包结构：uint,ushort看情况

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 包 | 对（datacrc+protoID+dataSize）组成的byte[] 进行crc计算而得到 | 对（数据内容）进行crc计算而得到 | 协议号 | 数据内容的字节长度 | 数据内容 |
| 字段 | headcrc | datacrc | protoID | dataSize | data |
| 类型 | uint | uint | ushort | ushort | byte[] |
| 字节数 | 4 | 4 | 2 | 2 | dataSize |

1. crc校验

问：TCP协议中，底层做了校验，那通信时我们还有必要再进行有CRC或者其他校验吗？

答：tcp 是可靠的, 就是有重传, 校验, 有序等保证. 在字节流层次, 你不用验证了.

但是协议层可靠不代表应用层可靠, 应用层数据校验只能自己做CRC/MD5/SHA1

因为tcp是基于流的,一个报文可能分多个包发送，你自己要要验证报文的完整性

TCP 的校验只能保证物理电路上如果出错, 可以发现并通过重传来修正. 但是对人为的对包恶意的修改是无法校验的。如果是安全要求比较高的地方, 最好还是自己再校验下.

问：为什么选crc ?

答：CRC（Cyclic Redundancy Check，循环冗余校验）

其校验准确度较之普通的奇偶校验、校验和等方法更高，当然计算也略微复杂；

较之MD5、SHA1等算法，CRC安全性和准确度方面又略显不足，

但计算较之这两者明显简单，效率更高。

所以如果仅仅针对网络数据的一致性校验，即收发端数据的是否一致

（因为在Socket编程里，单次收到数据的长度和发送数据的长度即使在阻塞模式下，也不一定是相同的，这个依赖于网络环境，虽然TCP协议保证了数据的完整性和一致性，但像人为对数据进行了分片的这种情况，在收到数据时视情况还是有必要进行一下校验），

针对这种校验要求，CRC32是明显足够，也不会带来很大的计算负担。

2.加密，解密

介绍：

不可逆加密：md5 ，sha1，加密后就不能解密，只能用于存储密码和校验文件变动，不能用网络通讯

可逆对称加密**：DES，3DES，AES**

可逆非对称加密**： RSA**

选定：对称加密：DES 非对称加密：RSA

.net有封装好的，DESCryptoServiceProvider, RSACryptoServiceProvider

流程：

**一般流程为：先用非对称加密去加密对称加密的密钥(对称加密的密钥比较短，可视为不怎么耗时) ，然后再用对称加密去加密数据。**

也就是说：

客户端用RSA生成公钥和私钥，发送公钥给服务器，服务器用收的公钥对3DES的key进行加密后，发还给客户端，客户端用私钥进行解密得到3DES的key，之后就可以用此key来进行加密解密。

3.数据内容形式：json, protobuf-net, protobuf-java, pbc, pb-lua-gen, sproto

选定:Client：pbc Server: protobuf-java (我的服务器是java写的)