buffer设计概要

网络库中为什么要有缓存

这个问题需要分发送缓存跟接收缓存分开来讨论。

对于发送缓存有时在发送数据的时候（异步发送）一次发送不完需要发送的数据，但是对于上层的逻辑来说他只管发送，不管其他，此时肯定需要有一个地方能暂存这些数据，等待下次能够发送的时候来发送，另外有时我们有合并小包的需求来降低send系统调用的消耗，这个时候数据被暂时存放在发送缓存中，待到合适时机一并发送。

对于接收缓存，一次调用recv并不一定是一个完整的包，如果此时包不完整，需要暂时放在接收缓存中，等待后续数据， 直到组成一个完成包，投递到上层逻辑。

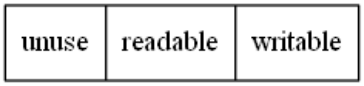
缓存如何设计

这个问题也是需要分发送缓存跟接收缓存的。

对于接收缓存，在投递到上层逻辑的时候尽量结构简单，比如一个指针一个长度就比较好，基于这样的需求，我们的接收缓存是连续的，上面分了3段，用两个字段隔开readpos，writepos，当writepos到达缓存末尾的时候会从新创建一个更大的缓存，并且把未读取的数据拷贝到新的缓存中并修改readpos跟writepos以让未读数据是贴在缓存的开始部分的，在读取数据到达缓存大小一半位置时，表示此时缓存空余非常多，会把可读数据拷贝到缓存开始位置并修改readpos跟writepos以让未读数据是贴在缓存的开始部分的。

对于发送缓存，采用的是多个小缓存拼成缓存的方式，这样读写数据的时候不用移动数据，逻辑层写的时候如果当前小缓存满了，再创建一个小缓存就行了，发送的时候每个子缓存单独发送就行了，这里可以优化的地方就是对于单个缓存，如果先写入了大半，但是还剩余那么几个字节，这样会导致send的时候有时只send几个字节。

接收缓存示意图



发送缓存示意图