在复杂的计算机网络世界，要做到可靠可不是一件简单的事情，**学习可靠传输相关问题时，请不要将目光局限于数据链路层**，你会发现这是后续学习TCP的基础和核心。

对了，**我们讨论下面可靠传输的问题时，考虑的是在互联网中传输，而不是简单的点对点通信**，前者才具有讨论的意义和价值。

一、棘手的可靠传输问题

我们在上一篇学习了差错检测问题，可使用差错检测技术比如CRC循环冗余校验来检测帧在传输过程中是否出现了误码（比特错误）。

- 当数据链路层只需要提供不可靠传输服务时，接收端仅仅丢弃有误码的帧即可，其他啥都不做。
- 当数据链路层需要向上层提供可靠传输服务时，一切就会变得棘手，接收端检测出帧有误码，那么就必须要告诉发送方你这个帧有误码，请重发。但是万一这个通知帧也出现了误码呢?

可以初步看出来，**要实现可靠传输可不是一件容易的事情，为了实现可靠传输是要付出很大的开销和代价的**。

因此，一般情况下：

- 有线链路的误码率比较低，为了减少开销，并不要求数据链路层向上提供可靠传输服务，即使出现了误码，可靠传输的问题由上层处理。
- 无线链路易受干扰，误码率比较高，因此要求数据链路层必须向上层提供可靠传输服务。

二、传输差错可不止比特差错

比特差错只是传输差错中的一种，在计算机网络中，传输差错还包括分组丢失、分组失序以及分组重复。

注意这里出现了分组这样的名词，这在《概述篇：交换技术和网络拓扑》文章中介绍的分组交换技术中提及到了分组概念，这实际上是网络层的概念，这意味着传输差错可不仅仅局限于数据链路层的比特差错。

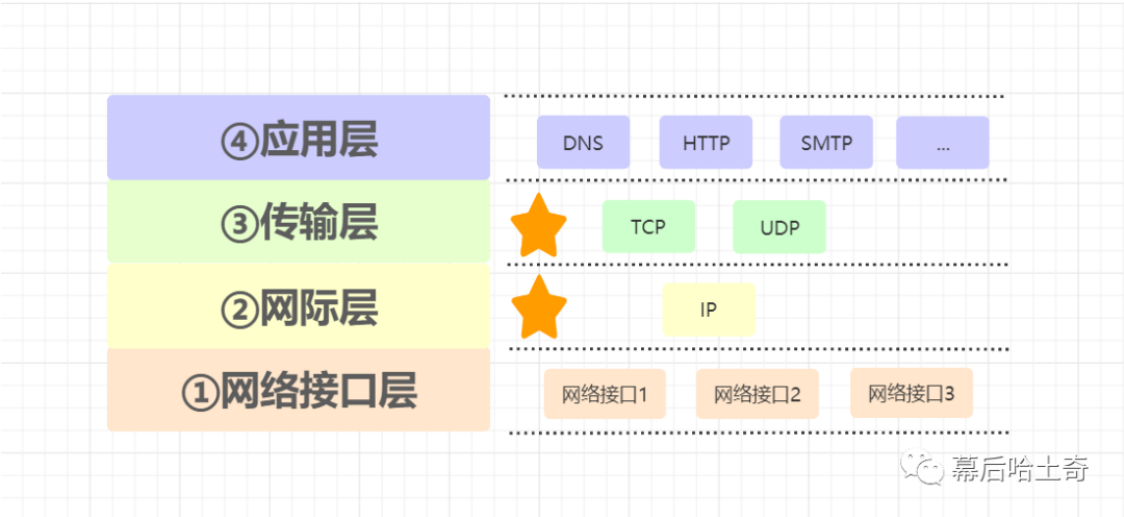
主机A需要给主机B发送数据，中间经过路由器R，可能会遇到如下情况：

- 分组丢失：当路由器R输入队列满了，则会主动丢弃该分组。
- 分组失序：主机A依次发送了分组1、2、3，但是到达主机B可能是3、2、1这样的顺序。
- 分组重复：某个分组由于某种原因在网络中滞留，没有及时到达主机B，这可能造成主机A对该分组的超时重发，重发的分组到达主机B，但是一段时间后滞留的分组也到达了主机B，引发分组重复问题。

分组丢失、分组失序、分组重复这些传输差错，一般不会出现在数据链路层，而是会出现在其上层。

从这里可以看出来，可靠传输服务不仅局限于数据链路层，其他层都可以选择实现可靠传输。

还记得TCP/IP协议族吗?



- 802.11无线局域网标准要求数据链路层实现可靠传输，而以太网不要求数据链路层实现可靠传输。
- IP网际层向其上层提供无连接、不可靠传输服务。
- TCP向其上层提供面向连接的可靠传输服务，UDP向其上层提供无连接、不可靠传输服务。

可以看到，除了数据链路层，传输层如果用的是TCP协议，也需要对上层负责可靠传输。

关于如何实现可靠传输，下面介绍三种实现机制：**停止等待协议SW、回退N帧协议GBN、选择重传协议SR**。

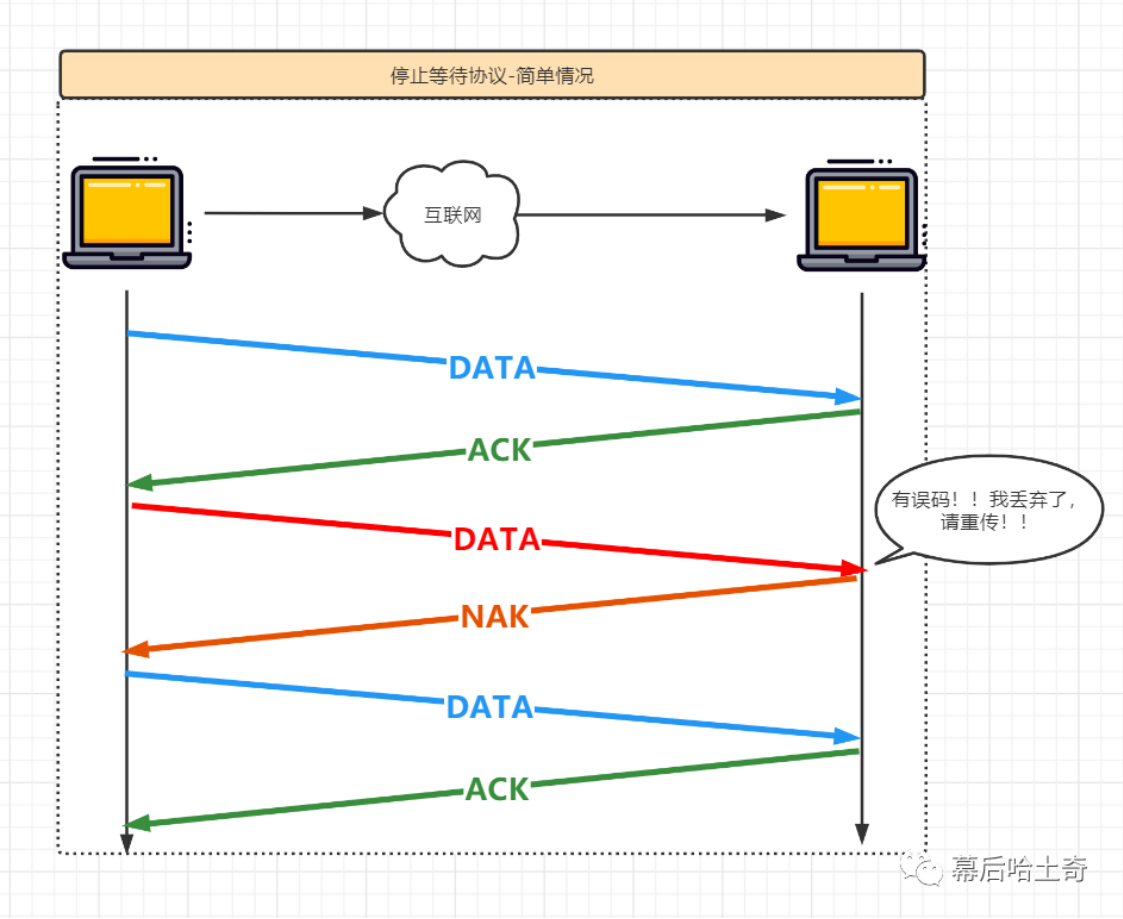
请注意，这三种可靠传输实现机制的基本原理并不仅限于数据链路层，可以应用到计算机网络体系结构的各层协议中。

三、停止等待协议

我们按照顺序，先来聊聊停止等待协议，英文名叫做stop-and-wait。

顾名思义，发送方每发送完一个分组，就停止发送下一个数据分组，等待来自接收方的确认分组或否认分组：

- 若收到确认分组（ACK），则可继续发送下一个数据分组；
- 若收到否认分组（NAK），则重发之前发送的那个数据分组。



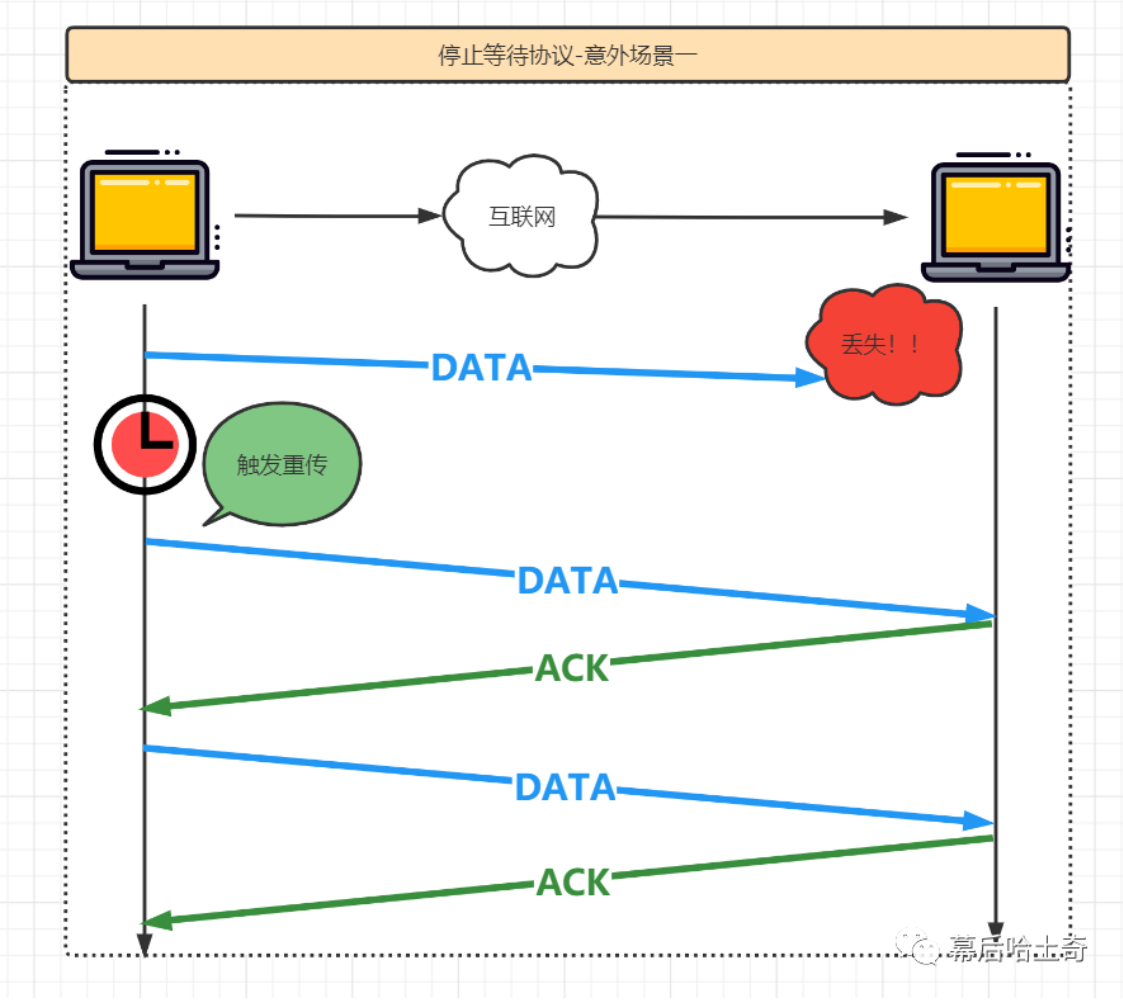
这样就实现了发送方发送什么、接收方就能收到什么的目的，确保了可靠传输。

但是实际情况会由于网络的不确定性，远不止这两种理想场景！

场景一：发送方的数据分组在传输过程中丢失

此时接收方收不到数据分组，也就不会发送ACK或NAK，如果不采取其他措施的话，发送方就会一直处于等待ACK或NAK的状态。

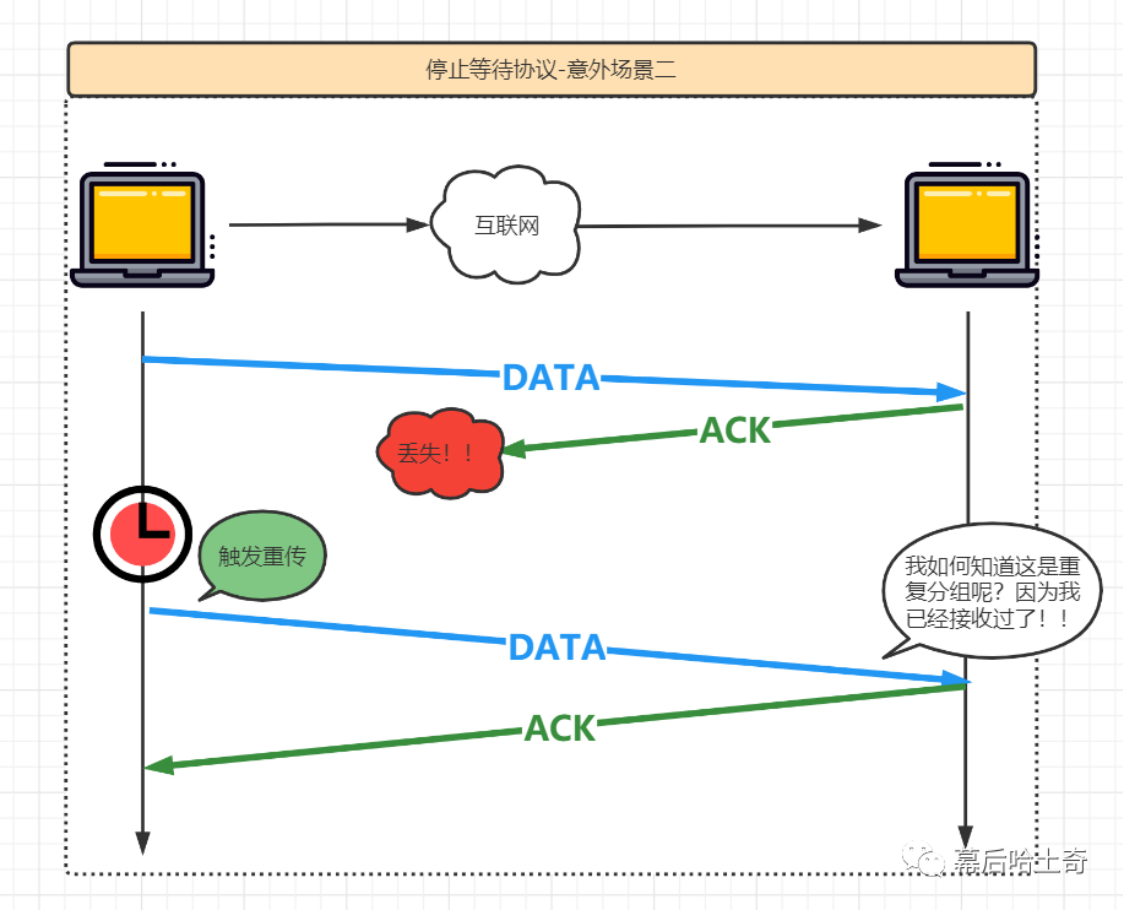
如何解决呢？可以在发送方发送完一个数据分组时，启动一个超时计时器，如果超出了重传时间仍然收不到接收方的ACK或NAK，则重传原来的数据分组，这就是超时重传。



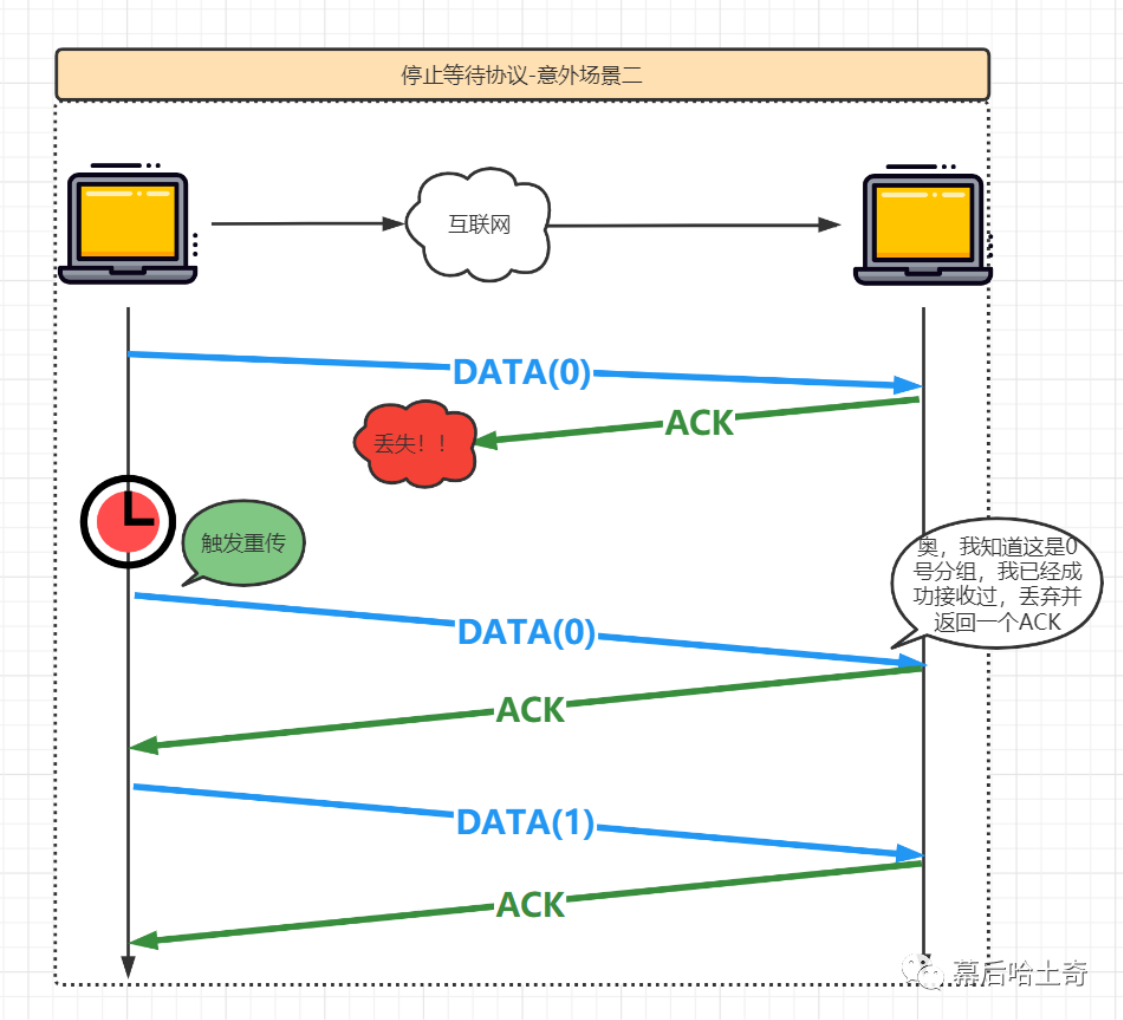
一般可将超时重传时间选为略大于“从发送方到接收方的平均往返时间”。

场景二：接收方的ACK或NAK分组在传输过程中丢失

发送方的数据分组成功到达了接收方，接收方发送的ACK或NAK分组丢失，此时由于超出了重传时间发送方仍然没有收到ACK或NAK分组，那么触发重传机制，此时刚才的数据分组重新又到达了接收方，那么接收方如何处理这种重复分组呢?



为了避免分组重复这种传输错误，必须给每个分组带上序号。根据序号，接收方即可判断该分组是不是重复分组了，接收方可丢弃重复的数据分组，并给发送方发送针对该重复数据分组的ACK分组，避免发送方再次超时重传。



场景三：接收方的ACK或NAK分组迟到了

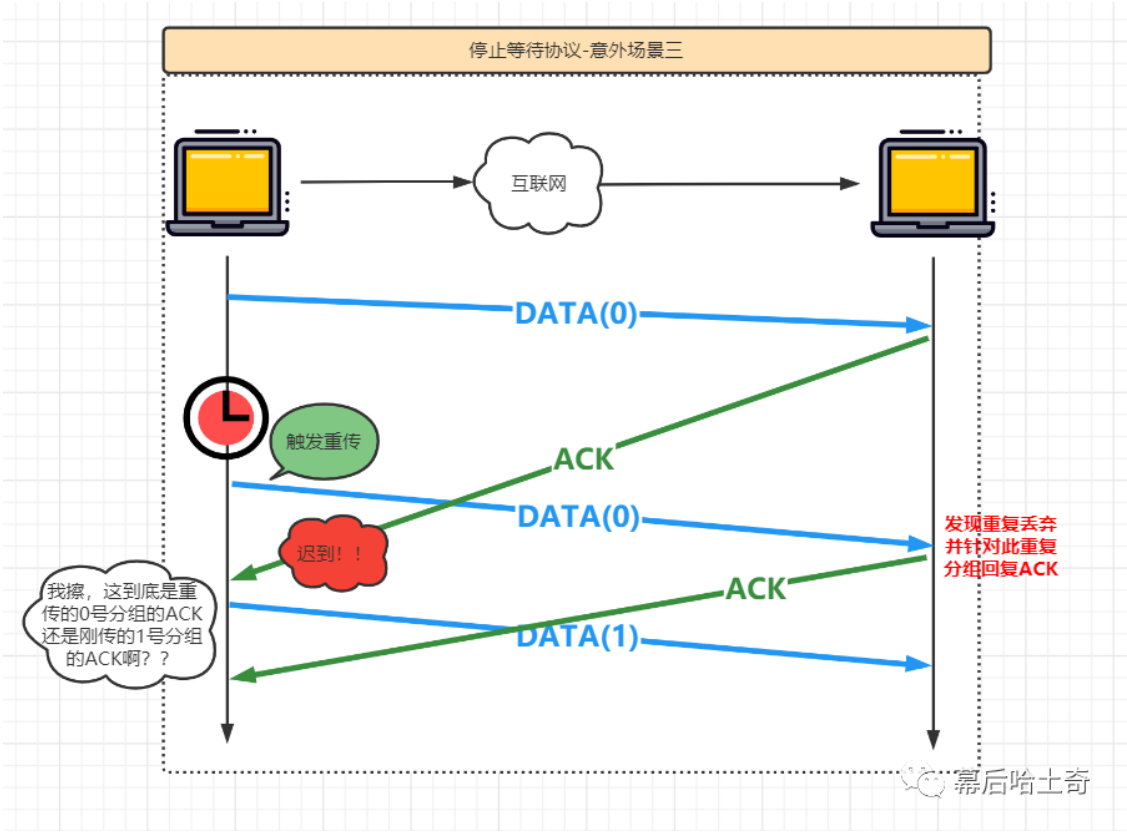
假设出现了下面的情况：

比如主机A发送了0号数据分组，主机B收到0号数据分组后回复ACK，但是ACK由于某种网络原因迟迟没有达到主机A。

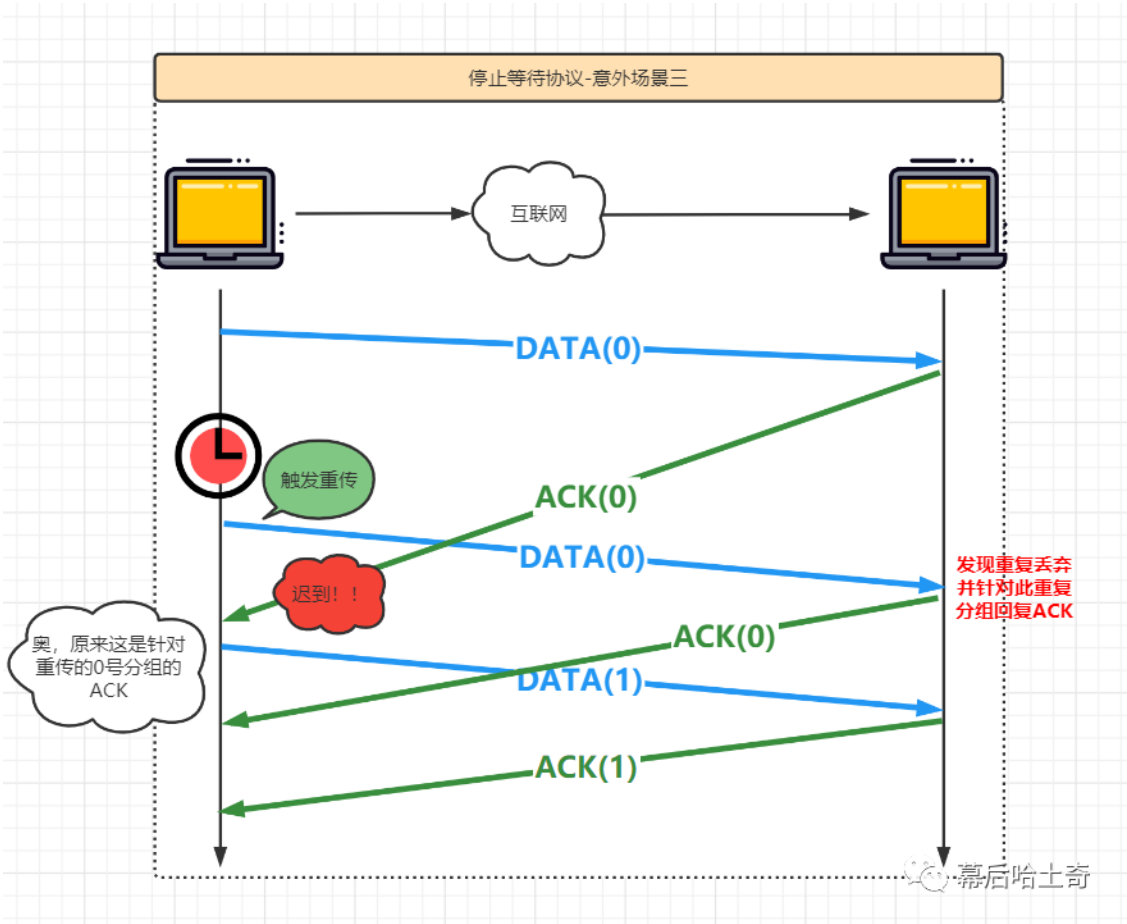
此时主机A超时重传0号数据分组，发送完后，恰好立即收到了第一次发送的0号数据分组的ACK确认分组。

此时主机A误以为是重传的确认分组，立马发送1号数据分组。但是重传0号的数据分组达到主机B后，主机B判断出这是个重复数据分组则丢弃，但为了避免主机A超时重传，仍然回复了ACK确认分组。

此时主机A收到了这个ACK确认分组，却没有办法区分到底是0号重传数据分组的回执还是1号数据分组的回执。



如何解决呢？答案就是给回执分组也进行编号，针对0号数据分组发送0号确认分组，针对1号数据分组发送1号确认分组，当出现以上情况时，就不会混乱了。



可以想一下，在停止等待协议中，由于是停等特性，分组编号只要0或1即可，那么用一个比特即可进行编号了。

四、小结

- 接收端检测到数据分组有误码时，将其丢弃并等待发送方的超时重传，但对于误码率较高的链路，为使发送方尽早重传，也可给发送方发送NAK分组；
- 为了让接收方能够判断所收到的数据分组是否是重复的，需要给数据分组编号，由于停止等待协议的停等特性，只需1个比特编码就够了，即编号0和1；
- 为了让发送方能够判断所收到的ACK分组是否是重复的，需要给ACK分组编号，所用比特数量与数据分组所用比特数量一样，数据链路层一般不会出现ACK分组迟到的情况，因此在数据链路层实现停止等待协议可以不用给ACK分组编号；
- 超时计时器设置的重传时间应仔细选择，一般可将重传时间选为略大于“从发送方到接收方的平均往返时间”
-
    - 数据链路层点对点的往返时间比较确定，重传时间比较好设定
    - 在传输层，由于端到端往返时间非常不确定，设置合适的重传时间有时并不容易

停止等待协议十分简单，发送一个确认一个，但是相应地有一个致命的弱点：**信道利用率太低**。

因为通信的过程中基本都处于等待，信道往往处于空闲状态。当出现超时重传时，信道利用率更低，这对于宝贵的信道资源来说是极其浪费的，**为了提高信道利用率，引出了回退N帧协议和选择重传协议，他们又是如何提高信道利用率的呢?**

本篇文章到此为止，下篇文章继续介绍。