET,SO\_BROADCAST,&ok,sizeof(ok)); if(r==0){**

**struct addrinfo \*first\_info;**

**struct addrinfo hints;**

**memset(&hints, 0, sizeof(struct addrinfo));**

**hints.ai\_family = AF\_INET;**

**hints.ai\_socktype=SOCK\_DGRAM;**

**r=getaddrinfo("255.255.255.255","8888",NULL,&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;**

**}**

17

Exemple réception

**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(8888);**

**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;**

**}**

PR - MULTICAST 18

Le multicast en Java

* Du côté du récepteur
  + Au lieu de prendre une **DatagramSocket**
  + On prendra une classe **MulticastSocket** qui hérite de **DatagramSocket**
  + Constructeur
    - **MulticastSocket(int port)**
    - le port sera le port de multi-diffusion
  + On pourra alors utiliser la méthode
    - **void joinGroup(InetAddress mcastaddr)**
    - elle permet de rejoindre le groupe correspondant à l'adresse de multi-diffusion donné en argument
  + On peut aussi quitter un groupe
    - **void leaveGroup(InetAddress mcastaddr)**
* Pour l'émetteur, comme en UDP

PR - MULTICAST 20

Exemple envoi multicast

**public class EnvoiMulticast {**

**public** **static void main(String[] args){**

**try{**

**DatagramSocket dso=new DatagramSocket(); byte[]data;**

**for(int i=0;i <= 10; i++){**

**String s="MESSAGE "+i+" \n";**

**data=s.getBytes();**

**InetSocketAddress ia=new InetSocketAddress("225.1.2.4",9999);**

**DatagramPacket paquet=new**

**DatagramPacket(data,data.length,ia);**

**dso.send(paquet);**

**}**

**} catch(Exception e){**

**e.printStackTrace();**

**}**

**}**

**}**

PR - MULTICAST 21

Exemple réception multicast

**public class RecoitMulticast {**

**public** **static void main(String[] args){**

**try{**

**MulticastSocket mso=new MulticastSocket(9999); mso.joinGroup(InetAddress.getByName("225.1.2.4");**

**byte[]data=new byte[100];**

**DatagramPacket paquet=new**

**DatagramPacket(data,data.length);**

**while(true){**

**mso.receive(paquet);**

**String st=new**

**String(paquet.getData(),0,paquet.getLength()); System.out.println("J'ai reçu :"+st);**

**}**

**} catch(Exception e){**

**e.printStackTrace();**

**}**

**}**

**}**

PR - MULTICAST 22

Envoi multicast en C



**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("225.1.2.4","9999",**

**NULL,&first\_info);**

**if(r==0){**

**if(first\_info!=NULL){**

**struct sockaddr \*saddr=first\_info->ai\_addr; char tampon[100];**

**for(int 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;**

**}**

PR - MULTICAST 25

Réception multicast en C

**int main() {**



**int sock=socket(PF\_INET,SOCK\_DGRAM,0);**

**int ok=1;**

**int r=setsockopt(sock,SOL\_SOCKET,SO\_REUSEPORT,&ok,sizeof(ok));**

**struct sockaddr\_in address\_sock;**

**address\_sock.sin\_family=AF\_INET;**

**address\_sock.sin\_port=htons(9999);**

**address\_sock.sin\_addr.s\_addr=htonl(INADDR\_ANY);**

**r=bind(sock,(struct sockaddr \*)&address\_sock,sizeof(struct sockaddr\_in));**

**struct ip\_mreq mreq;**

**mreq.imr\_multiaddr.s\_addr=inet\_addr("225.1.2.4");**

**mreq.imr\_interface.s\_addr=htonl(INADDR\_ANY);**

**r=setsockopt(sock,IPPROTO\_IP,IP\_ADD\_MEMBERSHIP,&mreq,sizeof(mreq));**

**char tampon[100];**

**while(1){**

**int rec=recv(sock,tampon,100,0);**

**tampon[rec]='\0';**

**printf("Message recu: %s\n", tampon);**

**}**

**return 0;}**

PR - MULTICAST 26