Basic knowledge

**Unblock.cpp**

Fctl函数(file control)，提供对文件描述符的各种控制操作。

函数原型：

#include<fctl.h>

Int fcntl(int fd, int cmd, ..)

Fd是被操作文件描述符，cmd是执行何种类型的操作。

inet\_pton函数提供ip地址转换函数。

**linux下：**

#include <sys/socket.h>

#include <netinet/in.h>

#include<arpa/inet.h>

注：我写的函数原型是#include<arpa/inet.h>下的，clang编译器。

int inet\_pton(int af, const char \*src, void \*dst);这个是百度百科的。

int inet\_pton (int \_\_af, const char \*\_\_restrict \_\_cp,

            void \*\_\_restrict \_\_buf)

参数列表：

af是表示转换成哪种地址簇，cp表示要转换的ip，buf表示cp参数转换后的数据。

uint16\_t htons(uint16\_t hostshort);  
　　htons的功能：将一个无符号短整型数值转换为网络字节序，即大端模式(big-endian)  
　　参数u\_short hostshort: 16位无符号整数  
　　返回值:TCP / IP网络字节顺序  
　　htons 是把你机器上的整数转换成“网络字节序”， 网络字节序是 big-endian，也就是整数的高位字节存放在内存的低地址处。 而我们常用的 x86 CPU (intel, AMD) 电脑是 little-endian,也就是整数的低位字节放在内存的低字节处。举个例子吧。假定你的port是0x1234,在网络字节序里 这个port放到内存中就应该显示成addr addr+1，也就是：0x12 0x34；而在x86电脑上，0x1234放到内存中实际是：addr addr+1，也就是：0x34 0x12。htons 的用处就是把实际内存中的整数存放方式调整成“网络字节序”的方式。

第一个问题：为什么使用两个字节，也就是16位来存储。

　　这个简单一些，因为一个字节只能存储8位2进制数，而计算机的端口数量是65536个，也就是2^16，两个字节。

第二个为题：为什么计算机需要大端模式和小端模式？

小端模式 ：强制转换数据不需要调整字节内容，1、2、4字节的存储方式一样。  
　　大端模式 ：符号位的判定固定为第一个字节，容易判断正负。

来源博客：[详解C语言的htons函数 - 李秋 - 博客园 (cnblogs.com)](https://www.cnblogs.com/liqiu/archive/2013/03/26/2982406.html)

#include<sys/types.h>

#include<sys/socket.h>

socket函数原型

Domin参数告诉系统使用哪个底层协议簇。对于tcp/ip协议簇而言，该参数应该设置为PF\_INET（用于ipv4）,或PF\_INET6（ipv6）；对于UNIX,应设置为PF\_UNIX.

Type参数指定服务类型。服务类型主要有SOCK\_STREAM服务（流服务）和SOCK\_UGRAM(数据报服务)。前者为tcp协议在传输层中使用，后者为udp协议在传输层中使用。

Protocol参数是在前面两个参数构成的协议集合下再具体选择一个协议。默认情况下，一般为0.

Int socket (int domin, int type, int protocol)

Select函数

select函数用于在非阻塞中，当一个套接字或一组套接字有信号时通知你，系统提供select函数来实现多路复用输入/输出模型，原型：  
int select(int maxfd,fd\_set \*rdset,fd\_set \*wrset,fd\_set \*exset,struct timeval \*timeout);

   所在的头文件为：#include <sys/time.h> 和#include <unistd.h>

 对于fd\_set类型通过下面四个宏来操作：  
    FD\_ZERO(fd\_set \*fdset) 将指定的文件描述符集清空，在对文件描述符集合进行设置前，必须对其进行初始化，如果不清空，由于在系统分配内存空间后，通常并不作清空处理，所以结果是不可知的。  
    FD\_SET(fd\_set \*fdset) 用于在文件描述符集合中增加一个新的文件描述符。  
    FD\_CLR(fd\_set \*fdset) 用于在文件描述符集合中删除一个文件描述符。  
    FD\_ISSET(int fd,fd\_set \*fdset) 用于测试指定的文件描述符是否在该集合中。

**Client.cpp**

#include<sys/types.h>

#include<sys/socket.h>

Int connect(int sockfd, const struct sockaddr\* serv\_addr,

socklen\_t addrlen);

**sockfd参数由socket系统调用返回一个socket。Serv\_addr参数是服务器监听的socket地址，addrlen参数则指定这个地址的长度。**

**函数执行成功就返回0,失败返回-1并设置errno，其中errno有两种。**

**ECONNREFUSED和ETIMEDOUT,前者是目标端口不存在，连接被拒绝。后者是连接超时。**