目前包含的协议有：

baidu\_std

streaming\_rpc

hulu\_pbrpc

sofa\_pbrpc

rtmp

http

public\_pbrpc

nova\_pbrpc

nshead

mongo

ubrpc\_compack

redis

memcache

nshead\_mcpack

ubrpc\_mcpack2

esp

增加一个新协议

// 3 steps to add a new Protocol:

// Step1: Add a new ProtocolType in src/brpc/options.proto

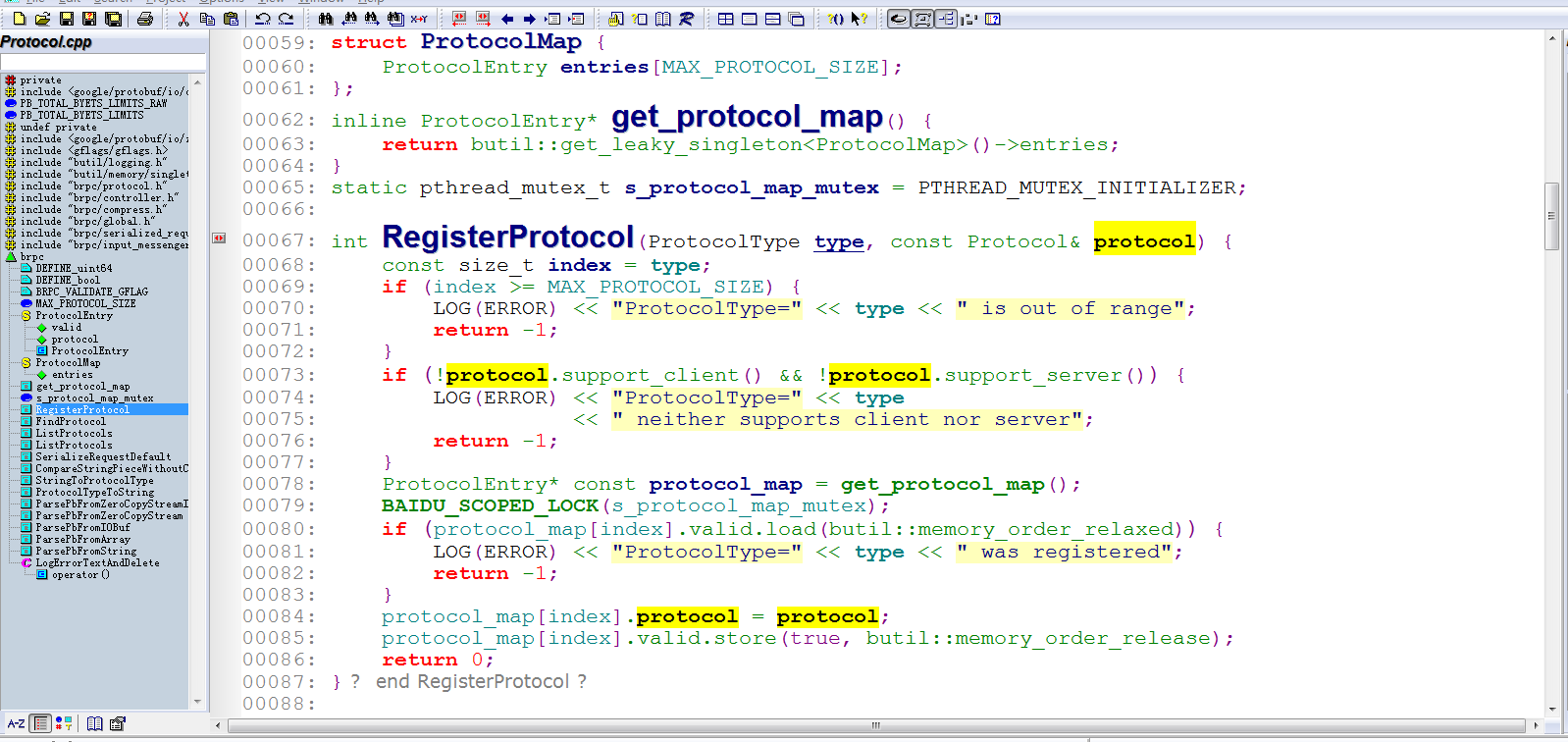
// as identifier of the Protocol.

// Step2: Implement callbacks of struct `Protocol' in policy/ directory.

// Step3: Register the protocol in global.cpp using `RegisterProtocol'

代码

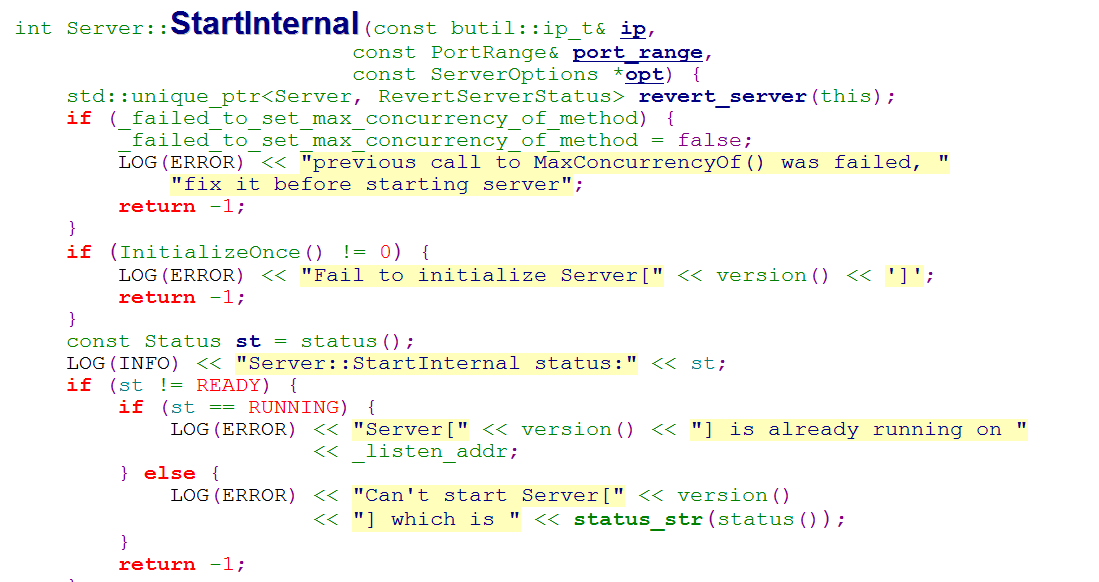
G:\开源\brpc\brpc-master\brpc-master\src\brpc\protocol.cpp

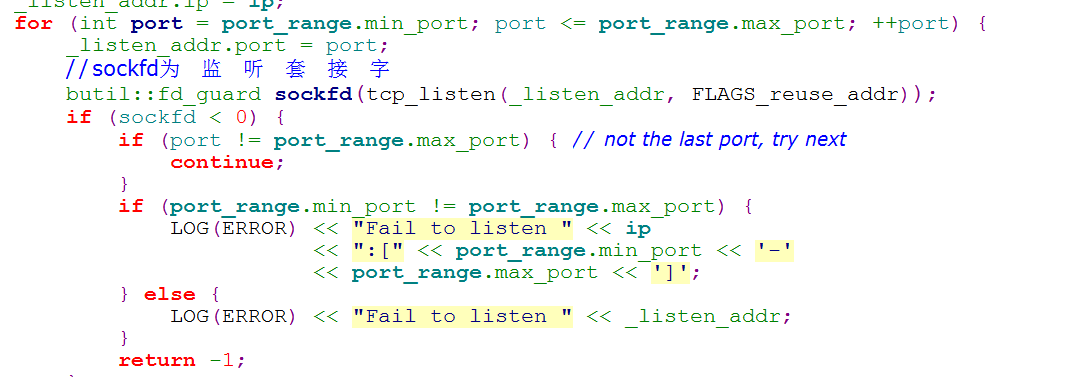


通信流程：

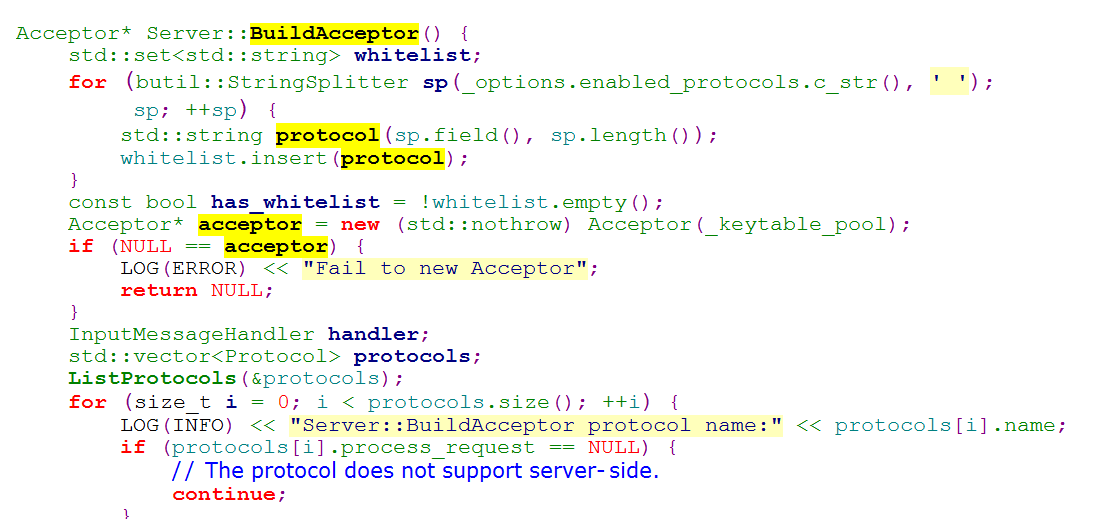
G:\开源\brpc\brpc-master\brpc-master\src\brpc\server.cpp

服务端开始监听listen，在指定的IP地址和端口绑定的套接字上

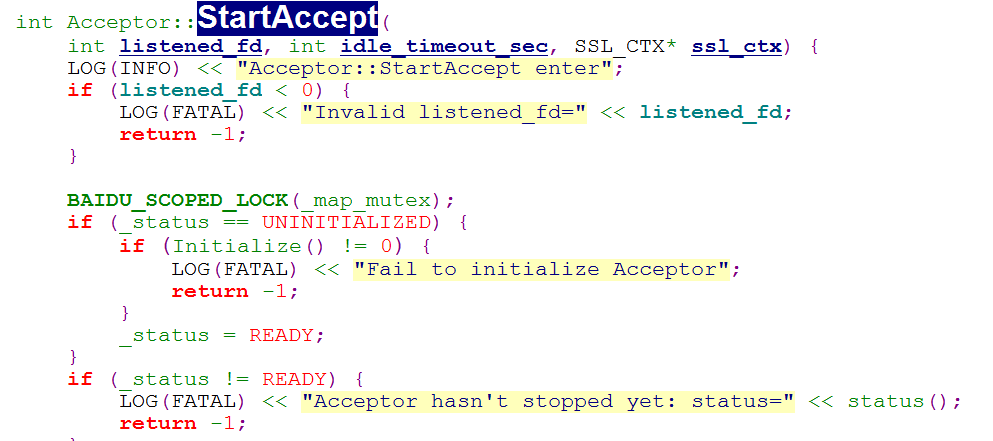


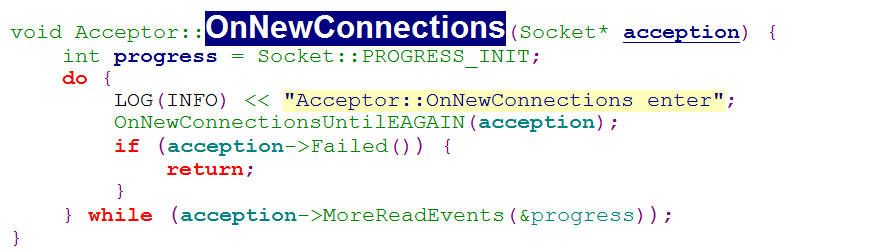


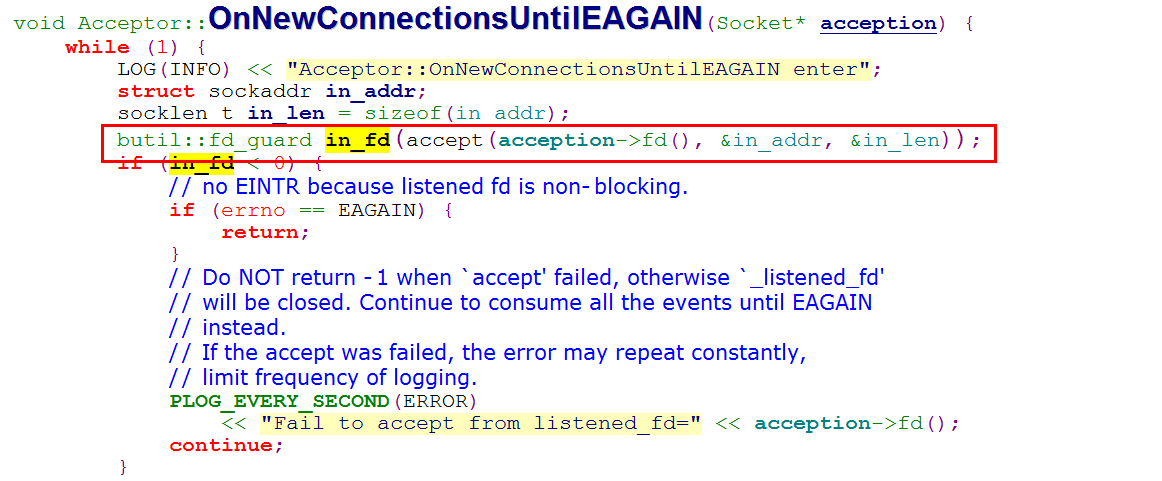
将指定协议的处理消息的函数加入到接收器中Accept



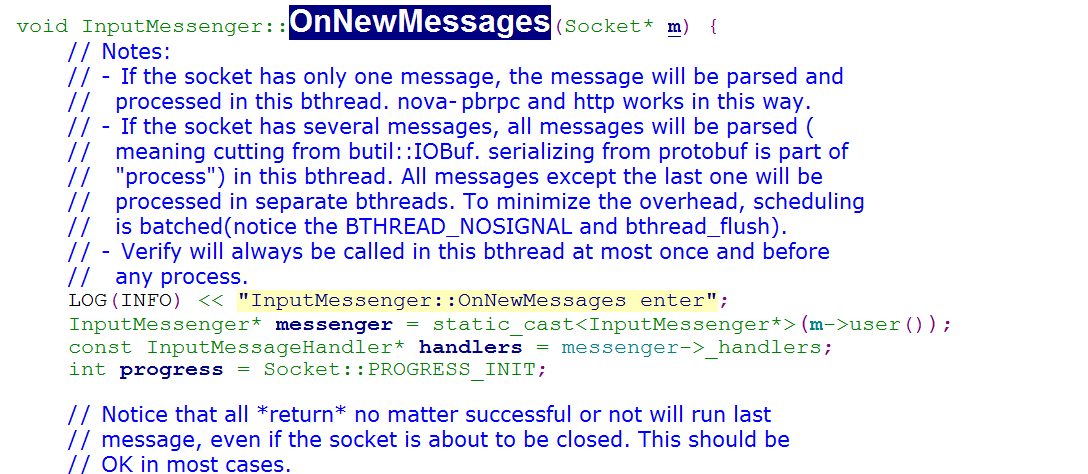
开启死循环等待客户端连接的套接字accept



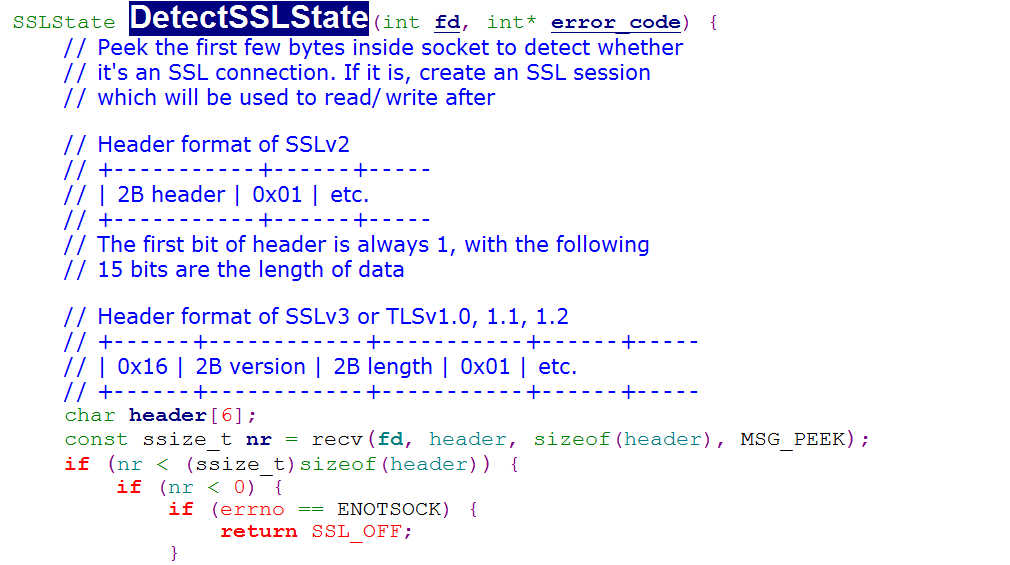




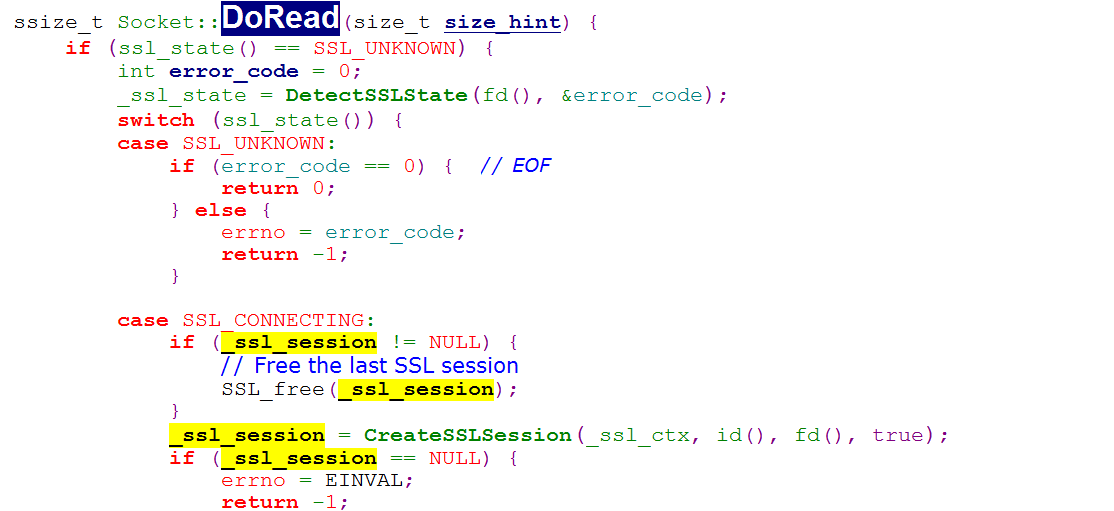
接受到客户端的套接字后，调用客户端的套接字来接受数据，处理数据

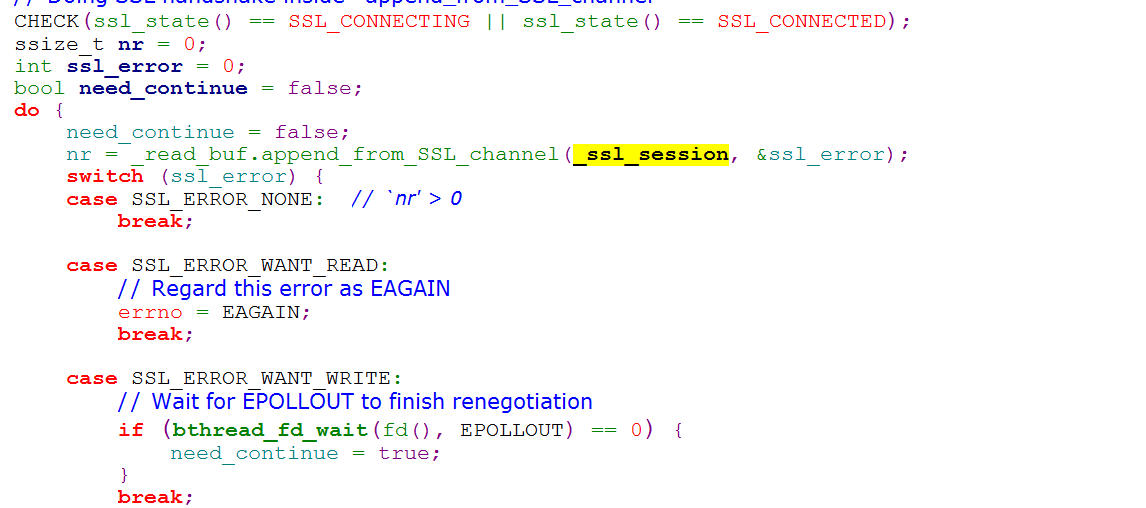


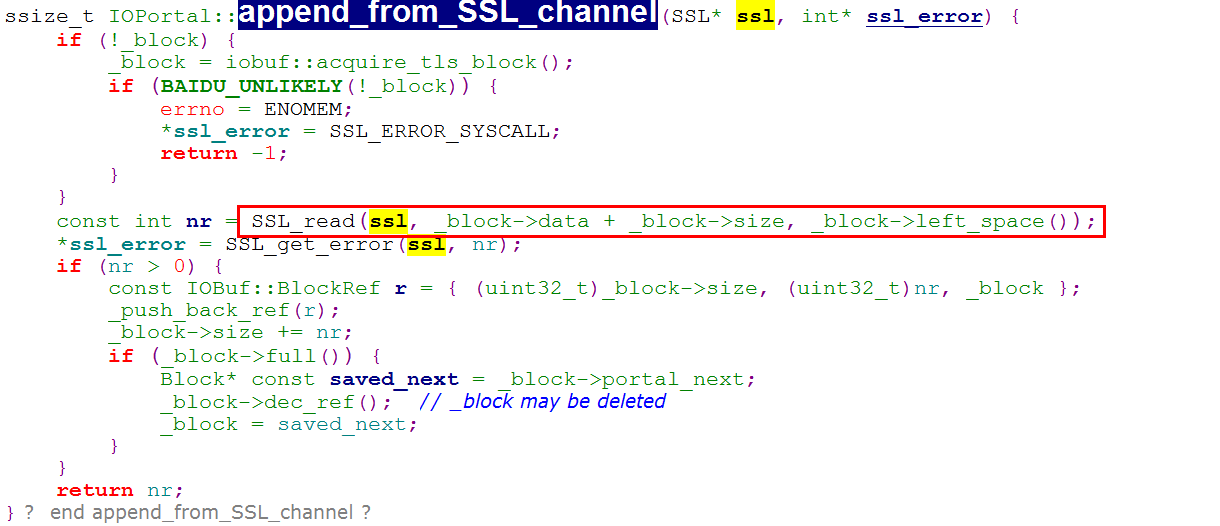
首先如果是第一次建立连接，首先读取SSL头，判断是否是要建立连接，若是，创建SSL对象

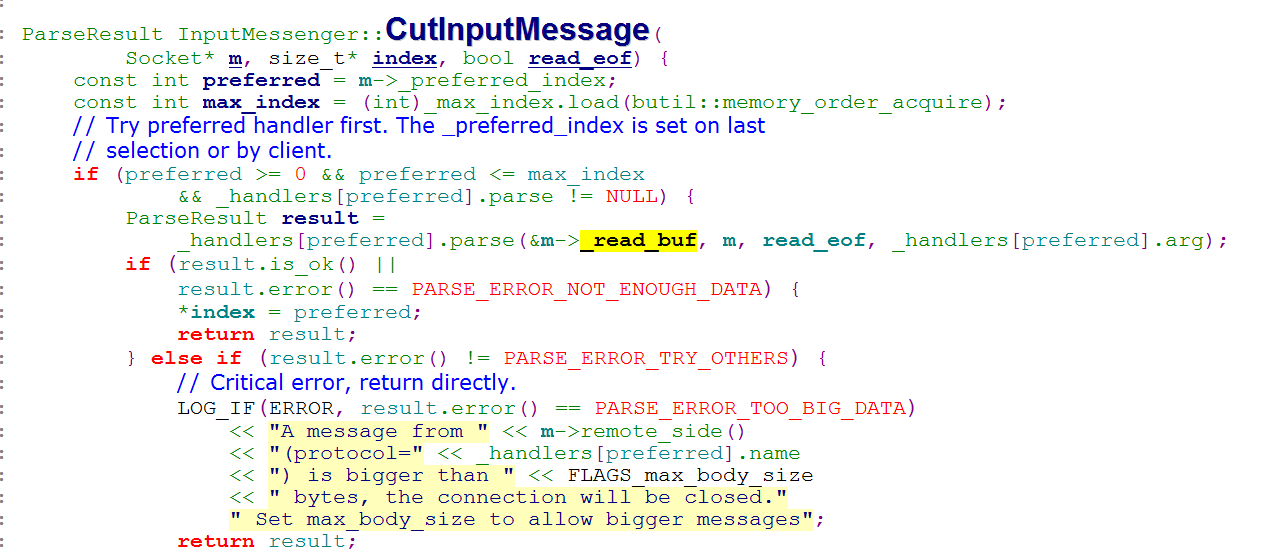


通过SSL对象读取客户端发过来的字符串，放入\_read\_buf



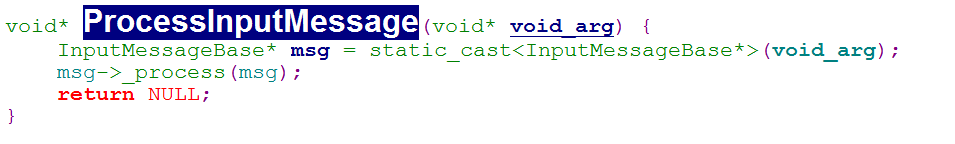


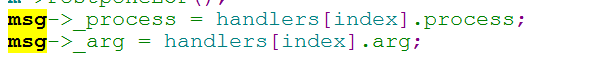
调用协议体的处理函数，解析字符串

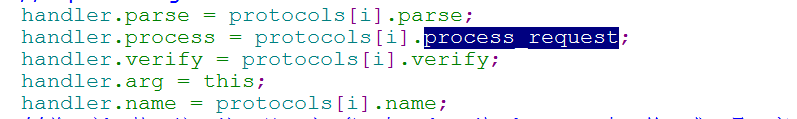


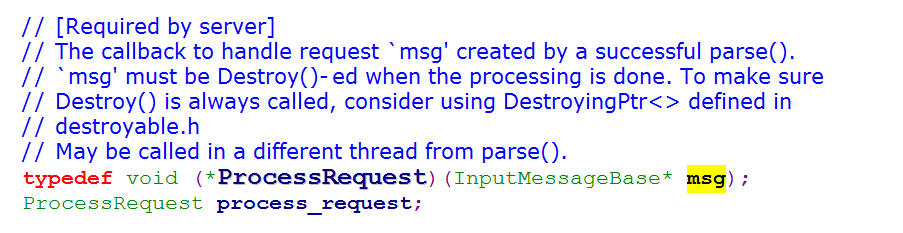
将解析后字符串加入到缓存队列中，开辟线程，调用协议体的处理函数指针





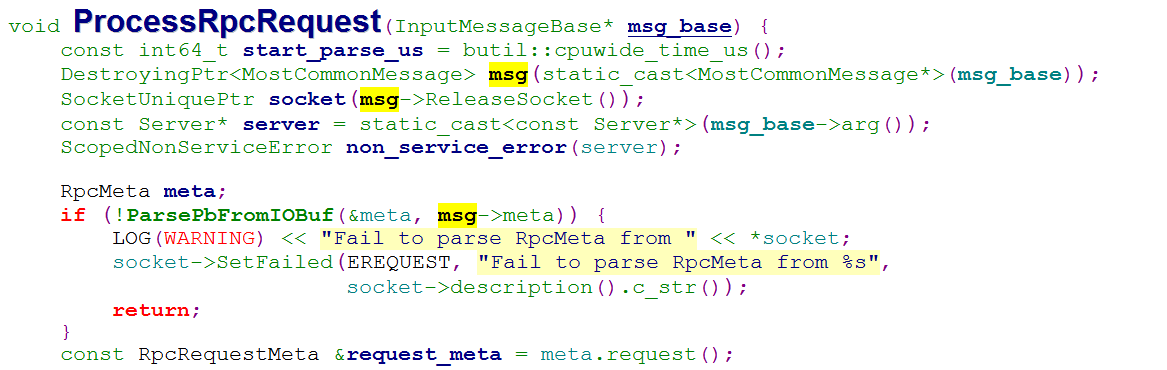


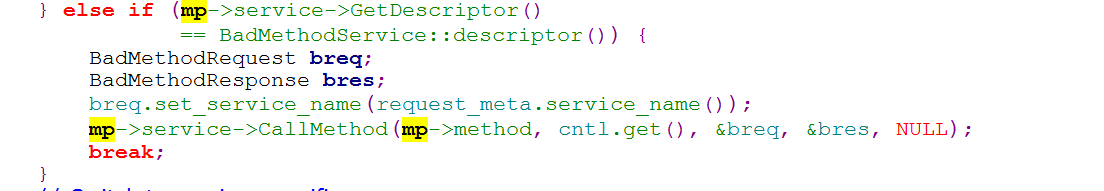




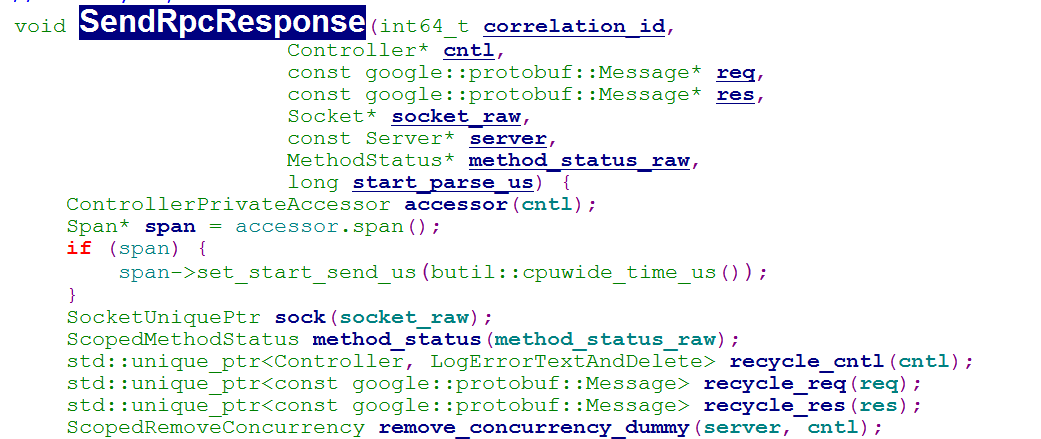
下面以baidu\_rpc为例进行说明

调用policy中的baidu\_rpc协议函数，该函数解析客户端字符串到调用的服务器对象中的函数。





将函数执行结果返回给客户端



客户端数据处理过程：

request 函数 SerializeRequest ——》request\_buf 函数PackRequest——》iobuf\_out/message

套接字发送

接收到message 函数ProcessResponse——》响应信息

服务端数据处理过程：

IOBuf 函数Parse ——》Msg 函数ProcessRequest

