1. 服务器之间内部rpc协议

+-------------------+----------+------------+-----------+---------+

| 4 Bytes | 7 Bit | 1 Bit | 4 Byte | 1 Byte |

+-------------------+----------+------------+-----------+---------+

|<=length of totle=>|<=msgNum=>|<=compress=>|<= msgno =>|<= ttl =>|

+---------+----------+------------------+---------+----------------+

| 2 Byte | 2Byte | 3 Bytes | 4Byte | N Bytes |

+---------+----------+------------------+---------+----------------+

|<=local=>|<=remote=>|<=length of data=>|<==cmd==>|<==== data ====>|

* 1. 支持最多128条msg合并成一个二进制消息输出，设m等于n个msg的总长，当m大于压缩阈值，会对所有内容进行压缩，允许最大发送长度m为uint32最大值（压缩后），编码格式为4Byte长度值+7bit消息数量n+1bit压缩标记
  2. Msgno为自动递增uint32数值，由网关获取生成。其他服务需要发送消息时，这个值为0，经过网关时，网关获取并计算msgno的值，自动递增
  3. Ttl指的是请求的次数，发起请求方将ttl设置为1，如需二次请求其他api接口，ttl不断递增+1，判断ttl值以避免请求形成死循环，如A请求B，B请求A，A又请求B，当ttl值过大就报错，需要对ttl和msgno进行追溯，找到死循环请求api
  4. local由1Byte服务序号+1Byte服务ID号，记录的是请求发起方的信息。
  5. remote由1Byte服务序号+1Byte服务ID号，当ID号为0的时候则进入均衡负载由网关自动分配
  6. cmd定义，在protocol里面编写go格式的struct结构体，由1Byte服务序号+由结构体名字进行crc32生成4Byte数值取前三位Byte，格式为int32
  7. 结构体命名规则 MSG\_服务名\_方法名
  8. 按照结构体从上到下的顺序，进行序列化到2进制data区域
  9. 支持int与uint，8bit到64bit，使用小端littleEndian编码，协议头的length使用uint32
  10. 支持string，将string强转为[]byte后，4Byte长度len()进行int32编码+[]byte消息体
  11. 支持[]byte，规则如string
  12. 支持bool，0=false或1=true
  13. 支持[]int与[]uint系列，4Byte长度len()进行uint32编码+len()个规则c编码，如[]int16{1,2}，将会编码为|02 00 00 00|01 00|02 00|，如[]int32{1,2},将会编码为|02 00 00 00|01 00 00 00|02 00 00 00
  14. 支持protocol内任意其他结构体指针，不支持非指针形式，由1Byte+0或nByte组成，当结构体为nil，第一个Byte为0，不添加其他数据，当结构体不为nil，第一个Byte为1，后续为结构体成员的编码。
  15. 支持[]protocol内任意其他结构体指针，编码格式为4Byte长度len()进行uint32编码+数个上述h格式编码，如[]\*A{nil,nil}，则为|02 00 00 00|00|00|
  16. Float因精度问题不予支持，转为string进行传递
  17. 当消息体所有的数据完成编码后，获取缓冲区长度，允许最大长度为16777216最大值(压缩前)，长度以3Byte写到length of data区，后面紧跟data数据（未压缩）