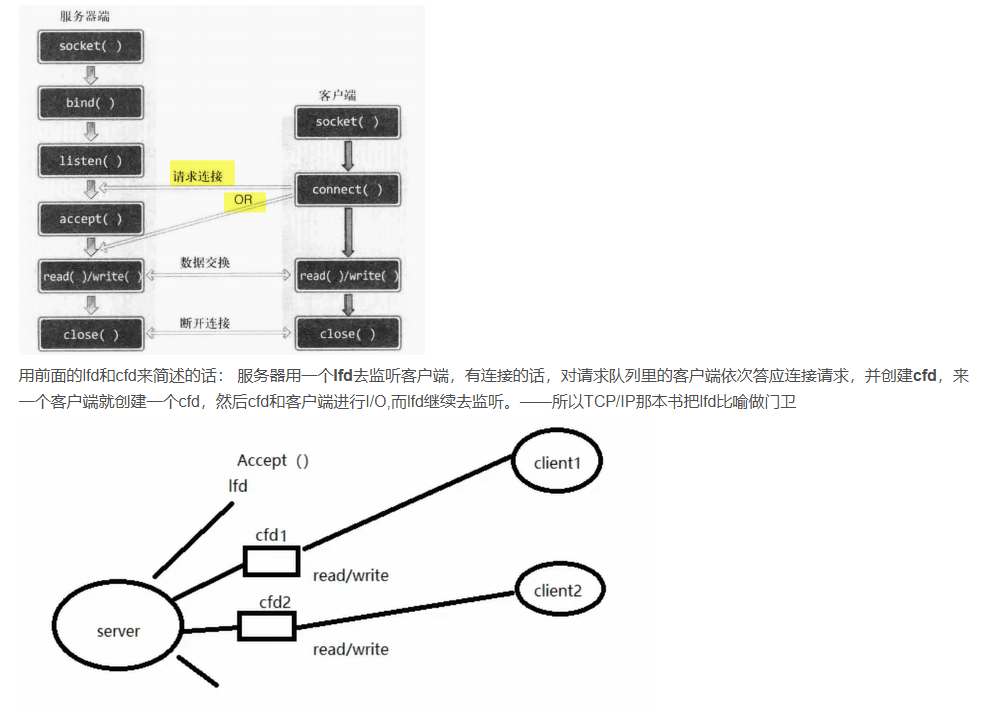
1. **连接建立**



第一步：建立socket，三个参数分别代表AF（IPV4或者IPV6）、套接字传输模式采用数据报（SOCKET\_PACKET）还是数据流（SOCKET\_STREAM）、传输协议IPPROTO（TCP、UDP等）。

第二步：分配套接字地址



第三步：绑定套接字fd和addr，bind方法最后一个参数代表套接字地址的size（以字节表示）

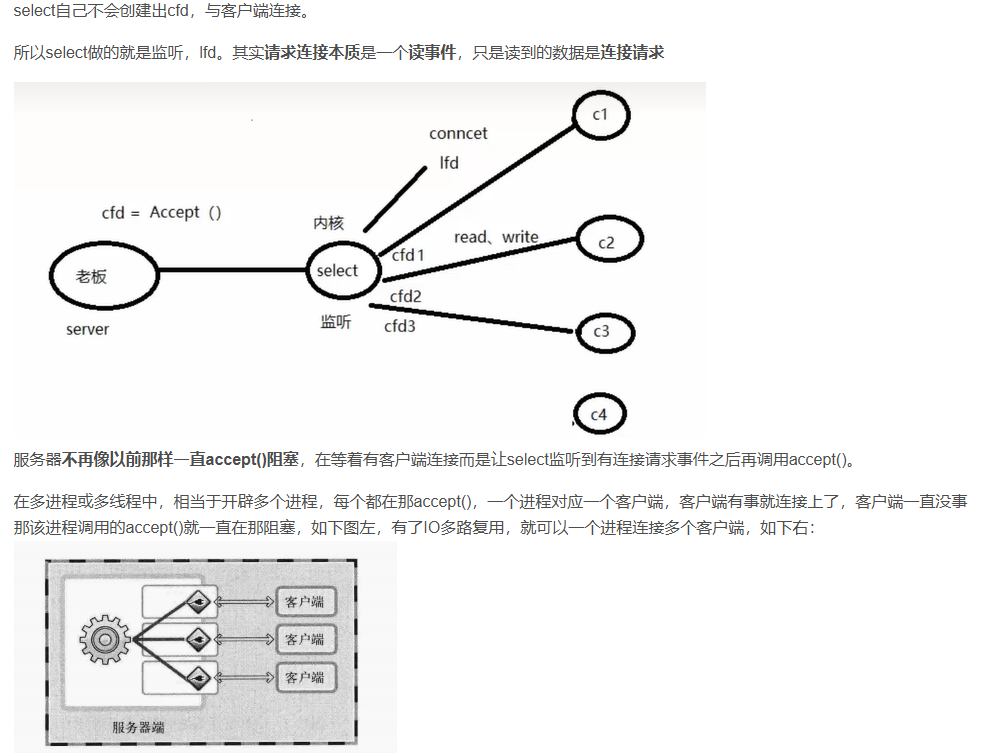
第四步：listen(fd, queue\_size)，最多接受N个连接

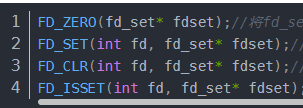
第五步：accept,其中第二个参数为新客户端地址信息,accept实际上是在检测server的全连接队列中有没有连接信息，而connect检测的是服务端是否发来了ACK数据包

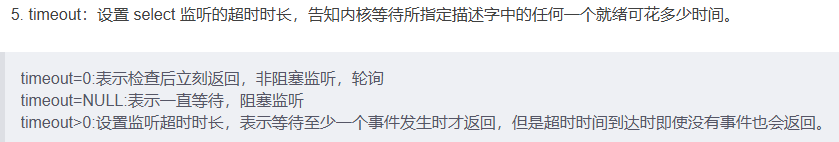
第六步：recv，检测rbuffer中是否有EOF标志

1. **Select**

有了select后可以实现io多路复用，即一个线程服务多个客户端，不用调用accept而避免阻塞。



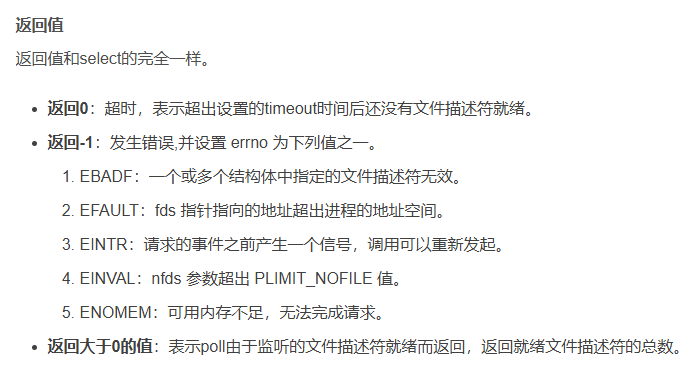




Select每次会检查传入的文件描述符集合，并将其中未就绪的置0，因此select前需要拷贝。Select实际上是一个轮询检查语句，直至有一个就绪或者超时则终止轮询

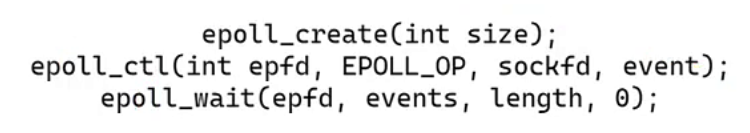
1. **Poll**



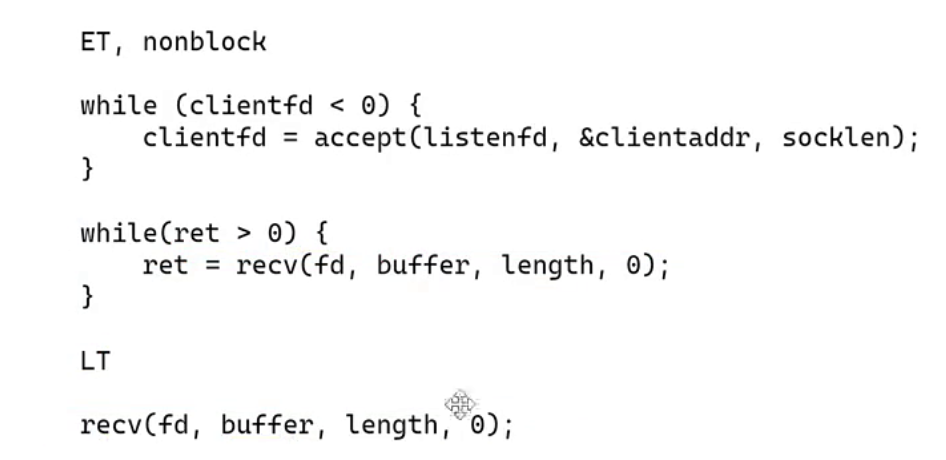


1. **Epoll**

Epoll的三个接口



int size大于0即可

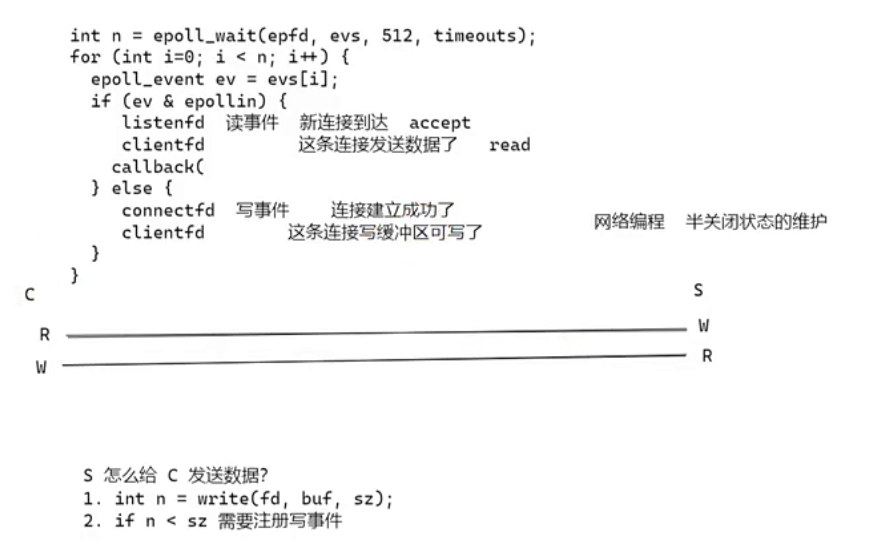


大数据，适合用LT，水平触发，不断地检查fd的可读可写状态，资源消耗大

小数据适合用ET，边沿触发，仅在未就绪转化为就绪通知一次，减少不必要的上下文切换，但数据较多时有可能丢包

1. **Reactor**





io是同步的，事件是异步的

1. **并发**

服务器五个参数：并发、qps、最大时延、每秒新建、吞吐量

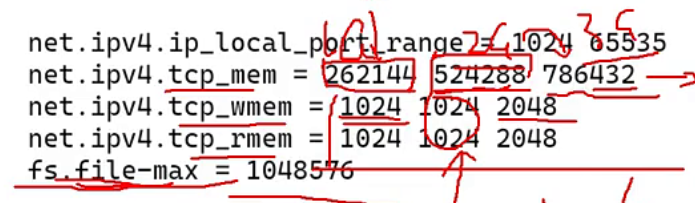
站在服务端的角度看五元组

(sip, dip, sport, dport, protocol)

sip，客户端地址，dip，服务端地址，sport，客户端端口，dport，服务端端口

站在客户端正好相反

一个tcp控制块由一个五元组唯一确定



Tcp\_mem，tcp协议栈内存空间，以页为单位，一页4KB，262144 \* 4KB = 1GB

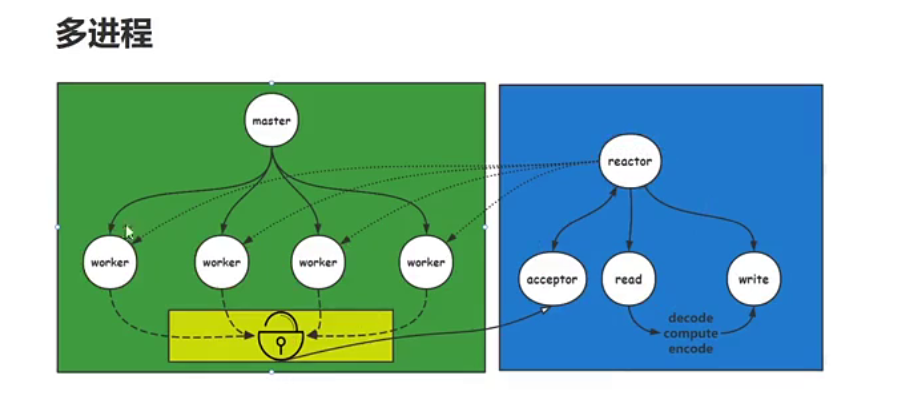
wmem，rmem以字节为单位，默认写缓存，读缓存大小为1KB

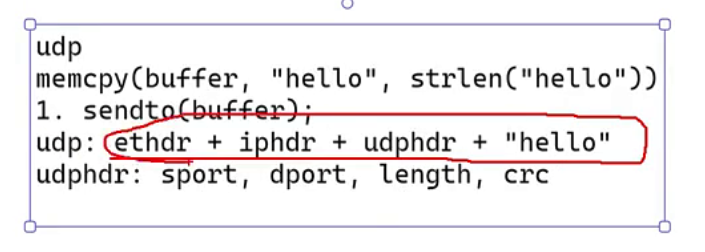
epoll本身可以达到100W并发级别

1. **http server五项须清晰阐明的技术**

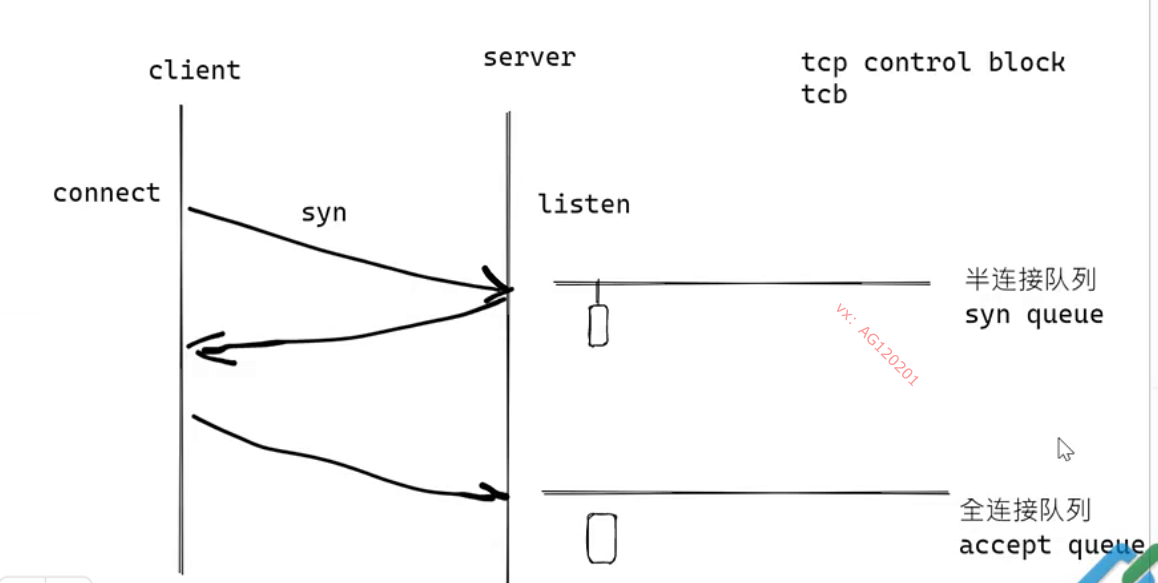


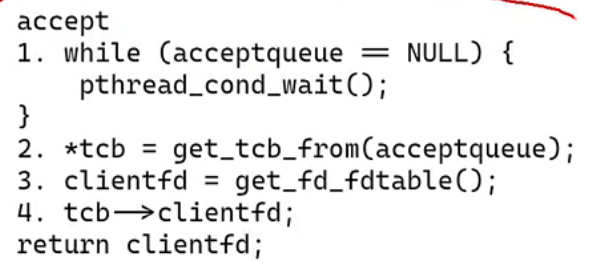
Redis，单reactor（单线程） 不好加锁，nginx多进程 业务交互少，memcached多线程 方便加锁且业务有关联





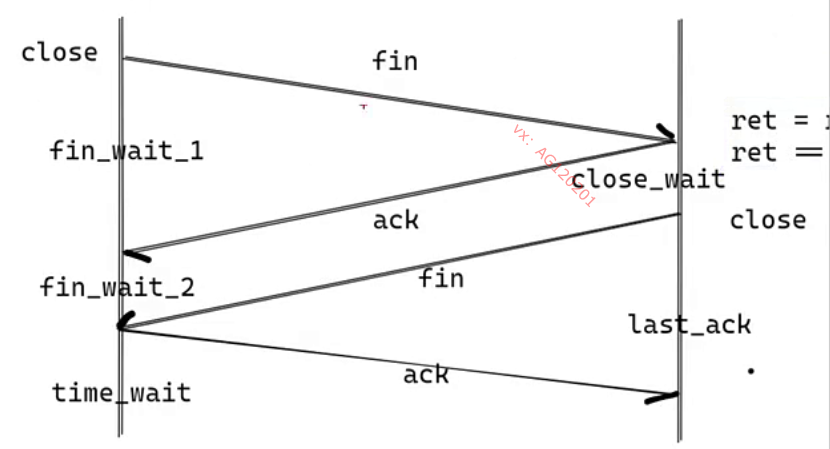
三次握手过程中，tcp数据头中的检验位一定全部清空





TCP数据头不带长度，网络每次传送的长度是由协议确定的

Seqnum和acknum都是以字节为下标的



1. 可靠性UDP协议KCP

客户端的KCP实例与服务端是一对一的关系，多客户端需要多KCP实例

