<https://blog.csdn.net/xjbclz/article/details/51834728>

1 recv和send   
recv和send函数提供了和read和write差不多的功能.不过它们提供 了第四个参数来控制读写操作.

int recv(int sockfd,void \*buf,int len,int flags)   
int send(int sockfd,void \*buf,int len,int flags)

前面的三个参数和read,write一样,第四个参数可以是0或者是以下的组合   
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_   
| MSG\_DONTROUTE | 不查找路由表 |   
| MSG\_OOB | 接受或者发送带外数据 |   
| MSG\_PEEK | 查看数据,并不从系统缓冲区移走数据 |   
| MSG\_WAITALL | 等待所有数据 |   
|————————————————————–|

MSG\_DONTROUTE:是send函数使用的标志.这个标志告诉IP协议.目的主机在本地网络上面,没有必要查找路由表.这个标志一般用网络诊断和路由程序里面.   
MSG\_OOB:表示可以接收和发送带外的数据.关于带外数据我们以后会解释的.

MSG\_PEEK:是recv函数的使用标志,表示只是从系统缓冲区中读取内容,而不清楚系统缓冲区的内容.这样下次读的时候,仍然是一样的内容.一般在有多个进程读写数据时可以使用这个标志.

MSG\_WAITALL是recv函数的使用标志,表示等到所有的信息到达时才返回.使用这个标志的时候recv回一直阻塞,直到指定的条件满足, 或者是发生了错误. 1)当读到了指定的字节时,函数正常返回.返回值等于len 2)当读到了文件的结尾时,函数正常返回.返回值小于len 3)当操作发生错误时,返回-1,且设置错误为相应的错误号(errno)

如果flags为0,则和read,write一样的操作.还有其它的几个选项,不过我们实际上用的很少,可以查看 Linux Programmer’s Manual得到详细解释.

2 recvfrom和sendto   
这两个函数一般用在非套接字的网络程序当中(UDP),我们已经在前面学会了.

3 recvmsg和sendmsg   
recvmsg和sendmsg可以实现前面所有的读写函数的功能.

int recvmsg(int sockfd,struct msghdr \*msg,int flags)   
int sendmsg(int sockfd,struct msghdr \*msg,int flags)

struct msghdr   
{   
void \*msg\_name;   
int msg\_namelen;   
struct iovec \*msg\_iov;   
int msg\_iovlen;   
void \*msg\_control;   
int msg\_controllen;   
int msg\_flags;   
}

struct iovec   
{   
void \*iov\_base; /\* 缓冲区开始的地址 \*/   
size\_t iov\_len; /\* 缓冲区的长度 \*/   
}

msg\_name和 msg\_namelen当套接字是非面向连接时(UDP),它们存储接收和发送方的地址信息.msg\_name实际上是一个指向struct sockaddr的指针,msg\_name是结构的长度.当套接字是面向连接时,这两个值应设为NULL. msg\_iov和msg\_iovlen指出接受和发送的缓冲区内容.msg\_iov是一个结构指针,msg\_iovlen指出这个结构数组的大小. msg\_control和msg\_controllen这两个变量是用来接收和发送控制数据时的 msg\_flags指定接受和发送的操作选项.和recv,send的选项一样 

4 套接字的关闭   
关闭套接字有两个函数close和shutdown.用close时和我们关闭文件一样.

5 shutdown

int shutdown(int sockfd,int howto)

TCP连接是双向的(是可读写的),当我们使用close时,会把读写通道都关闭,有时侯我们希望只关闭一个方向,这个时候我们可以使用shutdown.针对不同的howto,系统回采取不同的关闭方式.   
howto=0这个时候系统会关闭读通道.但是可以继续往接字描述符写.

howto=1关闭写通道,和上面相反,着时候就只可以读了.

howto=2关闭读写通道,和close一样 在多进程程序里面,如果有几个子进程共享一个套接字时,如果我们使用shutdown, 那么所有的子进程都不能够操作了,这个时候我们只能够使用close来关闭子进程的套接字描述符.