1. 多播即使在跨越不同网络的情况下，只要加入多播组就能接收数据。

广播只能向同一网络中的主机传输数据。与多播相同，广播也是基于UDP完成的。根据传输数据时使用的IP地址形式，广播分成2种：直接广播和本地广播

直接广播的IP地址中除了网络地址外，其余主机地址全部设置成1。如向192.12.34中的所有主机传送数据，使用地址为192.12.34.255.

本地广播使用的IP地址限定为255.255.255.255

2. 默认生成的套接字阻止广播，需要更改设置。数据通讯中使用的IP地址是与UDP示例的唯一区别。

1. **int** bcast = 1;
3. setsockopt(send\_sock,SOL\_SOCKET,SO\_BROADCAST,(**void** \*)&bcast, **sizeof**(bcast));

3. 示例：

sender.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <arpa/inet.h>
6. #include <sys/socket.h>
8. #define BUF\_SIZE    30
10. **void** error\_handling(**char** \*message);
12. **int** main(**int** argc, **char** \*argv[]){
13. **int** send\_sock;
14. **FILE** \*fp;
15. **char** buf[BUF\_SIZE];
16. **struct** sockaddr\_in broad\_adr;
17. **int** so\_brd = 1;
19. **if**(argc != 3){
20. printf("Usage : %s <group><port>\n",argv[0]);
21. exit(1);
22. }
24. send\_sock = socket(PF\_INET,SOCK\_DGRAM,0);
25. **if**(send\_sock == -1){
26. error\_handling("socket() error");
27. }
29. memset(&broad\_adr,0,**sizeof**(broad\_adr));
30. broad\_adr.sin\_family=AF\_INET;
31. broad\_adr.sin\_addr.s\_addr=inet\_addr(argv[1]);   //ip
32. broad\_adr.sin\_port=htons(atoi(argv[2]));  //port
34. setsockopt(send\_sock,SOL\_SOCKET,SO\_BROADCAST,(**void** \*)&so\_brd,**sizeof**(so\_brd));
35. **if**((fp = fopen("news.txt","r")) == NULL){
36. error\_handling("fopen");
37. }
39. **while**(!feof(fp)){
40. fgets(buf,BUF\_SIZE,fp);
41. sendto(send\_sock,buf,strlen(buf),0,(**struct** sockaddr \*)&broad\_adr,**sizeof**(broad\_adr));
42. sleep(2);
43. }
44. fclose(fp);
45. close(send\_sock);
47. **return** 0;
48. }
50. **void** error\_handling(**char** \*message){
52. fputs(message,stderr);
53. fputs("\n",stderr);
54. exit(1);
55. }

receiver.c

1. #include <stdio.h>
2. #include <stdlib.h>
3. #include <string.h>
4. #include <unistd.h>
5. #include <arpa/inet.h>
6. #include <sys/socket.h>
8. #define BUF\_SIZE    30
10. **void** error\_handling(**char** \*message);
12. **int** main(**int** argc, **char** \*argv[]){
13. **int** recv\_sock;
14. **int** str\_len;
15. **char** buf[BUF\_SIZE];
16. **struct** sockaddr\_in adr;
18. **if**(argc != 2){
19. printf("Usage : %s <port>\n",argv[0]);
20. exit(1);
21. }
23. recv\_sock = socket(PF\_INET,SOCK\_DGRAM,0);
24. **if**(recv\_sock == -1){
25. error\_handling("socket() error");
26. }
28. memset(&adr,0,**sizeof**(adr));
29. adr.sin\_family=AF\_INET;
30. adr.sin\_addr.s\_addr=htonl(INADDR\_ANY);  // ip
31. adr.sin\_port=htons(atoi(argv[1]));  //port
33. **if**(bind(recv\_sock,(**struct** sockaddr \*)&adr,**sizeof**(adr))==-1){
34. error\_handling("bind error");
35. }
37. **while**(1){
38. str\_len = recvfrom(recv\_sock,buf,BUF\_SIZE-1,0,NULL,0);
39. **if**(str\_len < 0){
40. **break**;
41. }
42. buf[str\_len] = 0;
43. fputs(buf,stdout);
44. }
45. close(recv\_sock);
47. **return** 0;
48. }
50. **void** error\_handling(**char** \*message){
52. fputs(message,stderr);
53. fputs("\n",stderr);
54. exit(1);
55. }