先从协议最基本的TCP粘包问题搞起，把计算机网络这部分基础夯实一下。

# TCP协议的简单介绍

## 1、TCP是面向连接的运输层协议

简单来说，在使用TCP协议之前，必须先建立TCP连接，就是我们常说的三次握手。在数据传输完毕之后，必须是释放已经建立的TCP连接，否则会发生不可预知的问题，造成服务的不可用状态。

## 2、每一条TCP连接的两个端点都是可靠连接

TCP连接是从Server端到Client端的点对点的，通过TCP传输数据，无差错，不重复不丢失。

## 3、TCP协议的通信是全双工的

TCP协议允许通信双方的应用程序在任何时候都能发送数据。TCP 连接的两端都设有发送缓冲区和接收缓冲区，用来临时存放双向通信的数据。发送数据时，应用程序把数据传送给TCP的缓冲后，就可以做自己的事情，而TCP在合适的时候将数据发送出去。在接收的时候，TCP把收到的数据放入接收缓冲区，上层应用在合适的时候读取数据。

## 4、TCP协议是面向字节流的

打个比方比喻TCP，你家里有个蓄水池，你可以里面倒水，蓄水池上有个龙头，你可以通过龙头将水池里的水放出来，然后用各种各样的容器装（杯子、矿泉水瓶、锅碗瓢盆）接水。

上面的例子中，往水池里倒几次水和接几次水是没有必然联系的，也就是说你可以只倒一次水，然后分10次接完。另外，水池里的水接多少就会少多少；往里面倒多少水，就会增加多少水，但是不能超过水池的容量，多出的水会溢出。

结合TCP的概念，水池就好比接收缓存，倒水就相当于发送数据，接水就相当于读取数据。好比你通过TCP连接给另一端发送数据，你只调用了一次write，发送了100个字节，但是对方可以分10次收完，每次10个字节；你也可以调用10次write，每次10个字节，但是对方可以一次就收完。（假设数据都能到达）但是，你发送的数据量不能大于对方的接收缓存（流量控制），如果你硬是要发送过量数据，则对方的缓存满了就会把多出的数据丢弃。

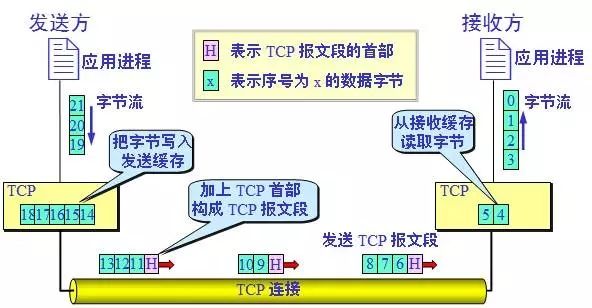
## 5、UDP面向报文

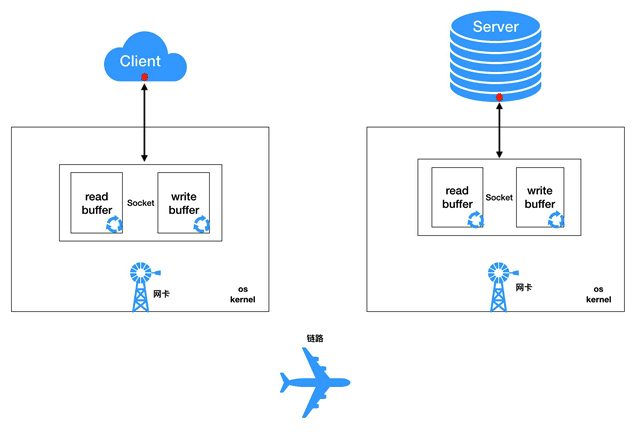
UDP和TCP不同，发送端调用了几次write，接收端必须用相同次数的read读完。UPD是基于报文的，在接收的时候，每次最多只能读取一个报文，报文和报文是不会合并的，如果缓冲区小于报文长度，则多出的部分会被丢弃。也就说，如果不指定MSG\_PEEK标志，每次读取操作将消耗一个报文。

# 粘包的原因

## 1. TCP传输的过程

TCP并不知道所传输的字节流的含义，TCP并不能保证接收方应用程序和发送方应用程序所发出的数据块具有对应大小的关系（这就是TCP传输过程中产生的粘包问题）。但是应用程序接收方最终受到的字节流与发送方发送的字节流是一定相同的。因此，我们在使用TCP协议的时候应该制定合理的粘包拆包策略。





## 2. TCP粘包问题出现

出现的粘包问题一共有三种情况。

现在假设客户端向服务端连续发送了两个数据包，用packet1和packet2来表示，那么服务端收到的数据可以分为三种，现列举如下：

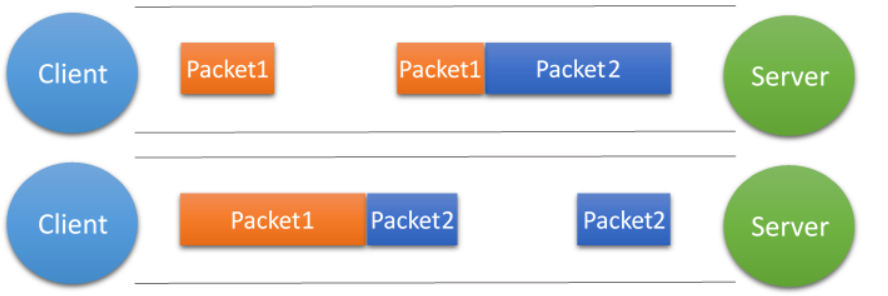
第一种情况，接收端正常收到两个数据包，即没有发生拆包和粘包的现象，此种情况不在本文的讨论范围内。



第二种情况，接收端只收到一个数据包，由于TCP是不会出现丢包的，所以这一个数据包中包含了发送端发送的两个数据包的信息，这种现象即为粘包。这种情况由于接收端不知道这两个数据包的界限，所以对于接收端来说很难处理。



第三种情况，这种情况有两种表现形式，如下图。接收端收到了两个数据包，但是这两个数据包要么是不完整的，要么就是多出来一块，这种情况即发生了拆包和粘包。这两种情况如果不加特殊处理，对于接收端同样是不好处理的。



**第一种情况：**

如上面第一个图中所示，服务端一共读到两个数据包，每个数据包都是完成的，并没有发生粘包的问题，这种情况比较好处理，服务器只需要简单的从网络缓冲区去读就好了，每次服务端读取到的消息都是完成的，并不会出现数据不正确的情况。

**第二种情况：**

服务端仅收到一个数据包，这个数据包包含客户端发出的两条消息的完整信息，这个时候基于第一种情况的逻辑实现的服务端就蒙了，因为服务端并不能很好的处理这个数据包，甚至不能处理，这种情况其实就是TCP的粘包问题。

**第三种情况：**

服务端收到了两个数据包，第一个数据包只包含了第一条消息的一部分，第一条消息的后半部分和第二条消息都在第二个数据包中，或者是第一个数据包包含了第一条消息的完整信息和第二条消息的一部分信息，第二个数据包包含了第二条消息的剩下部分，这种情况其实是发送了TCP拆包问题，因为发生了一条消息被拆分在两个包里面发送了，同样上面的服务器逻辑对于这种情况是不好处理的。

## 3、为什么会发生TCP粘包、拆包

1）应用程序写入的数据大于套接字缓冲区大小，这将会发生拆包。

2）应用程序写入数据小于套接字缓冲区大小，网卡将应用多次写入的数据发送到网络上，这将会发生粘包。

3）进行MSS（最大报文长度）大小的TCP分段，当TCP报文长度-TCP头部长度>MSS的时候将发生拆包。

4）接收方法不及时读取套接字缓冲区数据，这将发生粘包。

## 4、如何处理粘包、拆包

通常会有以下一些常用的方法：

1）使用带消息头的协议、消息头存储消息开始标识及消息长度信息，服务端获取消息头的时候解析出消息长度，然后向后读取该长度的内容。

2）设置定长消息，服务端每次读取既定长度的内容作为一条完整消息，当消息不够长时，空位补上固定字符。

3）设置消息边界，服务端从网络流中按消息边界分离出消息内容。

SOH开户ETX开户指令的报文内容EOT