## 1.linux系统编程

### 1.1标准c库和linux系统io函数对比

C语言实现的库函数可以实现跨平台，而Linux系统调用函数只适合本系统。

C语言库函数往往有采用缓存，减少读写磁盘书，效率高。系统调用直接操作磁盘效率低。

网络通信一般系统调用实时直接。 读写磁盘一般C语言库提高效率。

## 2.linux多进程

**ulimit命令：**用来限制系统用户对shell资源的访问。

**子程序和子进程：**

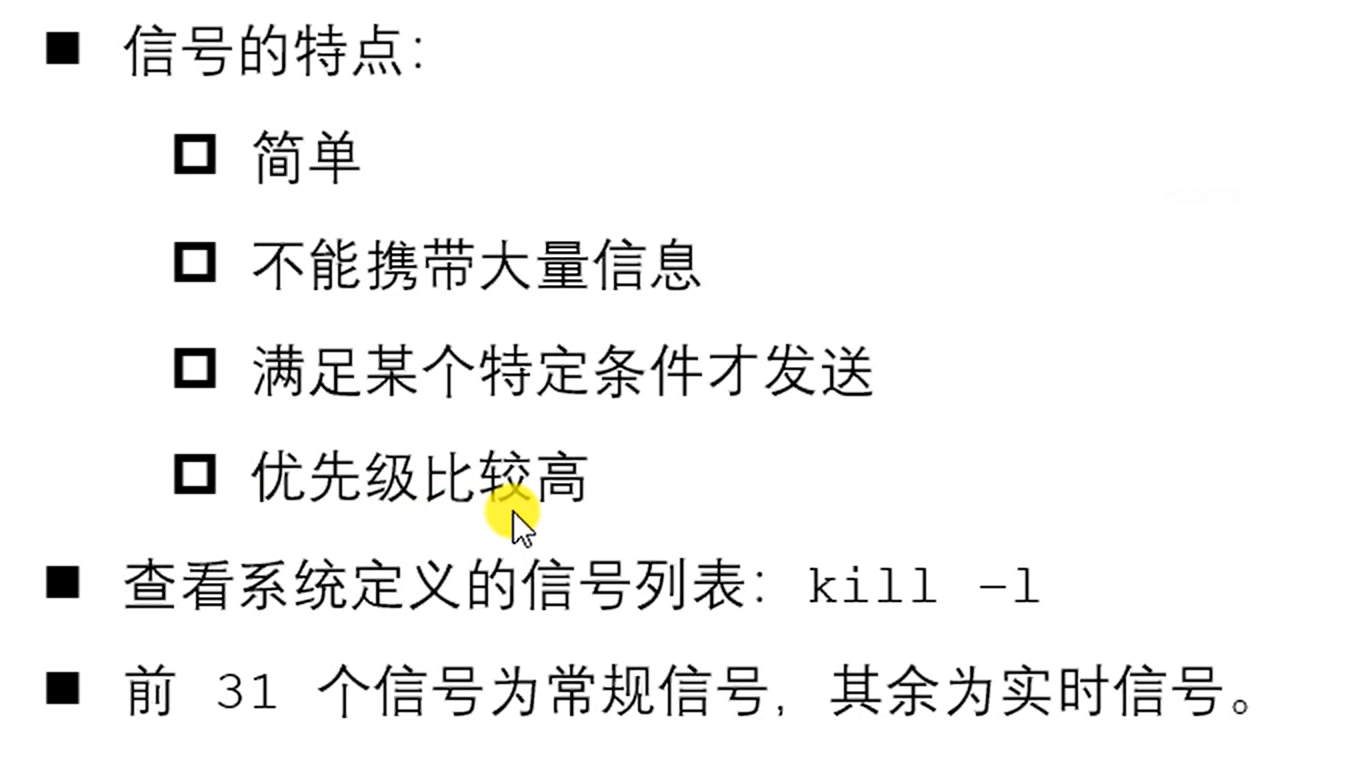
父进程运行的程序，创建了子进程，此时父进程和进程并发执行，是两个进程；主程序调用子程序后，主程序暂停，子程序开始执行，它们还是一个进程。

**Ctrl+C:** 强制 中断程序的执行 ，这个时候程序就 已经释放内存 ，运行结束。

**Ctrl+Z:** 将 程序暂停 ，但并没有释放内存 ，此时在的进程是挂起转态，可以使用 fg 和bg 命令 前台或 后台 继续该任务。

**硬中断**：由与系统相连的外设(比如网卡、硬盘)自动产生的。

**软中断**：它们仅仅是由当前正在运行的进程所产生的。



Ps- aux 查看进程信息

## 3.命令格式

常见执行Linux命令的格式是这样的： 命令名称 [命令参数] [命令对象]

注意，命令名称、命令参数、命令对象之间请用空格键分隔。

命令对象一般是指要处理的文件、目录、用户等资源，而命令参数可以用长格式（完整的选项名称），

也可以用短格式（单个字母的缩写），两者分别用--与-作为前缀

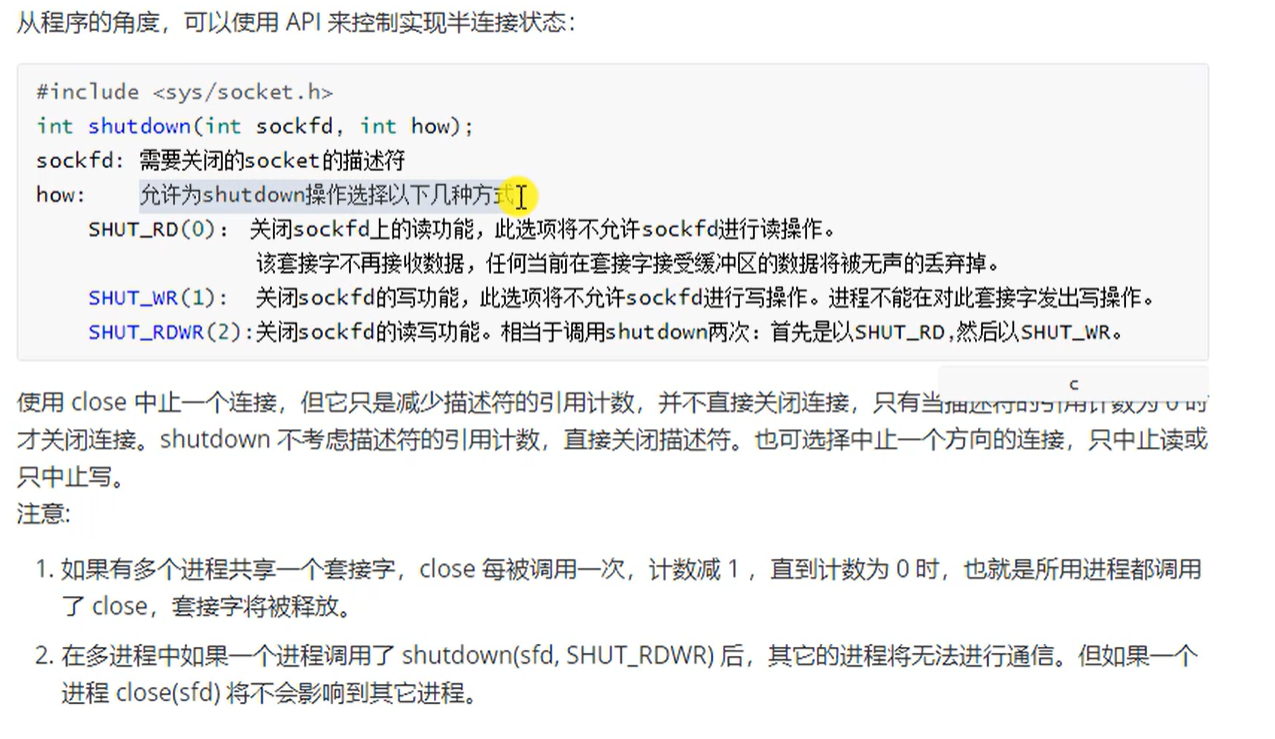
令参数的长格式与短格式示例

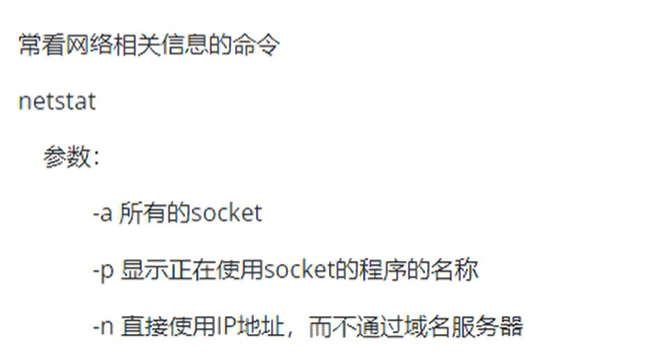
|  |  |
| --- | --- |
| 长格式 | man **--**help |
| 短格式 | man **-**h |

## 4.linux socket通信

### 4.24半关闭

Close关闭是读写都关闭。





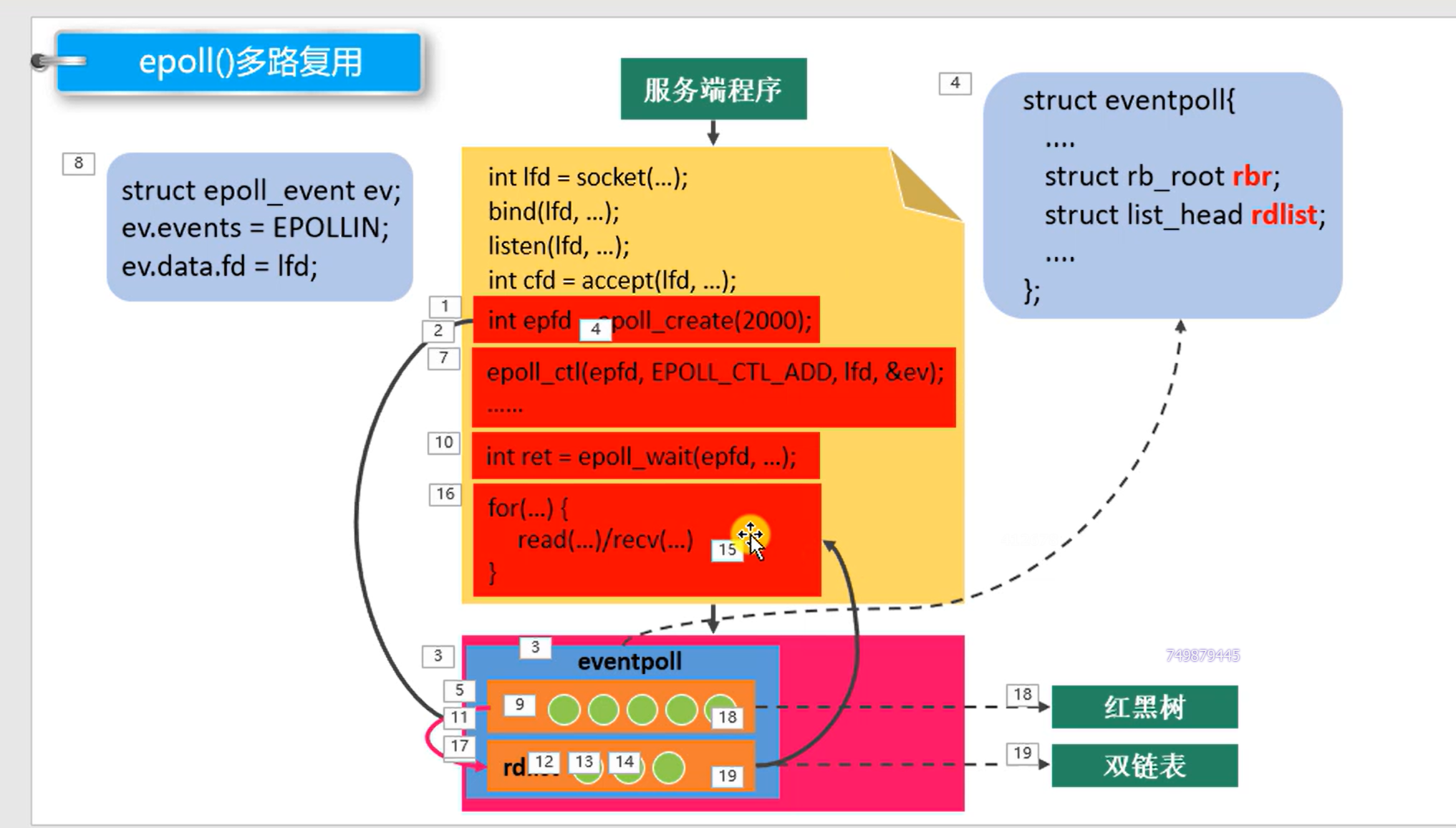
socket在每次执行select的时候都得要重新对time进行赋值 防止select修改了time参数 对下一次的判断造成干扰

在一般情况下，当您使用select函数或类似的多路复用机制来接收数据时，如果您没有在接收数据后读取数据或者没有读取完数据，并且再次调用select函数进行下一轮的多路复用操作，那么之前接收到的数据可能会丢失。丢失是当又有新数据发送过来了。没有新数据的话不会丢失。

Epoll 用红黑树管理的

Epoll 直接在内核区创建eventpoll。

Epoll 实例有两个指针一个指向要检测的，一个指向就绪的。



netstat -napt查看状态是否连接的