核心问题：

1. 加密，crc校验等
2. 字节对齐，字节摆放的顺序
3. 试图设计“完美”的协议
4. 没有消息序号，或者消息序列号只有2个字节，或者没有清晰缠绵如何编号。
5. 代码实现的位置。

头和体可以分开发送吗？

func()

{

send(fd, head);

xxxxoooo

send(fd, body);

}

时刻1

T1：

send(fd, head);

时刻2

T2:

send(fd, head);

xxxxoooo

send(fd, body);

时刻3：

T1 :

xxxxooo;

send(body)

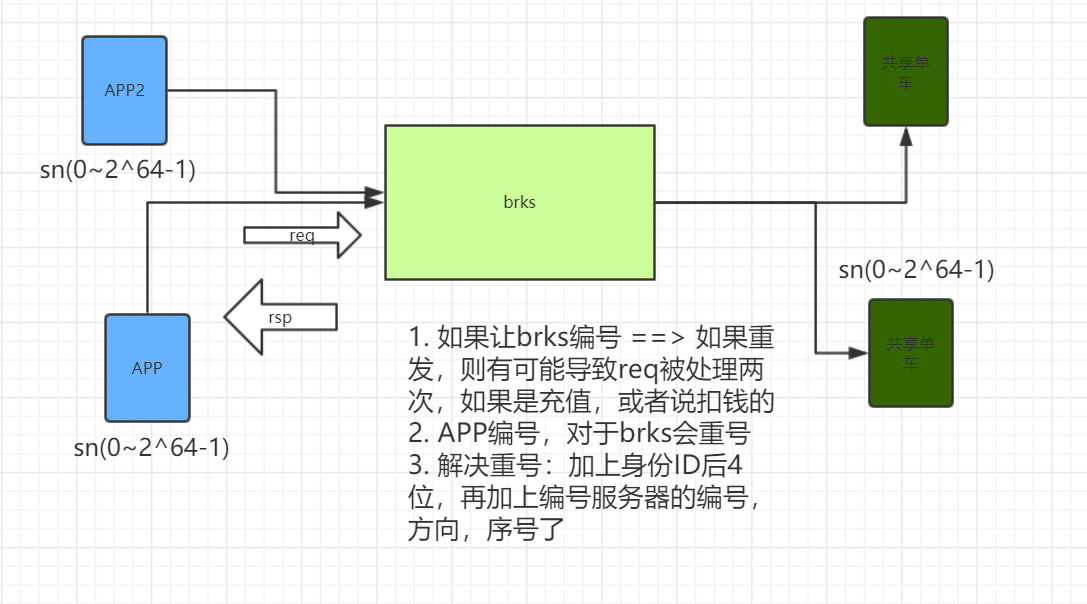
head1

head2

body2

body1

消息ID设计：



TCP粘包

msg1 m | sg2 msg3

Nagel : 40ms,

1. 如果是一个小包（ < MSS ），则等待40ms
2. 如果在40ms中间，有其他的包要发送，那就组包，mtu
3. 如果到了40ms，没有其他的包，则超时后，发送msg1

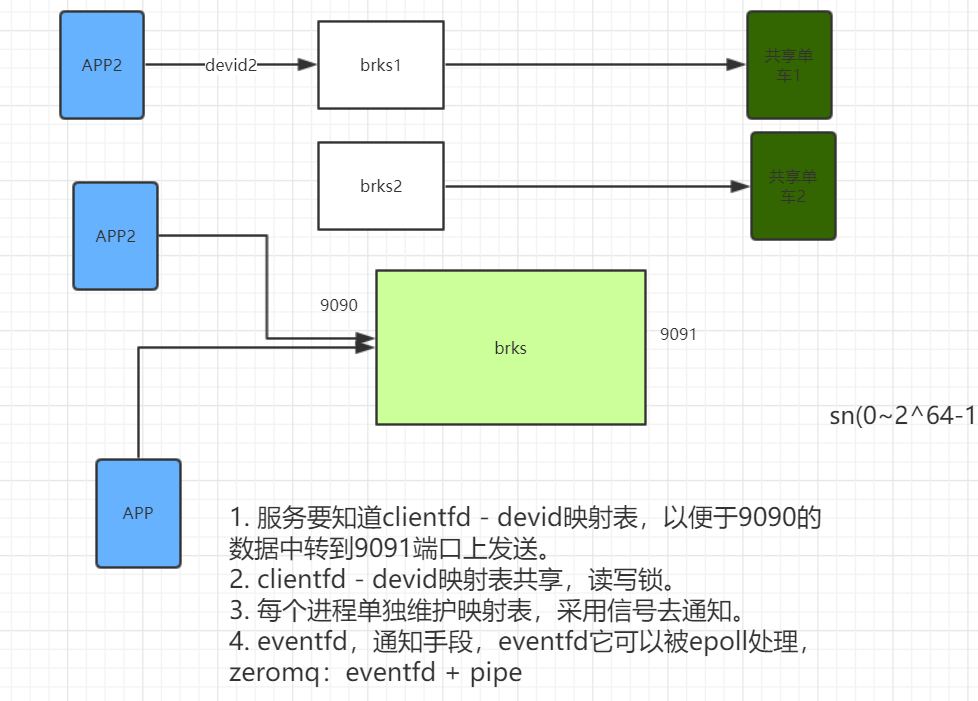
为什么消息设计成了一请求一应答的形式：

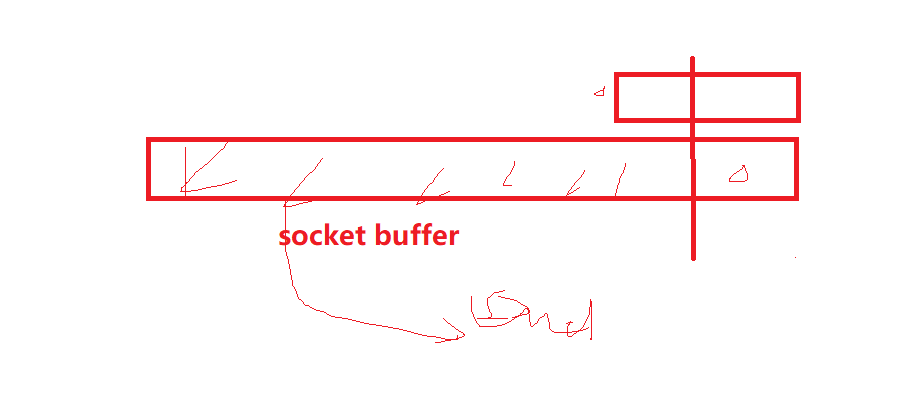
UDP如何做到“可靠”传输，TCP重发，流量控制。

内存🡪指的是内核的buffer，但是我们是做业务开发的。“可靠”的意思是接收端已经正确的处理了，或者出现了异常，但是这个异常是可以通知到发送端的。

nio\_write

思考题：





接下来的设计：

# protobuf：

1. 轻量，编码后整个消息所耗的内存空间比较小。
2. 高效，效率很高。
3. 编译安装protobuf
4. 定义proto文件
5. 编译proto文件，生成目标代码：
6. 把生成的代码文件引入到工程，SerializeToOstream，

<https://developers.google.com/protocol-buffers/docs/reference/cpp/google.protobuf.message>

<https://developers.google.com/protocol-buffers/docs/cpptutorial>

那我们改变消息结构还需要重新替换生成的protoBuffer 文件？这样有点不方便啊：

1. Extend proto definition

A 定义msg

B extend msg{

required string xxxooo = 126;

}

我们之前在华为，09年没有protobuf，也没有用json，那我们当时的协议怎么定义的？

TLV：

T : type，int uint char int64

L : length，int16->length,

V : value, 0xFBFC

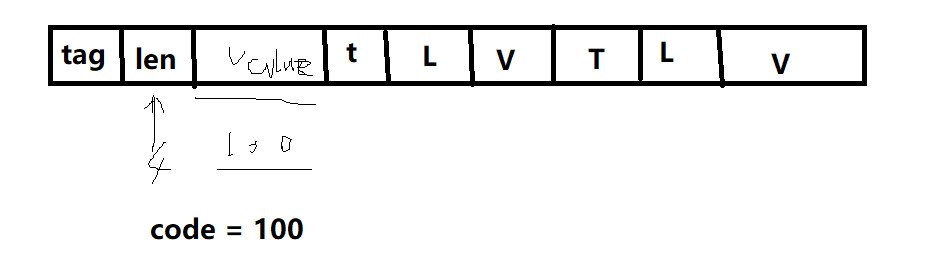
char\* ; 不定长的，

TLV:

L : 4个字节，表示接下来的内容长度。

V： buffer + 4; **-> content**

protobuf -> TLV:



int32 -> 4 , value = 100, 1个字节 BYTE（< 256）

1110 0100, 1xxxxxx00000, 0xxxoooo

传数组：

a[1] : T L V

a[2] : T L V

a[3] : T L V

T L V L V L V L V

67 行，

里原始需求分析（OR）， 到立项， 到交付（客户现场用起来）。 67行 / 人均。

代码： review UT MT BBIT ST，

200 行 review 0 UT MT BBIT ST zore bug

管理方式 和团队文化

杭州研发中心 成都团队

团队文化 ：

Nokia ： 内敛 静下来 宁静致远

WCDMA ：

feature：需求 HZ -> Test -> git

pronto：bug CD 线上测试 / ST