1. 前言：

本项目是针对于Gowild Ho 型号的研究，并不兼容HE。

关于琥珀虚颜的复活我有几点要说，它的研究历程一共经历了以下：

1. 搞机佬

通过USB，连接设备可以提取文件和安装应用，也可以远程操控当手机用。

* ADB（一种安卓调试工具）提取软件资源、得到数据文件，将资源到别的地方使用
* 刷机，当手机用或进行一些功能性更改比如播放其他设备音频
* 逆向APK文件从而进行解析或生成新的软件安装到别的系统上

1. 硬件流

该分支需破坏外壳，对底层硬件如电路板就行研究和改造。

* 研究硬件上预留的伏笔和基础结构
* 进行硬件上的蓝牙音箱的改造

1. 服务器流

这也是我的出发点，将设备请求拦截下来重定向到本地服务器而实现再次使用，通过反编译的客户端代码去反推服务端响应，是当前主流的复活方向。

* DNS请求的拦截和重定向
* 本地服务器的建立
* 加密方式的破解
* Protobuf2数据的制作
* TTS（文字转语音）STT(语音转文字)

如果你也对琥珀感兴趣，想要出一份力，接下来我将会为你介绍他的客户端流程，你需要掌握基础编程知识，当然你如果不会编程也可以来测试。

Ho的软件是个缝合怪，人脸识别是百度的，文字转语音是标贝的，语音识别是讯飞的，一堆厂商的服务拼凑起来的，本身有一个TTS（文字转语音）但还是走了网络。

一些杂谈

我接触到琥珀是在2018年，触碰到第一台琥珀则是2024年，在我之前就有致力于破解和修复的人，可以把修复的人分为两个流派，也就是上面提到的那些人，可以简单的理解成懂代码和不懂代码的，搞机佬是最多也是最快对其进行研究的。

因为出发点的不一样和他们资料的交接做的不怎样，导致我浪费了大量的时间去理解他们在做什么，当我得到ho后，我的研究终于能正式推进。

首先我使用抓包软件它的网络请求抓数据包，得到服务器域名后自己搭一台DNS服务器，将指定域名解析成我的计算机，然后我就得到了传过来的二进制码。

随着我的不断推进，出现了一个大问题，那就是它服务器目前是关闭状态，我没有之前正确的响应数据，我不知道要怎么回，要继续向后推进的可能极其渺茫，本来以为代理服务器流至此也是终了，但是之前提取apk反编译的那些人里有一位大佬，从逆向出的程序代码中找到了响应信息的源码，在此之前我也没想到会有加密，在我提供了请求数据之后，他成功解开了数据的加密，并建立了持续的服务器连接。

到这里，琥珀复活终于进入正轨。

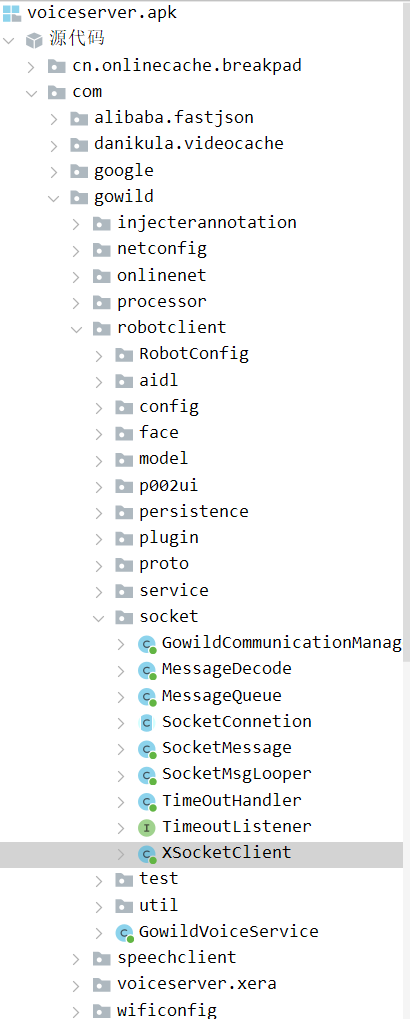
首先来个入门代码（以下代码你不需要实现，看看即可），数据的传输是加密的，通过这样的算法来保证数据的一致性：

**public** **static** **byte**[] encode(**byte**[] plainText, **int**[] encryptKey) {  
 **int** length = plainText.length;  
 **int** lastCipherByte = (**byte**) ((plainText[0] ^ encryptKey[0]) & 255);  
 plainText[0] = (**byte**) lastCipherByte;  
 encryptKey[1] = (encryptKey[1] ^ lastCipherByte) & 255;  
 **int** lastCipherByte2 = (((plainText[1] ^ encryptKey[1]) & 255) + lastCipherByte) & 255;  
 plainText[1] = (**byte**) lastCipherByte2;  
 **int** keyLen = encryptKey.length;  
 **int** i = 4;  
 **int** j = 0;  
 **while** (i < length) {  
 **if** (j == keyLen) {  
 j = 0;  
 }  
 encryptKey[j] = ((encryptKey[j] + lastCipherByte2) ^ i) & 255;  
 lastCipherByte2 = (((plainText[i] ^ encryptKey[j]) & 255) + lastCipherByte2) & 255;  
 plainText[i] = (**byte**) lastCipherByte2;  
 i++;  
 j++;  
 }  
 **return** plainText;  
 }

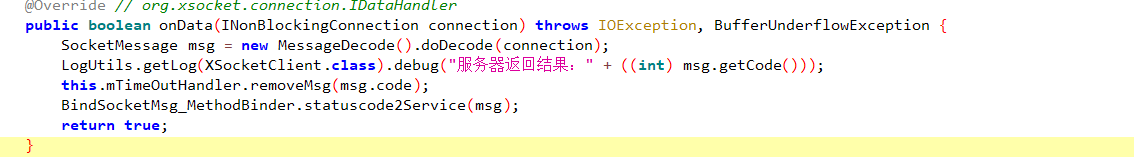
在我拦截它发的请求时我发现，连上wifi它就请求了一下百度域名，然后请求rbtapi.gowildholoera.com，因为狗尾草倒闭所以它找不到ip，然后就会请求router.gowild.info，如果还找不到还有一个域名的请求，设备是这个我直接全重定向到自己主机，安卓端软件也是走同样的服务器的，所以可以用本服务器进入手机app主页面，域名是app.gowildholoera.com、appapi.gowildholoera.com、appapi.gowildholoera.com.mshome.net

客户端

客户端使用XSocket框架构建，你可以从com.gowild.robotclient.socket.XSocketClient这个位置开始看，

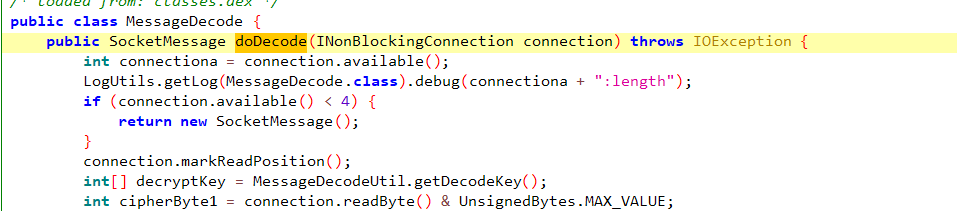


这里得到服务器响应，它将所有数据装成SocketMessage



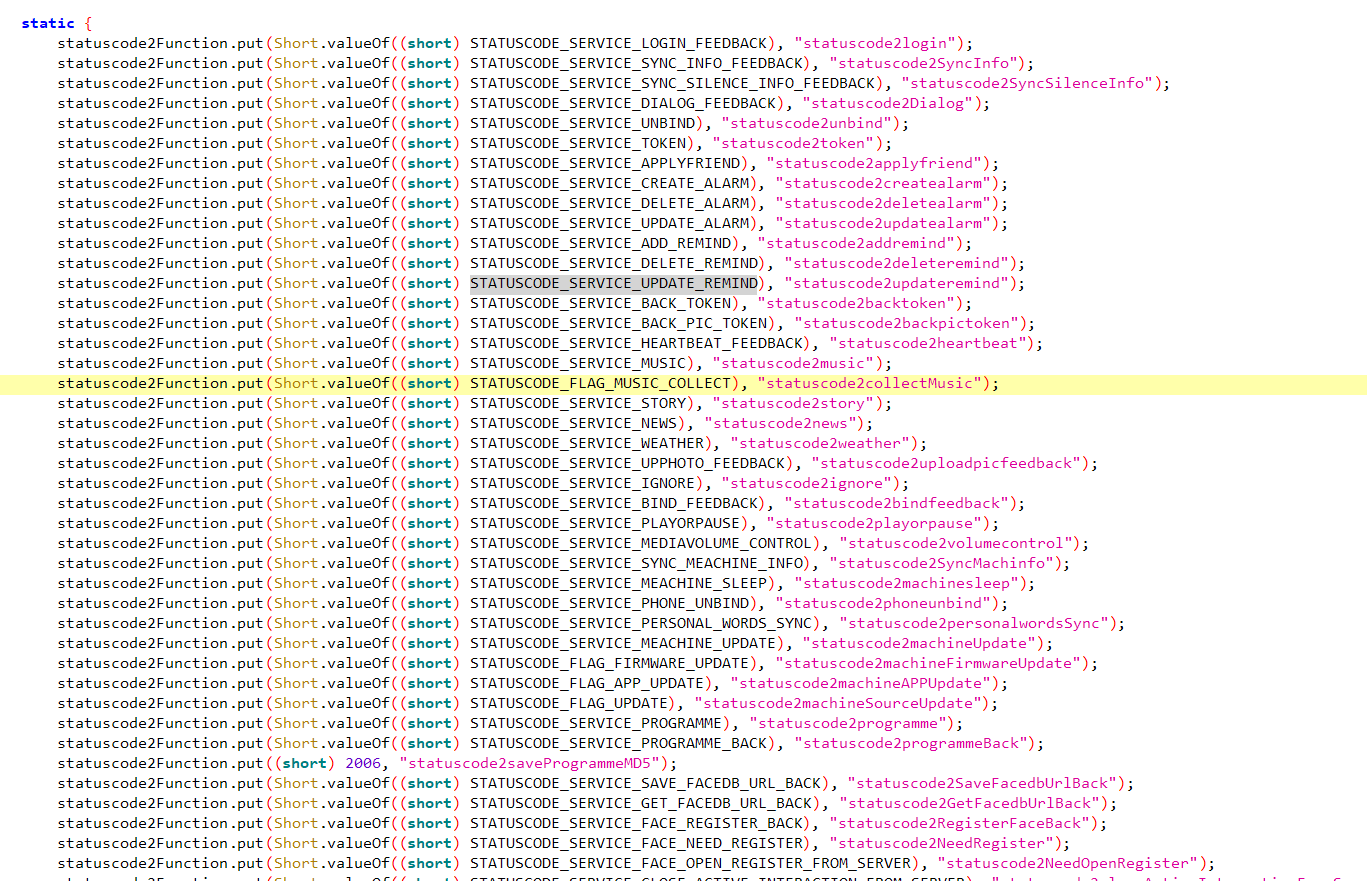
INonBlockingConnection是xsocket中的一个接口，这里你可以理解成一个二进制输入流

得到后先对头进行验证，协议格式在配置项文件中已做说明，就是7ffe[长度][状态码][结束位][data]（二进制），然后读出头指定长度的字节，进行解包。



然后就是关键，读到的状态码判断protobuf数据转为对应的业务类

这个很长，在com.gowild.robotclient.service.GowildStatus类中我们可以看到



这一堆都是它的业务，复活服务器就是去推这些响应。

注：我找到最贴近它的protobuf工具版本是protobuf-java-2.5.0.jar

Jadx按ctrl+鼠标右键快速跳转

com.gowild.speechclient.SpeechPlugin

这里是语音的处理输入和输出你都能在这个包中找到影子。

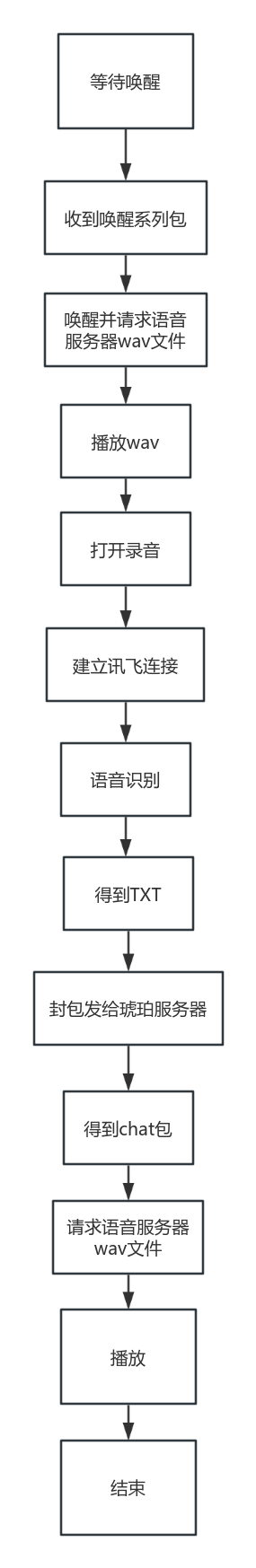
235是更新服务器地址请求，我们通过246响应码更新了它的标贝TTS服务器地址，protobuf相关的知识自行连接，向它发请求就是用protobuf2工具把你想发的信息做一个二进制数组，然后接上数据头，protobuf的工具在java中生成的是java类，它对应的格式你其实并不需要自己准备，在**package** com.gowild.robotclient.proto;

这个包中你可以找到已生成完毕的所有java类，拷贝出来build一下就可以了，如果你使用的是非java的编程语言请自行生成。

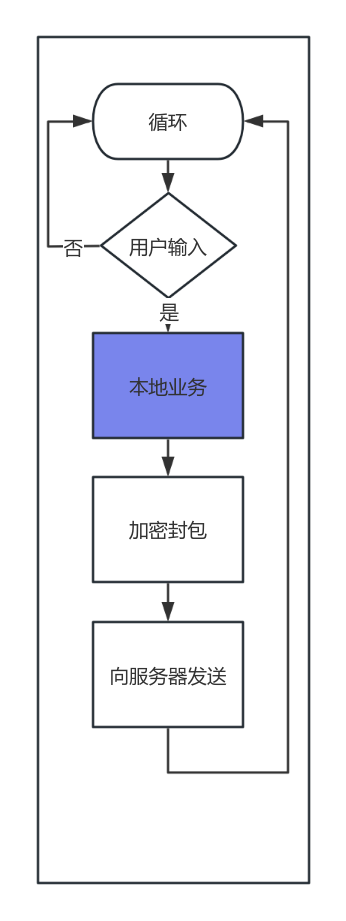
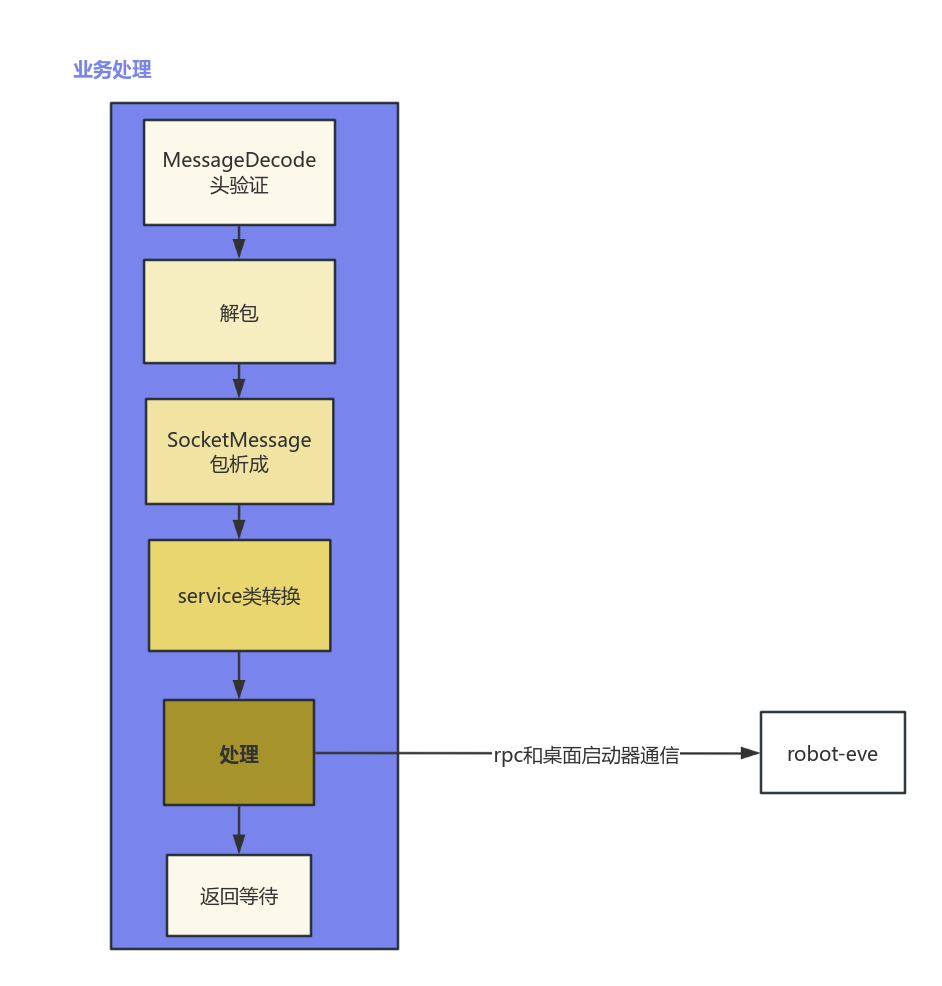
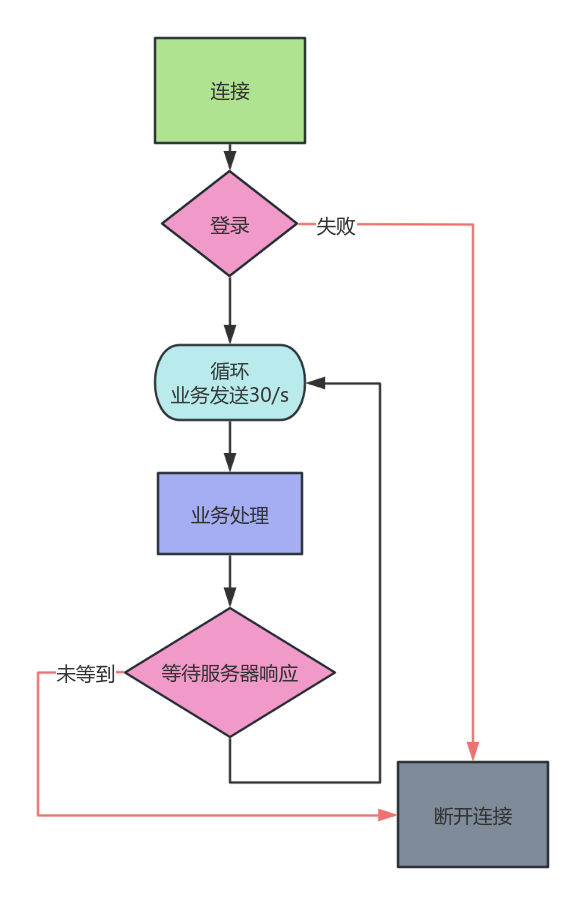
有大佬已经完成了它的TTS，他们从逆向的apk文件中提取并还原了本地的语言功能，并成功合成了语音，最后搭建一个语音服务器，将标贝语音服务器地址更新成本地搭建的语音服务器地址，我们就能成功让她说话，211状态码是她的唤醒，这个包发了后会主动请求一个语音。而tts服务器会返回一个wav，记得添加响应头"application/octet-stream";

前面的工作我们已经做了，你只需要在我们的基础上进行即可，目前就是搞语音识别，也请你从这里切入，问题很大主要看SpeechPlugin、SpeechHandler

以下是我估测的聊天流程



Ps:1.聊天的业务类就一个——ChatService

1. 

大致是这样吧，实际上循环是异步的，有心跳每隔最多最多30秒会发一个心跳包给服务器或一个当前电量的包，服务器也要一回个心跳包。

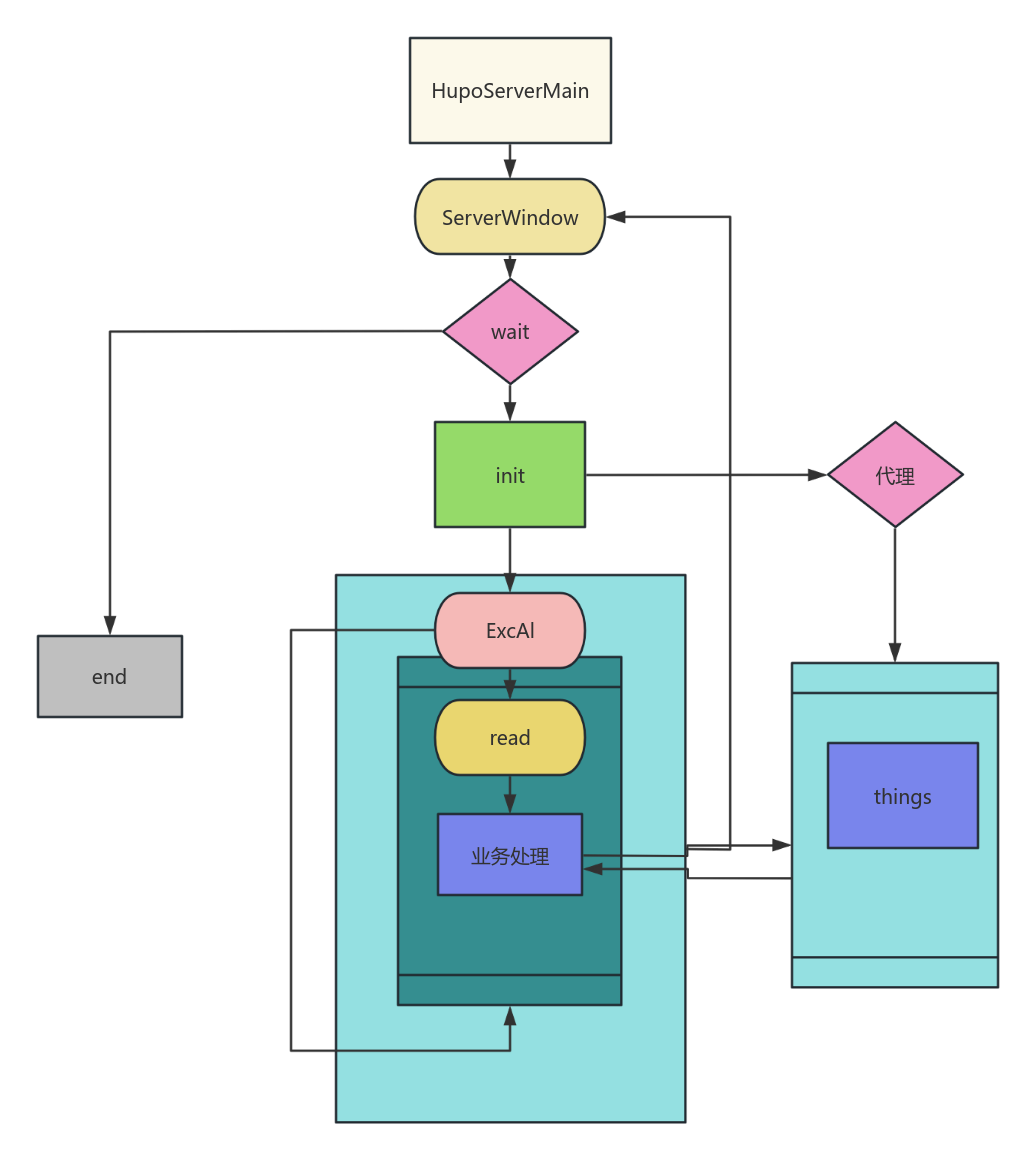
3.211是用户唤醒状态码，用户按中间的按钮或说琥珀琥珀（唤醒词是本地的），要发两个响应过去，应该是状态唤醒，另一个使网络稳定。

1. 登录二维码格式为  
   “{“accound”:”用户账号/手机号”,”ssid”:”wifi名”,”pwd”:”密码”}”

虽然我写的不怎样，凑合看吧，好过没有。

接下来帮助你理解我的服务器架构。

Wm\_HupoServer



从入口HupoServerMain创建窗口线程ServerWindow,等待用户启动，ServerWindow中isdebug变量为true时会将接收到的数据和解密数据保存到线程名件中（自动建），启动后会进行初始化VerProper类，它会读取同路径下response.ini文件，里面是针对不同状态码的响应和服务器的一些配置，如何配置请看我的.ini配置文档这里就不再多做讲述了。

Proxy类负责代理服务器的事件，就是与对应服务器建立连接然后把数据推过去，关于这个类我的写法有的奇葩，我希望简易创建的同时又能实现复用，它是一开始就被创建的，当用户把客户端推过去时再去请求连接，只支持一个连接，多了解密的key也会出问题

当启动后，如果你配置了代理服务器，右击在线用户列表，可以看到一个用代理服务器转发选项，会将数据解密后再重新开始加密向代理的服务器发过去，我称为半代理，如果服务器本读取或解密出现问题代理的服务器解密也会出错。

启动初始化完成后会创建一个ExcAI类，它会阻塞的一直在等待Socket连接，得到后会为其创建一个线程Excagent放到ServerWindow类中的Vector<Excagent> onlineList 同时会获取socket的ip创建一个字符串放进JList<String> Useronlise更新在线用户列表（连接断了也会同步更新）

每一个用户有一个对应的用于解密的UserKey，业务流程开始后会调用一次之前创建的VerProper 中的ttime这个方法，这个方法为其创建一个线程，这个线程用来处理需要定时重复发送的业务，如服务器心跳包。

然后就是循环的读数据，如果有读到数据就把数据接收到缓冲区，然后用工具验证头是否为0x7FFE,没问题就读这一个包出来，解密，然后判断需要并回复。

总体来说逻辑比较简单，希望能够对你有帮助。

Ps：protobuf的类java这么用

SendMsgForXXBSMsgProto.SendMsgForXXBSMsg.Builder build = SendMsgForXXBSMsgProto.SendMsgForXXBSMsg.*newBuilder*();

build.setText("123123");

build.setFunType("news");

SendMsgForXXBSMsg smg = build.build();

System.***out***.println(smg.toByteArray());