3 停等协议

停等协议，它使用了差错控制，*但是没有流量控制机制*。发送方在某一时刻发送一个分组，并且在发送下一个分组之前等待确认。为了发现被破坏分组，我们需要在每个数据分组中加入校验和。当一个分组到达接收端时，它就被检测。如果校验和不正确，分组就是被破坏的并被悄悄地丢弃。接收方的沉默对发送方来说是一种信号，即那个分组不是被破坏就是丢失了。每当发送方发送一个分组时，它都开启一个计时器。如果在计时器超时之前接收到确认，那么计时器就被关闭并且发送下一个分组（如果它有待发送分组）。如果计时器超时，发送方就认为分组丢失或被破坏，于是重发之前的分组。这意味着在确认到来之前，发送方都需要存储分组的副本。

协议使用序号和确认号来防止重复分组。一个字段被加入分组头部来保存那个分组的序号。一件需要着重考虑的事情就是序号的范围。由于想使分组大小最小化，所以我们寻找能提供无歧义通信的最小的序号范围。让我们来讨论一下所需要的序号范围。假设我们使用 x 作为序号；我们只需要在之后使用 x + 1，不需要 x + 2。

为了表示这种情况，假设发送端已经发送了带有序号 x 的分组。可能发生三件事：

1．分组安全完整地到达接收端；接收方发送一个确认。确认到达发送端，使发送端发送下一个序号为 x + 1 的分组。

2．分组被破坏或未到达接收端；发送方在超时后重新发送分组（序号 x）。接收方返回一个确认。

3．分组安全完整到达接收端；接收方发送一个确认，但是确认被破坏或丢失了。发送方在超时后重传分组（序号 x）。注意，这里分组是重复的。接收方可以认出这个事实，因为它等待分组

x + 1，但是收到了分组 x。

我们可以看到，由于接收方需要区分情况 1 和 3，因此需要序号 x 和 x + 1。但是不需要一个编号为 x + 2 的分组。在情况 1 中，分组可以再次被编号 x，由于分组 x 和 x + 1 被确认，两端都不会产生歧义。在情况 2 和 3 中，新的分组是 x + 1 而不是 x + 2。如果仅仅需要 x 和 x + 1，我们可以令 x = 0 且 x + 1 = 1。这意味着序号是 0、1、0、1、0，等等。这称为模 2 运算。

确认号

由于序号必须适合于数据分组和确认，因此我们使用这种惯例：确认号总声明接收方预

期接收的下一个分组（next packet expected）序号。例如，如果 0 号分组已经安全完整到达，接收方发送一个确认号为 1 的 ACK（意味着 1 号分组是预期接收的下一个分组）。如果 1 号分组已经安全完整到达，接收方发送一个确认号为 0 的 ACK（意味着 0 号分组是预期接收的下一个分组）。

参考 HTTP://BOOK.51CTO.COM/ART/201212/375304.HTM