计网2----网络是怎么变得靠谱的

传输层

1. 概念：传输层为第四层架构，前三层架构只能保证数据包能够从A传输到B，但是还是存在一些问题。

2. UDP协议：

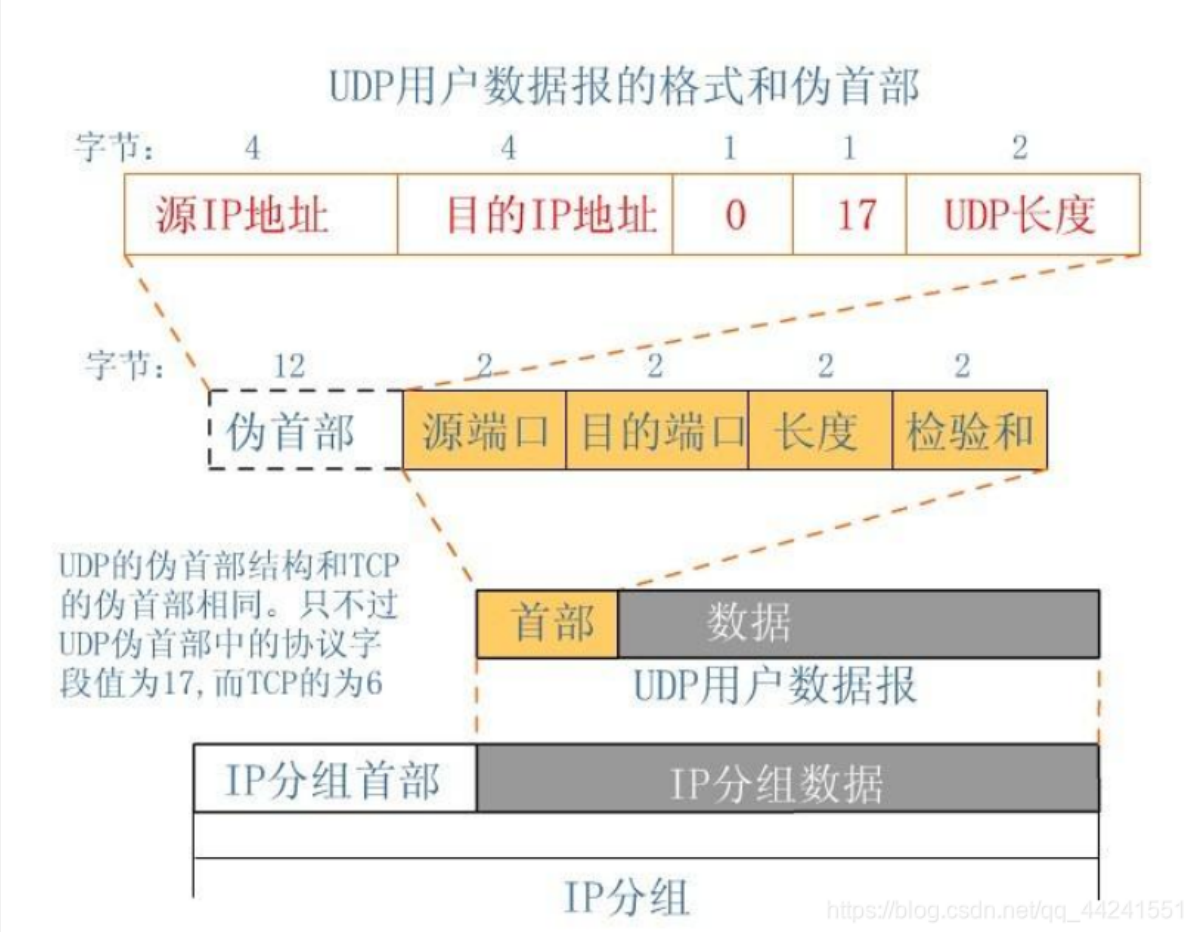
   UDP协议主要是将网络数据流量压缩成数据包的形式，数据包由首部和数据构成。（用于解决：电脑之中有不同的进程，只从A给B是不知道具体是哪个进程给哪个进程的）

首部（包头）有着目的端口，源端口，数据包长度和校验和

UDP报头

偏移	字节	0								1								2								3							
字节	位	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0	0	来源连接端口																目的连接端口														
4	32	报文长度																校验和														

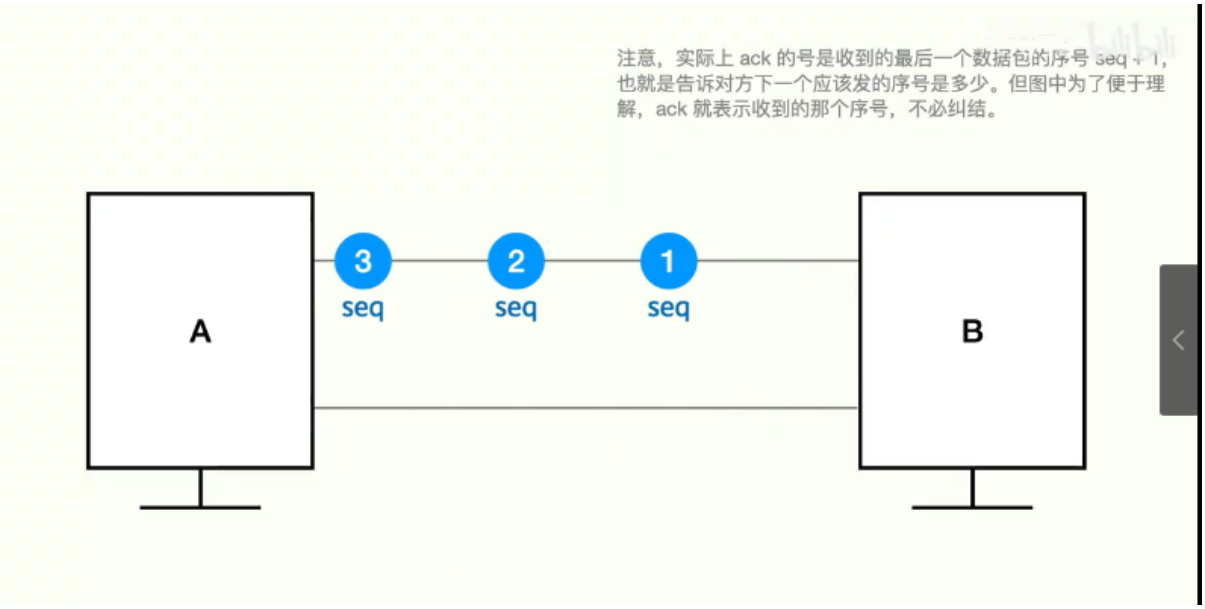
此外还有伪包头，其中包含源IP和目标IP，暂且不表



如此一来，数据包就变成了这样，如此一来将主机与主机的通信升级成了端口与端口之间的通信

3. 停止等待协议：

    1. 概念：A每次发送一个包，一定要接收到来自B的确认，即ACK，才能继续发送下一个包，否则重发这个包。(用于解决网络不可靠的问题，即传输过程中数据包可能丢失，无法确定B是否真的收到了A的数据包)
    2. 特点：这个特点称为***可靠交付***
    3. 改良：鉴于一个个进行协议效率太低，于是采用流水线的方式提高效率
    4. 改良的改良：但是有些时候传输数据包时有很多种方式，不能保证电脑B收到包的顺序和发出报的顺序相同。于是我们新加上了***确认序号*** sql和ack



值得一提的是，电脑B发回的确认序号ack为***累计确认/应答***，例如收到ack序号为3，不仅表示收到了三号包，还意味着三号包之前的两个包也受到了

4. 滑动窗口：(详见视频4min50s)

    1. 概念： 电脑A与B会互相传输***窗口大小***，这个值意味着它们各自的接受能力。而发送者会根据接收到窗口大小将自己的数据分为四类

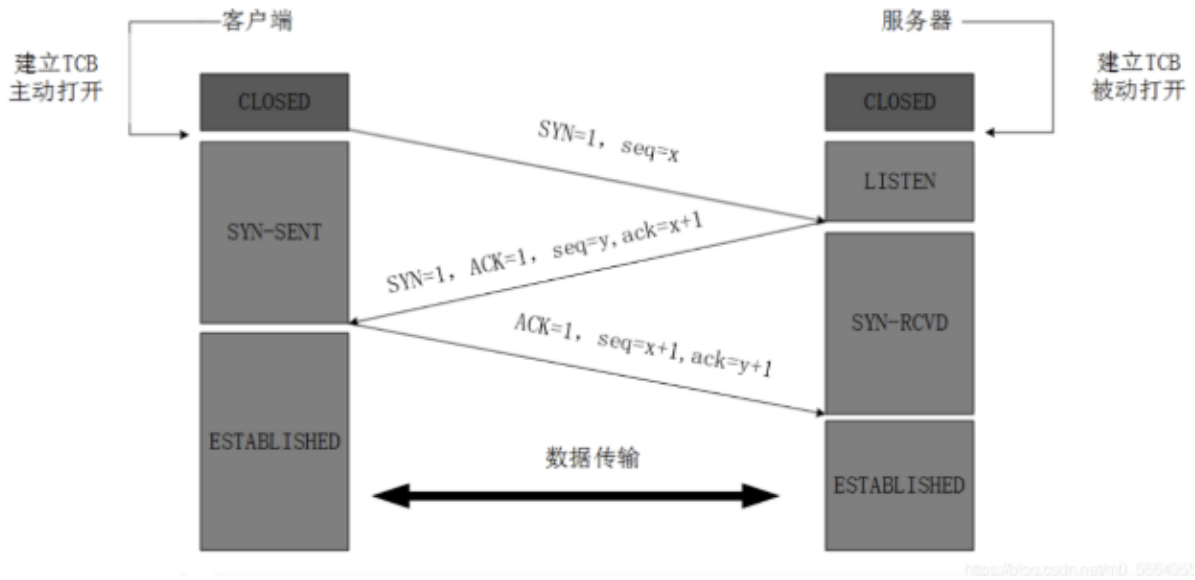
    2. 滑动窗口：又因为对方发来的每一个包都可以重新设定一个窗口大小，于是窗口在不断后移的过程中（发送数据）窗口大小还会不断改变，此现象被形象的称为滑动窗口

5. 拥塞窗口：

1. 概念：在发送数据包的过程中可能会因为网络堵塞导致丢包，为此我们需要探测网络的接受能力。于是发送这用各种各样的算法进行测试，获得的一个值成为 ***堵塞窗口***，其反映了网络的接收能力
2. 通信窗口：同行窗口毫无疑问要取二者的最小值，即 min = （堵塞窗口，滑动窗口）

6. TCP三次握手：

    1. TCP连接过程：



    最开始的时候客户端和服务器都是处于CLOSED关闭状态。主动打开连接的为客户端，被动打开连接的是服务器。

    TCP服务器进程先创建传输控制块TCB，时刻准备接受客户进程的连接请求，此时服务器就进入了 LISTEN 监听状态

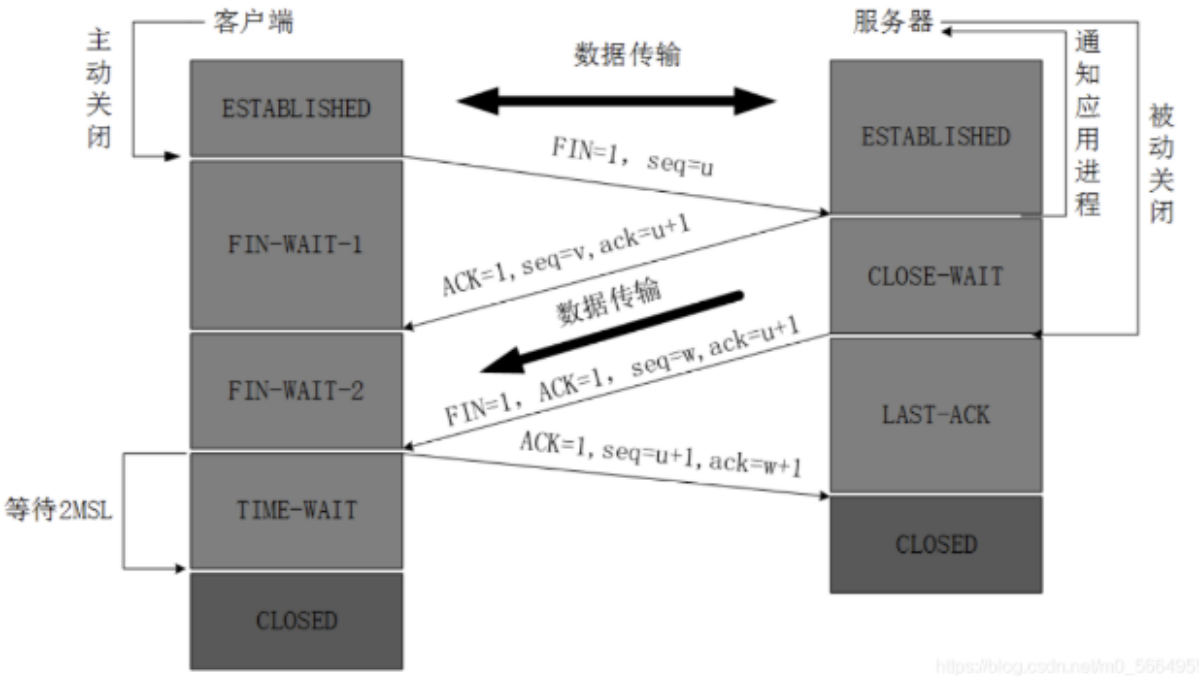
    第一次握手 TCP客户进程也是先创建传输控制块TCB，然后向服务器发出连接请求报文，这是报文首部中的同部位SYN=1，同时选择一个初始序列号 seq=x ，此时，TCP客户端进程进入了 SYN-SENT 同步已发送状态

    第二次握手 TCP服务器收到请求报文后，如果同意连接，则会向客户端发出确认报文。确认报文中应该 ACK=1，SYN=1，确认号是ack=x+1，同时也要为自己初始化一个序列号 seq=y，此时，TCP服务器进程进入了 SYN-RCVD 同步收到状态

    第三次握手 TCP客户端收到确认后，还要向服务器给出确认。确认报文的ACK=1，ack=y+1，自己的序列号seq=x+1，此时，TCP连接建立，客户端进入ESTABLISHED已建立连接状态 触发三次握手

7. TCP四次挥手：

1. 挥手过程：



数据传输完毕后，双方都可释放连接。最开始的时候，客户端和服务器都是处于ESTABLISHED状态，然后客户端主动关闭，服务器被动关闭。

第一次挥手 客户端发出连接释放报文，并且停止发送数据。释放数据报文首部，FIN=1，其序列号为seq=u（等于前面已经传送过来的数据的最后一个字节的序号加1），此时，客户端进入FIN-WAIT-1（终止等待1）状态

第二次挥手 服务器端接收到连接释放报文后，发出确认报文，ACK=1，ack=u+1，并且带上自己的序列号seq=v，此时，服务端就进入了CLOSE-WAIT 关闭等待状态

第三次挥手 客户端接收到服务器端的确认请求后，客户端就会进入FIN-WAIT-2（终止等待2）状态，等待服务器发送连接释放报文，服务器将最后的数据发送完毕后，就向客户端发送连接释放报文，服务器就进入了LAST-ACK（最后确认）状态，等待客户端的确认。

第四次挥手 客户端收到服务器的连接释放报文后，必须发出确认，ACK=1，ack=w+1，而自己的序列号是seq=u+1，此时，客户端就进入了TIME-WAIT（时间等待）状态，但此时TCP连接还未终止，必须要经过2MSL后（最长报文寿命），当客户端撤销相应的TCB后，客户端才会进入CLOSED关闭状态，服务器端接收到确认报文后，会立即进入CLOSED关闭状态，到这里TCP连接就断开了，四次挥手完成

2. 为什么客户端要等待2MSL?

主要原因是为了保证客户端发送那个的第一个ACK报文能到到服务器，因为这个ACK报文可能丢失，并且2MSL是任何报文在网络上存在的最长时间，超过这个时间报文将被丢弃，这样新的连接中不会出现旧连接的请求报文。

总结

网络的前三层架构保证了包能够发到另一台电脑。而第四层结构保证了发送的可靠性（解决了包丢了怎么办，传快了怎么办，传多了怎么办，没收到怎么办等等等等）

> 在 TCP 协议中，ACK、FIN 和 SEQ 是用来控制连接和数据传输的标志位和字段，下面将一一解释它们的含义：
>
> 1. ACK (Acknowledgment)： 定义：ACK 是一个标志位，用来表示接收到的数据包的确认信息。在 TCP 数据包的头部中，有一个 "确认号"（Acknowledgment Number），该字段表示发送方期望接收到的下一个字节的序列号。也就是说，ACK=1 表示这个报文是一个确认报文，意味着接收方已经成功接收到前一个数据包中的数据。 作用：通过 ACK，接收方可以告诉发送方它已经成功收到数据包，从而实现数据的可靠传输。
> 2. FIN (Finish)： 定义：FIN 是连接释放的标志位，表示发送方已经没有数据要发送，并且希望关闭连接。当一方发送带有 FIN=1 的报文时，它表示自己没有更多数据要发送，准备断开连接。 作用：通过 FIN，一方告诉对方自己准备关闭连接，进入连接终止阶段。需要注意的是，FIN 只能表示“关闭连接请求”，而不会立即结束连接，双方需要通过四次挥手来确保连接正确地关闭。
> 3. SEQ (Sequence Number)： 定义：SEQ 是序列号字段，每个 TCP 数据包都带有一个序列号，它表示该数据包中的第一个字节的位置。序列号是 TCP 中流控制和数据重组的重要部分，因为它帮助接收方按照正确的顺序重新组装数据包。 作用：序列号是实现可靠数据传输的基础。每发送一个数据包，序列号都会增加，接收方通过序列号来确认数据是否按顺序到达，是否有丢失或重复的数据包。

> 举个例子： 假设客户端和服务器之间正在传输数据： 客户端向服务器发送数据时，会设置一个序列号 seq=100，并通过 ACK=0 来表示这是一个数据报文，而非确认报文。
> 服务器接收到这个数据后，会返回一个确认报文，ACK=1，并在确认报文中带上一个确认号 ack=101（表示它期待收到序列号为 101 的数据）。
> 在断开连接时，客户端发送 FIN=1 报文，表示它已经没有数据要发送了，并希望关闭连接。服务器收到后，返回确认报文，ACK=1，确认客户端的关闭请求。最终双方通过四次挥手完成连接的关闭。
> 通过这些字段，TCP 能够确保数据的顺利传输和连接的正确管理。