socket常用API详解

一、socket概述

socket是在应用层和传输层之间的一个抽象层，它把TCP/IP层复杂的操作抽象为几个简单的接口供应用层调用已实现进程在网络中通信。

socket起源于UNIX，在Unix一切皆文件哲学的思想下，socket是一种"打开—读/写—关闭"模式的实现，服务器和客户端各自维护一个"文件"，在建立连接打开后，可以向自己文件写入内容供对方读取或者读取对方内容，通讯结束时关闭文件。

二、接口详解

socket()：创建socket

bind()：绑定socket到本地地址和端口，通常由服务端调用

listen()：TCP专用，开启监听模式

accept()：TCP专用，服务器等待客户端连接，一般是阻塞态

connect()：TCP专用，客户端主动连接服务器

send()：TCP专用，发送数据

recv()：TCP专用，接收数据

sendto()：UDP专用，发送数据到指定的IP地址和端口

recvfrom()：UDP专用，接收数据，返回数据远端的IP地址和端口

closesocket()：关闭socket

2.1 socket()

原型：int socket (int domain, int type, int protocol)

功能描述：初始化创建socket对象，通常是第一个调用的socket函数。 成功时，返回非负数的socket描述符；失败是返回-1。socket描述符是一个指向内部数据结构的指针，它指向描述符表入口。调用socket()函数时，socket执行体将建立一个socket，实际上"建立一个socket"意味着为一个socket数据结构分配存储空间。socket执行体为你管理描述符表。

参数解释：

domain

domain - 指明使用的协议族。常用的协议族有，AF_INET、AF_INET6、AF_LOCAL（或称AF_UNIX，Unix域socket）、AF_ROUTE等等。协议族决定了socket的地址类型，在通信中必须采用对应的地址，如AF_INET决定了要用ipv4地址（32位的）与端口号（16位的）的组合、AF_UNIX决定了要用一个绝对路径名作为地址。

type

指明socket类型，有3种：

```
SOCK_STREAM -- TCP类型，保证数据顺序及可靠性；
SOCK_DGRAM -- UDP类型，不保证数据接收的顺序，非可靠连接；
SOCK_RAW -- 原始类型，允许对底层协议如IP或ICMP进行直接访问，不太常用。
```

protocol

通常赋值"0"，由系统自动选择。

2.2 bind()

原型：int bind(int sockfd, const struct sockaddr* myaddr, socklen_t addrlen)

功能描述：将创建的socket绑定到指定的IP地址和端口上，通常是第二个调用的socket接口。返回值：0 - 成功，-1 - 出错。当socket函数返回一个描述符时，只是存在于其协议族的空间中，并没有分配一个具体的协议地址（这里指IPv4/IPv6和端口号的组合），bind函数可以将一组固定的地址绑定到sockfd上。

通常服务器在启动的时候都会绑定一个众所周知的协议地址，用于提供服务，客户就可以通过它来接连服务器；而客户端可以指定IP或端口也可以都不指定，未分配则系统自动分配。这就是为什么通常服务器端在listen之前会调用bind()，而客户端就不会调用，而是在connect()时由系统随机生成一个。

注意：

(1) 如果有多个可用的连接（多个IP），内核会根据优先级选择一个IP作为源IP使用。

(2) 如果socket使用bind绑定到特定的IP和port，则无论是TCP还是UDP，都会从指定的IP和port发送数据。

参数解释：

sockfd:
socket()函数返回的描述符;

myaddr

指明要绑定的本地IP和端口号，使用网络字节序，即大端模式（详见3.1）。

addrlen

常被设置为sizeof(struct sockaddr)。

可以利用下边的赋值语句，自动绑定本地IP地址和随机端口：

my_addr.sin_port = 0; /* 系统随机选择一个未被使用的端口号 /
my_addr.sin_addr.s_addr = INADDR_ANY; / 填入本机IP地址 */

另外要注意的是，当调用函数时，一般不要将端口号置为小于1024的值，因为1~1024是保留端口号，你可以使用大于1024中任何一个没有被占用的端口号。

2.3 listen()

原型：int listen(int sockfd, int backlog)

功能描述：listen()函数仅被TCP类型的服务器程序调用，实现监听服务，它实现2件事情：

“1. 当socket()创建1个socket时，被假设为主动式套接字，也就是说它是一个将调用connect()发起连接请求的客户端套接字；函数listen()将套接口转换为被动式套接字，指示内核接受向此套接字的连接请求，调用此系统调用后tcp 状态机由close转换到listen。
2.第2个参数指定了内核为此套接字排队的最大连接个数。”

listen()成功时返回0，错误时返回-1。

参数解释：

sockfd

socket()函数返回的描述符;

backlog

指定内核为此套接字维护的最大连接个数，包括“未完成连接队列 - 未完成3次握手”、“已完成连接队列 - 已完成3次握手，建立连接”。大多数系统缺省值为20。

2.4 accept()

原型： int accept (int sockfd, struct sockaddr *addr, socklen_t *addrlen)

功能描述：accept()函数仅被TCP类型的服务器程序调用，从已完成连接队列返回下一个建立成功的连接，如果已完成连接队列为空，线程进入阻塞态睡眠状态。成功时返回套接字描述符，错误时返回-1。

如果accpet()执行成功，返回由内核自动生成的一个全新socket描述符，用它引用与客户端的TCP连接。通常我们把accept()第一个参数成为监听套接字（listening socket），把accept()功能返回值成为已连接套接字（connected socket）。一个服务器通常只有1个监听套接字，监听客户端的连接请求；服务器内核为每一个客户端的TCP连接维护1个已连接套接字，用它实现数据双向通信。

参数解释：

sockfd

socket()函数返回的描述符;

addr

输出一个的sockaddr_in变量地址，该变量用来存放发起连接请求的客户端的协议地址;

addrten

作为输入时指明缓冲器的长度，作为输出时指明addr的实际长度。

2.5 connetct()

原型： int connect(int sockfd, struct sockaddr *serv_addr, int addrlen)

功能描述：connect()通常由TCP类型客户端调用，用来与服务器建立一个TCP连接，实际是发起3次握手过程，连接成功返回0，连接失败返回1。

注意：

(1) 可以在UDP连接使用使用connect()，作用是在UDP套接字中记住目的地址和目的端口。
(2) UDP套接字使用connect后，如果数据报不是connect中指定的地址和端口，将被丢弃。没有调用connect的UDP套接字，将接收所有到达这个端口的UDP数据报，而不区分源端口和地址。

参数解释：

sockfd

本地客户端额socket描述符;

serv_addr

服务器协议地址;

addrlen
地址缓冲区的长度。

2.6 send()

原型：int send(int sockfd, const void *msg, int len, int flags)

功能描述：TCP类型的数据发送。

每个TCP套接口都有一个发送缓冲区，它的大小可以用SO_SNDBUF这个选项来改变。调用send函数的过程，实际是内核将用户数据拷贝至TCP套接口的发送缓冲区的过程：若len大于发送缓冲区大小，则返回-1；否则，查看缓冲区剩余空间是否容纳得下要发送的len长度，若不够，则拷贝一部分，并返回拷贝长度（指的是非阻塞send，若为阻塞send，则一定等待所有数据拷贝至缓冲区才返回，因此阻塞send返回值必定与len相等）；若缓冲区满，则等待发送，有剩余空间后拷贝至缓冲区；若在拷贝过程出现错误，则返回-1。关于错误的原因，查看errno的值。

如果send在等待协议发送数据时出现网络断开的情况，则会返回-1。注意：send成功返回并不代表对方已接收到数据，如果后续的协议传输过程中出现网络错误，下一个send便会返回-1 发送错误。TCP给对方的数据必须在对方给予确认时，方可删除发送缓冲区的数据。否则，会一直缓存在缓冲区直至发送成功（TCP可靠数据传输决定的）。

参数解释：

sockfd

发送端套接字描述符（非监听描述符）。

msg

待发送数据的缓冲区。

len

待发送数据的字节长度。

flags
一般情况下置为0。

2.7 recv()

原型：int recv(int sockfd, void *buf, int len, unsigned int flags)

功能描述：TCP类型的数据接收。

recv()从接收缓冲区拷贝数据。成功时，返回拷贝的字节数，失败返回-1。阻塞模式下，recv/recvfrom将会阻塞到缓冲区里至少有一个字节(TCP)/至少有一个完整的UDP数据报才返回，没有数据时处于休眠状态。若非阻塞，则立即返回，有数据则返回拷贝的数据大小，否则返回错误-1。

参数解释：

sockefd

接收端套接字描述符（非监听描述符）；

buf

接收缓冲区的基地址；

len

以字节计算的接收缓冲区长度；

flags

一般情况下置为0。

2.8 sendto()

原型：int sendto(int sockfd, const void *msg, int len, unsigned int flags, const struct sockaddr *dst_addr, int addrlen)

功能描述：用于非可靠连接(UDP)的数据发送，因为UDP方式未建立连接socket，因此需要制定目的协议地址。

当本地与不同目的地址通信时，只需指定目的地址，可使用同一个UDP套接口描述符sockfd，而TCP要预先建立连接，每个连接都会产生不同的套接口描述符，体现在：客户端要使用不同的fd进行connect，服务端每次accept产生不同的fd。

因为UDP没有真正的发送缓冲区，因为是不可靠连接，不必保存应用进程的数据拷贝，应用进程中的数据在沿协议栈向下传递时，以某种形式拷贝到内核缓冲区，当数据链路层把数据传出后就把内核缓冲区中数据拷贝删除。因此它不需要一个发送缓冲区。写UDP套接口的sendto/write返回表示应用程序的数据或数据分片已经进入链路层的输出队列，如果输出队列没有足够的空间存放数据，将返回错误ENOBUFS.

参数解释：

sockfd

发送端套接字描述符（非监听描述符）；

msg

待发送数据的缓冲区；

len

待发送数据的字节长度；

flags

一般情况下置为0；

dst_addr

数据发送的目的地址；

addrlen

地址长度。

2.9 recvfrom()

原型：int recvfrom(int sockfd, void *buf, size_t len, int flags, struct sockaddr *src_addr, int*fromlen)

功能描述：用于非可靠连接(UDP)的数据接收。

参数解释：

sockfd

接收端套接字描述；

buf

用于接收数据的应用缓冲区地址；

len

指名缓冲区大小；

flags

通常为0；

src_addr

数据来源端的地址；

fromlen

作为输入时，fromlen常置为sizeof(struct sockaddr)；当输出时，fromlen包含实际存入buf中的数据字节数。