# Linux socket close和shutdown的区别

## 1. close函数

int close**(**int sockfd**);**     //返回成功为0，出错为-1

close 一个套接字的默认行为是把套接字标记为已关闭，然后立即返回到调用进程，该套接字描述符不能再由调用进程使用，也就是说它不能再作为read或write的第一个参数，然而TCP将尝试发送已排队等待发送到对端的任何数据，发送完毕后发生的是正常的TCP连接终止序列。

在多进程并发服务器中，父子进程共享着套接字，套接字描述符引用计数记录着共享着的进程个数，当父进程或某一子进程close掉套接字时，描述符引用计数会相应的减一，当引用计数仍大于零时，这个close调用就不会引发TCP的四路握手断连过程。

## 2.shutdown函数

int shutdown**(**int sockfd**,**int howto**);**  //返回成功为0，出错为-1.

该函数的行为依赖于howto的值

**(**1**)**SHUT\_RD：值为0，关闭连接的读这一半。

**(**2**)**SHUT\_WR：值为1，关闭连接的写这一半。

**(**3**)**SHUT\_RDWR：值为2，连接的读和写都关闭。

终止网络连接的通用方法是调用close函数。但使用shutdown能更好的控制断连过程（使用第二个参数）。

## 3.区别1

close函数会关闭套接字ID，如果有其他的进程共享着这个套接字，那么它仍然是打开的，这个连接仍然可以用来读和写，并且有时候这是非常重要的 ，特别是对于多进程并发服务器来说。

而shutdown会切断进程共享的套接字的所有连接，不管这个套接字的引用计数是否为零，那些试图读的进程将会接收到EOF标识，那些试图写的进程将会检测到SIGPIPE信号，同时可利用shutdown的第二个参数选择断连的方式。

## 4.区别2

从TCP协议角度来看，一个已建立的TCP连接有两种关闭方式，一种是正常关闭，即四次挥手关闭连接；还有一种则是异常关闭，我们通常称之为连接重置（RESET)。

首先说一下正常关闭时四次挥手的状态变迁，关闭连接的主动方状态变迁是FIN\_WAIT\_1->FIN\_WAIT\_2->TIME\_WAIT，而关闭连接的被对方的状态变迁是CLOSE\_WAIT->LAST\_ACK->TIME\_WAIT。在四次挥手过程中ACK包都是协议栈自动完成的，而FIN包则必须由应用层通过closesocket或shutdown主动发送，通常连接正常关闭后，recv会得到返回值0，send会得到错误码10058。

除此之外，在我们的日常应用中，连接异常关闭的情况也很多。比如应用程序被强行关闭、本地网络突然中断(禁用网卡、网线拔出)、程序处理不当等都会导致连接重置，连接重置时将会产生RST包，同时网络络缓冲区中未接收（发送）的数据都将丢失。连接重置后，本方send或recv会得到错误码10053(closesocket时是10038)，对方recv会得到错误码10054，send则得到错误码10053(closesocket时是10054)。

操作系统为我们提供了两个函数来关闭一个TCP连接，分别是closesocket和shutdown。通常情况下，closesocket会向对方发送一个FIN包，但是也有例外。比如有一个工作线程正在调用recv接收数据，此时外部调用closesocket，会导致连接重置，同时向对方发送一个RST包，这个RST包是由本方主动产生的。

shutdown可以用来关闭指定方向的连接，该函数接收两个参数，一个是套接字，另一个是关闭的方向，可用值为SD\_SEND,SD\_RECEIVE和SD\_BOTH。方向取值为SD\_SEND时，无论socket处于什么状态(recv阻塞，或空闲状态)，都会向对方发送一个FIN包，注意这点与closesocket的区别。此时本方进入FIN\_WAIT\_2状态，对方进入CLOSE\_WAIT状态，本方依然可以调用recv接收数据；方向取值为SD\_RECEIVE时，双发连接状态没有改变，依然处于ESTABLISHED状态，本方依然可以send数据，但是，如果对方再调用send方法，连接会被立即重置，同时向对方发送一个RST包，这个RST包是被动产生的，这点注意与closesocket的区别。

从tcp协议上来分析shutdown和closesocket的行为（behavior)：closesocket或shutdown(使用SD\_SEND当作参数时）,会向通信对方发出一个fin包，而此时套接字的状态会由ESTABLISHED变成FIN\_WAIT\_1，然后对方发送一个ACK包作为回应，套接字又变成FIN\_WAIT\_2，如果对方也关闭了连接则对方会发出FIN，我方会回应一个ACK并将套接字置为TIME\_WAIT。因此可以看出closesocket,shutdown所进行的TCP行为是一样的，所不同的是函数部分，shutdown会确保建立的数据传输队列中的数据不被丢失，而closesocket会冒然的抛弃所有的数据，因此如果你愿意closesocket完全可以取代shutdown,然而在数据交互十分复杂的网络协议程序中，最好还是shutdown稳妥一些！