RzLib TCP数据包解析算法

一 背景

由于TCP是面向流的协议，在服务端，所有向SOCKET写入的数据，都将直接推到流中，而不会对推送到流中的数据分包，因此分次写入到流中的数据，会根据客户端的读取方式，尽可能的提供给客户端。

如：

在服务端，服务器可能连续推送3笔数据到SOCKET，每笔100个byte。如果客户端一次从SOCKET读300个byte，那么这3笔数据会一次性提供给客户端。如果客户端一次从SOCKET读250个byte，那么第3笔数据会被拆成2部分，后半部分的50个byte仍然未被读走，这时客户端的数据解析可能就会出现困难。

二 组包

在客户端/服务段，数据的接收仍然是以包的形式组织的。我们需要知道该笔数据的用途，用来做不同的操作，同时还要知道该笔数据的长度。

我们可以用一个byte来存储CMD的类型。由于TCP传输的最大长度可以有1500个byte，我们最少需要2个byte来存储有效数据的长度。

数据包的组织大致如下图：



data可能为空，此时Length都为0。

三 RzLib的CMD定义



四 数据包同步算法

为了解决一个TCP数据读出多条CMD或者CMD被拆分的问题，RzLib目前采用如下方式读取数据：

* 读取3个byte，获取CMD的类型以及该笔CMD数据包的大小
* 以该大小再次读取以获取剩余在流中的有效信息