**网络编程必知**

**1：tcp和udp的区别**

TCP：是面向连接的流传输控制协议，具有高可靠性，确保传输数据的正确性，有验证重发机制，因此不会出现丢失或乱序。

UDP：是无连接的数据报服务，不对数据报进行检查与修改，无须等待对方的应答，会出现分组丢失、重复、乱序，但具有较好的实时性，UDP段结构比TCP的段结构简单，因此网络开销也小。

**2：流量控制和拥塞控制**

拥塞控制  
网络拥塞现象是指到达通信子网中某一部分的分组数量过多,使得该部分网络来不及处理,以致引起这部分乃至整个网络性能下降的现象,严重时甚至会导致网络通信业务陷入停顿,即出现死锁现象。拥塞控制是处理网络拥塞现象的一种机制。  
流量控制  
数据的传送与接收过程当中很可能出现收方来不及接收的情况,这时就需要对发方进行控制,以免数据丢失。

**3：多线程如何同步**

windows

线程同步有四种方式：[临界区、内核对象、互斥量、信号量](http://blog.csdn.net/windows_nt/article/details/8953088)。

[**Linux**](http://lib.csdn.net/base/linux)

线程同步有最常用的是：[互斥锁、条件变量和信号量。](http://blog.csdn.net/windows_nt/article/details/28676157)

**4：进程间通讯的方式有哪些，各有什么优缺点**

**[进程间通信](http://blog.csdn.net/windows_nt/article/details/28682333" \t "_blank)**

**5：tcp连接建立的时候3次握手，断开连接的4次握手的具体过程**

建立连接采用的3次握手协议，具体是指：  
第一次握手是客户端connect连接到server，server accept client的请求之后，向client端发送一个消息，相当于说我都准备好了，你连接上我了，这是第二次握手，第3次握手就是client向server发送的，就是对第二次握手消息的确认。之后client和server就开始通讯了。  
断开连接的4次握手,具体如下：  
断开连接的一端发送close请求是第一次握手，另外一端接收到断开连接的请求之后需要对close进行确认，发送一个消息，这是第二次握手，发送了确认消息之后还要向对端发送close消息，要关闭对对端的连接，这是第3次握手，而在最初发送断开连接的一端接收到消息之后，进入到一个很重要的状态time\_wait状态，这个状态也是面试官经常问道的问题，最后一次握手是最初发送断开连接的一端接收到消息之后。对消息的确认。

**6：epoll与select的区别**

select在一个进程中打开的最大fd是有限制的，由FD\_SETSIZE设置，默认值是2048。不过 epoll则没有这个限制，它所支持的fd上限是最大可以打开文件的数目，这个数字一般远大于2048，一般来说内存越大，fd上限越大，1G内存都能达到大约10w左右。  
  
select的轮询机制是系统会去查找每个fd是否数据已准备好，当fd很多的时候，效率当然就直线下降了，epoll采用基于事件的通知方式，一旦某个fd数据就绪时，内核会采用类似callback的回调机制，迅速激活这个文件描述符，而不需要不断的去轮询查找就绪的描述符，这就是epoll高效最本质的原因。  
  
无论是select还是epoll都需要内核把FD消息通知给用户空间，如何避免不必要的内存拷贝就很重要，在这点上，epoll是通过内核于用户空间mmap同一块内存实现的，而select则做了不必要的拷贝

**7：epoll中et和lt的区别与实现原理**

LT：水平触发，效率会低于ET触发，尤其在大并发，大流量的情况下。但是LT对代码编写要求比较低，不容易出现问题。LT模式服务编写上的表现是：只要有数据没有被获取，内核就不断通知你，因此不用担心事件丢失的情况。  
ET：边缘触发，效率非常高，在并发，大流量的情况下，会比LT少很多epoll的系统调用，因此效率高。但是对编程要求高，需要细致的处理每个请求，否则容易发生丢失事件的情况。