请解释下UDP和TCP的区别

TCP（Transmission Control Protocol，传输控制协议）是面向连接的协议，也就是说，在收发数据前，必须和对方建立可靠的连接。一个TCP连接必须要经过三次“对话”才能建立，其中的过程非常复杂，过程：主机A向主机B发出连接请求数据包：“我想给你发数据，可以吗？”，这是第一次对话；主机B向主机A发送同意连接和要求同步（同步就是两台主机一个在发送，一个在接收，协调工作）的数据包：“可以，你什么时候发？”，这是第二次对话；主机A再发出一个数据包确认主机B的要求同步：“我现在就发，你接着吧！”，这是第三次对话。三次“对话”的目的是使数据包的发送和接收同步，经过三次“对话”之后，主机A才向主机B正式发送数据。

TCP建立连接要进行3次握手 1) 主机A通过向主机B 发送一个含有同步序列号的标志位的数据段给主机B ,向主机B 请求建立连接,通过这个数据段,主机A告诉主机B 两件事:我想要和你通信;你可以用哪个序列号作为起始数据段来回应我 . 2) 主机B 收到主机A的请求后,用一个带有确认应答(ACK)和同步序列号(SYN)标志位的数据段响应主机A,也告诉主机A两件事:我已经收到你的请求了,你可以传输数据了;你要用序列号作为起始数据段来回应我 3)主机A收到这个数据段后,再发送一个确认应答,确认已收到主机B 的数据段:"我已收到回复,我现在要开始传输实际数据了 3次握手就完成了,主机A和主机B 就可以传输数据 3次握手的特点: 没有应用层的数据 SYN这个标志位只有在TCP建产连接时才会被置1 握手完成后SYN标志位被置0

TCP断开连接要进行4次 1) 当主机A完成数据传输后,将控制位FIN置1,提出停止TCP连接的请求 2) 主机B收到FIN后对其作出响应,确认这一方向上的TCP连接将关闭,将ACK置1 3) 由B 端再提出反方向的关闭请求,将FIN置1 4) 主机A对主机B的请求进行确认,将ACK置1,双方向的关闭结束. 由TCP的三次握手和四次断开可以看出,TCP使用面向连接的通信方式,大大提高了数据通信的可靠性,使发送数据端 和接收端在数据正式传输前就有了交互,为数据正式传输打下了可靠的基础

名词解释 ACK TCP报头的控制位之一,对数据进行确认.确认由目的端发出,用它来告诉发送端这个序列号之前的数据段都收到了.比如,确认号为X,则表示前X-1个数据段都收到了,只有当ACK=1时,确认号才有效,当ACK=0时,确认号无效,这时会要求重传数据,保证数据的完整性. SYN 同步序列号,TCP建立连接时将这个位置1 FIN 发送端完成发送任务位,当TCP完成数据传输需要断开时,提出断开连接的一方将这位置1

TCP的包头结构： 源端口 16位 目标端口 16位 序列号 32位 回应序号 32位 TCP头长度 4位 reserved 6位 控制代码 6位 窗口大小 16位 偏移量 16位 校验和 16位 选项 32位(可选) 这样我们得出了TCP包头的最小长度，为20字节。

UDP（User Data Protocol，用户数据报协议） （1） UDP是一个非连接的协议，传输数据之前源端和终端不建立连接，当它想传送时就简单地去抓取来自应用程序的数据，并尽可能快地把它扔到网络上。在发送端，UDP传送数据的速度仅仅是受应用程序生成数据的速度、计算机的能力和传输带宽的限制；在接收端，UDP把每个消息段放在队列中，应用程序每次从队列中读一个消息段。 （2） 由于传输数据不建立连接，因此也就不需要维护连接状态，包括收发状态等，因此一台服务机可同时向多个客户机传输相同的消息。 （3） UDP信息包的标题很短，只有8个字节，相对于TCP的20个字节信息包的额外开销很小。 （4） 吞吐量不受拥挤控制算法的调节，只受应用软件生成数据的速率、传输带宽、源端和终端主机性能的限制。 （5）UDP使用尽最大努力交付，即不保证可靠交付，因此主机不需要维持复杂的链接状态表（这里面有许多参数）。 （6）UDP是面向报文的。发送方的UDP对应用程序交下来的报文，在添加首部后就向下交付给IP层。既不拆分，也不合并，而是保留这些报文的边界，因此，应用程序需要选择合适的报文大小。 我们经常使用“ping”命令来测试两台主机之间TCP/IP通信是否正常，其实“ping”命令的原理就是向对方主机发送UDP数据包，然后对方主机确认收到数据包，如果数据包是否

到达的消息及时反馈回来，那么网络就是通的。

UDP的包头结构： 源端口 16位 目的端口 16位 长度 16位 校验和 16位

小结TCP与UDP的区别： 1.基于连接与无连接； 2.对系统资源的要求（TCP较多，UDP少）； 3.UDP程序结构较简单； 4.流模式与数据报模式 ； 5.TCP保证数据正确性，UDP可能丢包，TCP保证数据顺序，UDP不保证。

UDP应用场景： 1.面向数据报方式 2.网络数据大多为短消息 3.拥有大量Client 4.对数据安全性无特殊要求 5.网络负担非常重，但对响应速度要求高

TCP: TCP编程的服务器端一般步骤是： 1、创建一个socket，用函数socket()； 2、设置socket属性，用函数setsockopt(); * 可选 3、绑定IP地址、端口等信息到socket上，用函数bind(); 4、开启监听，用函数listen()； 5、接收客户端上来的连接，用函数accept()； 6、收发数据，用函数send()和recv()，或者read()和write(); 7、关闭网络连接； 8、关闭监听；

TCP编程的客户端一般步骤是： 1、创建一个socket，用函数socket()； 2、设置socket属性，用函数setsockopt();* 可选 3、绑定IP地址、端口等信息到socket上，用函数bind();* 可选 4、设置要连接的对方的IP地址和端口等属性； 5、连接服务器，用函数connect()； 6、收发数据，用函数send()和recv()，或者read()和write(); 7、关闭网络连接；

UDP: 与之对应的UDP编程步骤要简单许多，分别如下： UDP编程的服务器端一般步骤是： 1、创建一个socket，用函数socket()； 2、设置socket属性，用函数setsockopt();* 可选 3、绑定IP地址、端口等信息到socket上，用函数bind(); 4、循环接收数据，用函数recvfrom(); 5、关闭网络连接；

UDP编程的客户端一般步骤是： 1、创建一个socket，用函数socket()； 2、设置socket属性，用函数setsockopt();* 可选 3、绑定IP地址、端口等信息到socket上，用函数bind();* 可选 4、设置对方的IP地址和端口等属性; 5、发送数据，用函数sendto(); 6、关闭网络连接；