CentOS release 6.7 (Final)

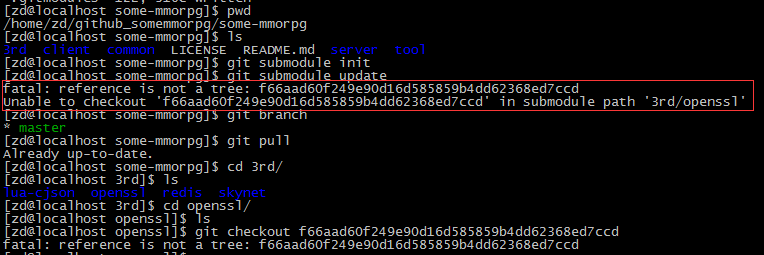
当前目标是将some-mmorpg项目改造为简单的挂机游戏项目，换句话说是将当前的挂机游戏项目移植到skynet上，但前期只选取部分简单的功能模块，比如玩家基本信息，背包，商店等。

2019.04.08

通过 git clone git@github.com:AddisKMud/some-mmorpg.git --recursive

递归检出项目下所有目录

实际只顺利检出redis和skynet目录下的代码，openssl目录检出失败，提示如图：



最后只能采取之前分别clone依赖项目目录的方法。

--------------------------------------------------------------------------------------------------------------------------------

查看3rd目录下的openssl,redis以及skynet下的代码是否为特定的旧版本，如果旧版本代码，为了兼容之前项目中已有的代码，暂不更新代码。

--------------------------------------------------------------------------------------------------------------------------------

从metoo项目的protocol/protogenpb.sh中借鉴从.proto生成.pb的脚本，并做了少量修改，兼容目录下有其他同名但后缀名不同的文件

2018.12.11 00:21

通过git clone检出 some-mmorpg项目

发现通过git submodule update --init命令无法检出some-mmorpg/3rd下的openssl,redis以及skynet目录，通过手动执行git clone命令分别检出openssl,redis以及skynet对应的项目目录

准备将sproto转为对应的protobuf，目前只转了login\_proto.lua为login.proto

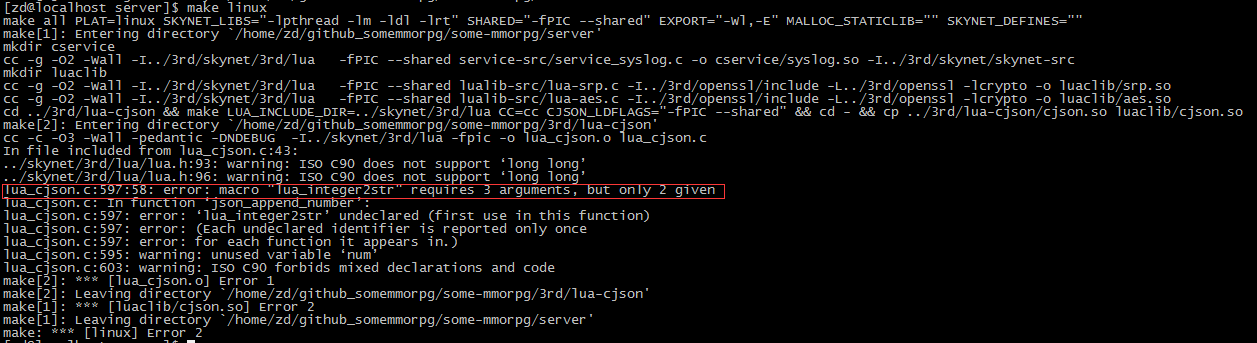
2018.12.13 22:44

使用命令 ./Configure linux-x86\_64 no-shared -fPIC; make 编译3rd/openssl

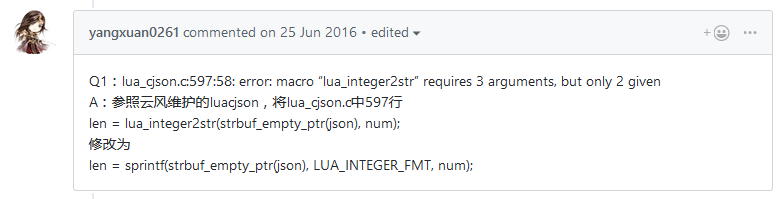
使用make linux 编译3rd/skynet

在server目录下使用make linux进行编译

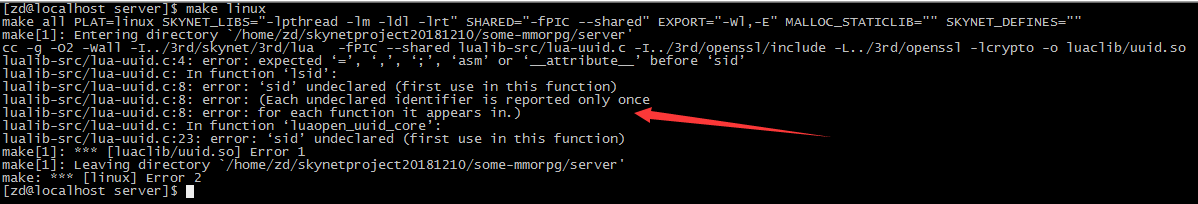
编译lua-cjson目录报错，报错如图：



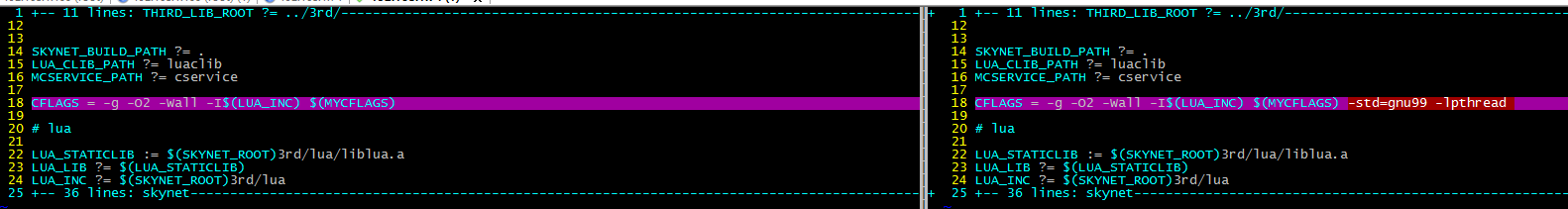
解决方法在some-mmorpg的原作者的Issues #2下，链接：https://github.com/jintiao/some-mmorpg/issues/2，解决方法如图：



编译lualib-src/lua-uuid.c时报错，报错如下:

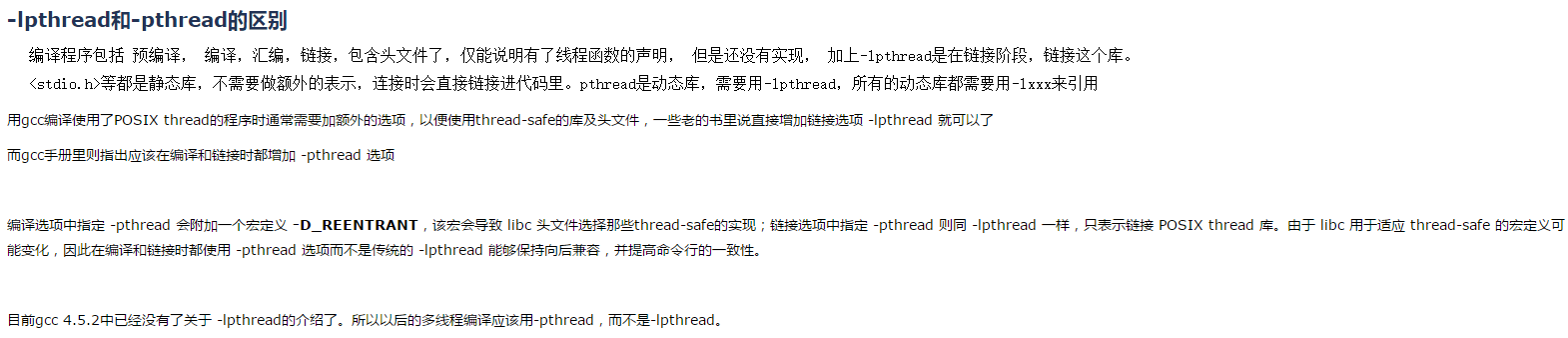


修改Makefile，编译选项增加 -std=gnu99 -lpthread，如图



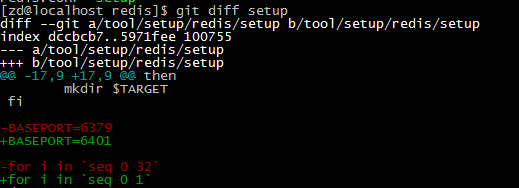
2019.04.15

通过与skynetproject20181210目录下的some-mmorpg的编译文件进行比较，发现编译参数选择-pthread比较好，具体参阅百度搜索---pthread和-lpthread有什么区别，如图：

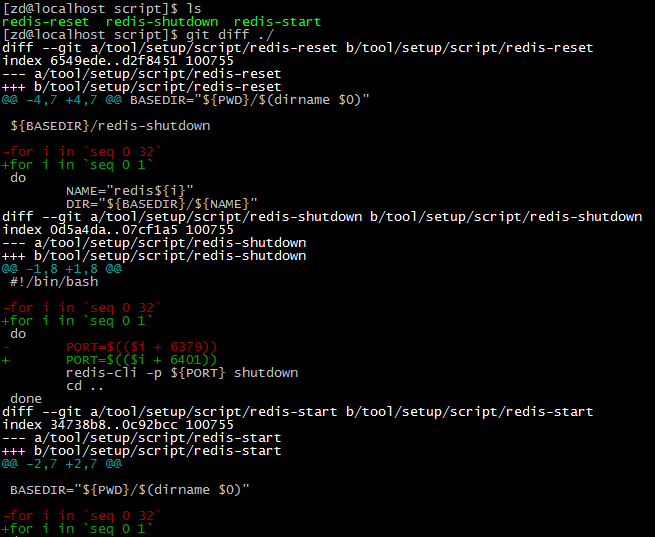


参考链接：https://www.cnblogs.com/suntp/p/6473751.html

修改 tool/setup/redis/setup 脚本（修改端口和redis数量），如图



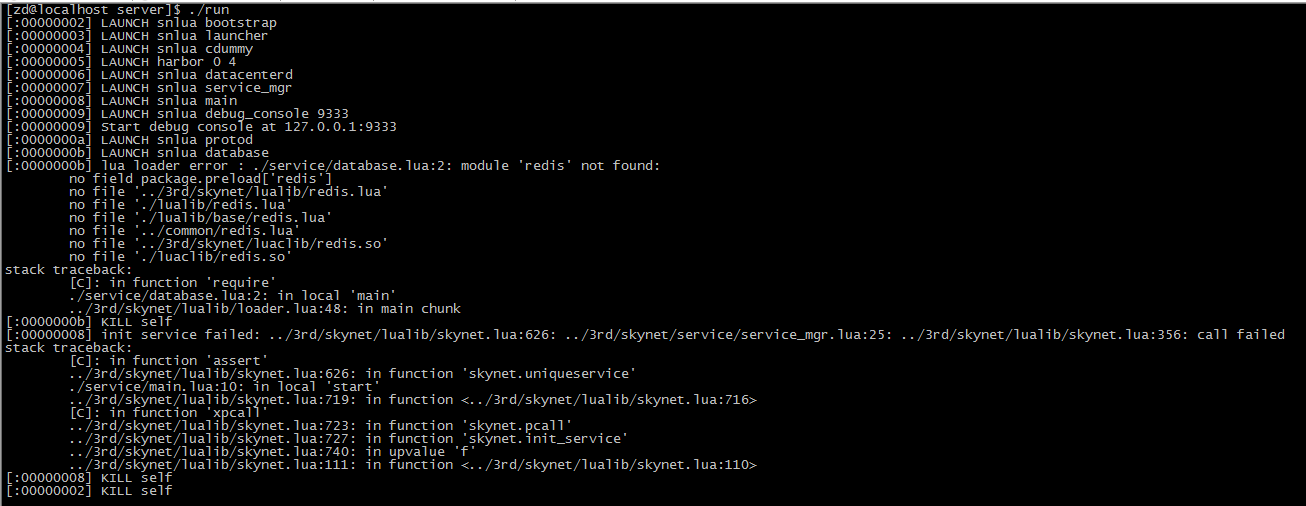
修改tool/setup/script目录下的三个redis相关脚本，如图



执行 tool/setup/setup 脚本，它会在项目根目录下创建一个var目录，创建好数据库需要的文件和管理脚本

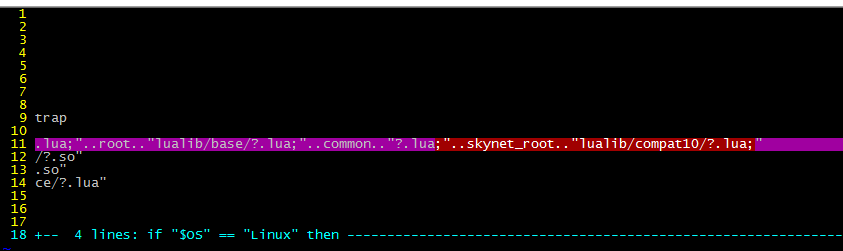
运行 var/redis-start 脚本，启动好对应的数据库实例

在server目录下使用./run运行，报错



看报错应该是最新版本的skynet的部分文件目录有更改，导致依赖文件无法找到，修改config中间中的lua\_path项，在末尾添加..skynet\_root.."lualib/compat10/?.lua;"，修改为

lua\_path = skynet\_root.."lualib/?.lua;"..root.."lualib/?.lua;"..root.."lualib/base/?.lua;"..common.."?.lua;"..skynet\_root.."lualib/compat10/?.lua;"，如图

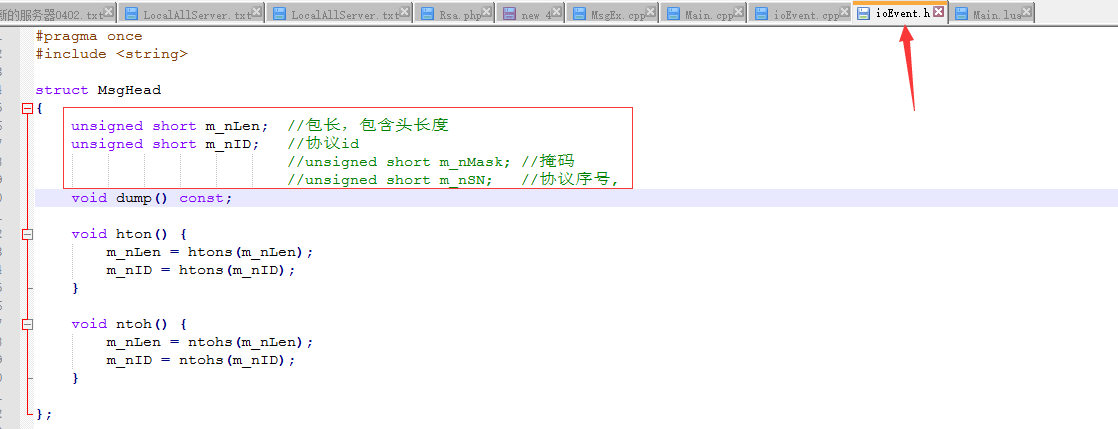


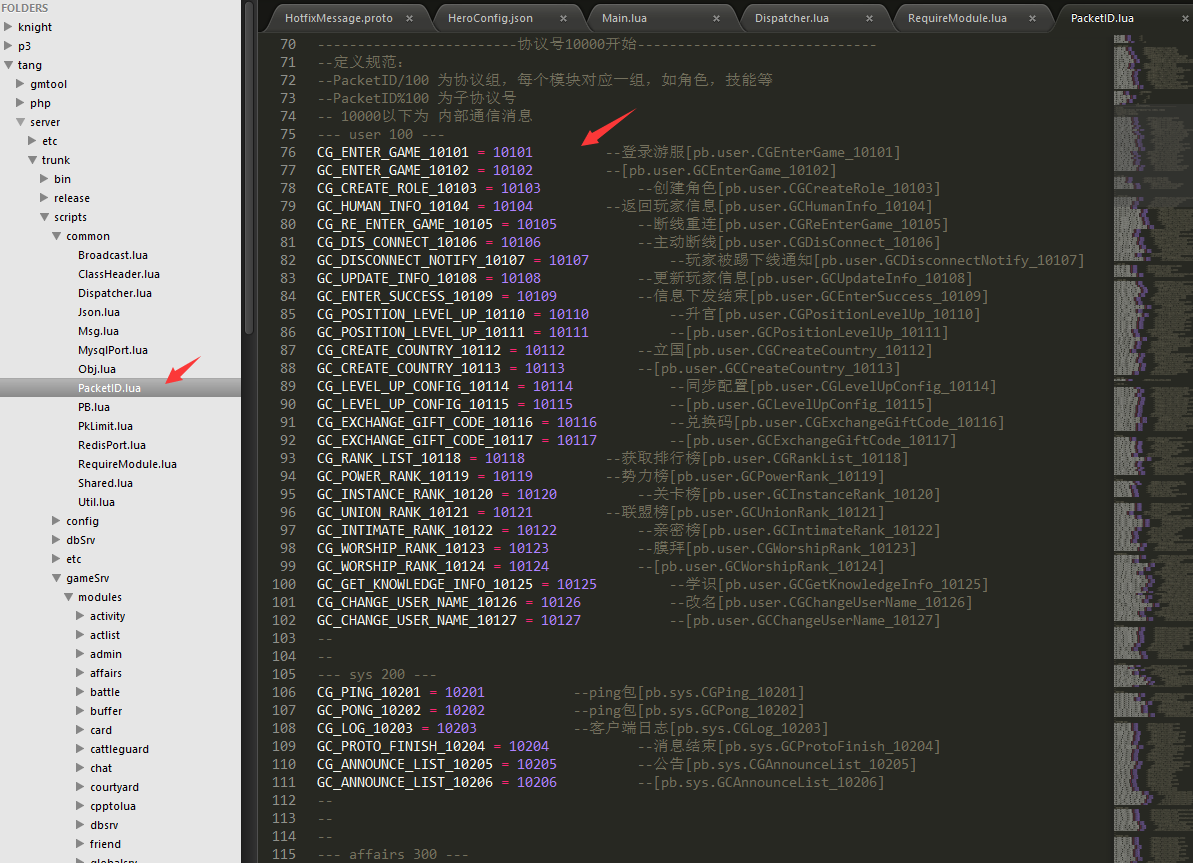
在server目录下重新使用./run运行，正常运行

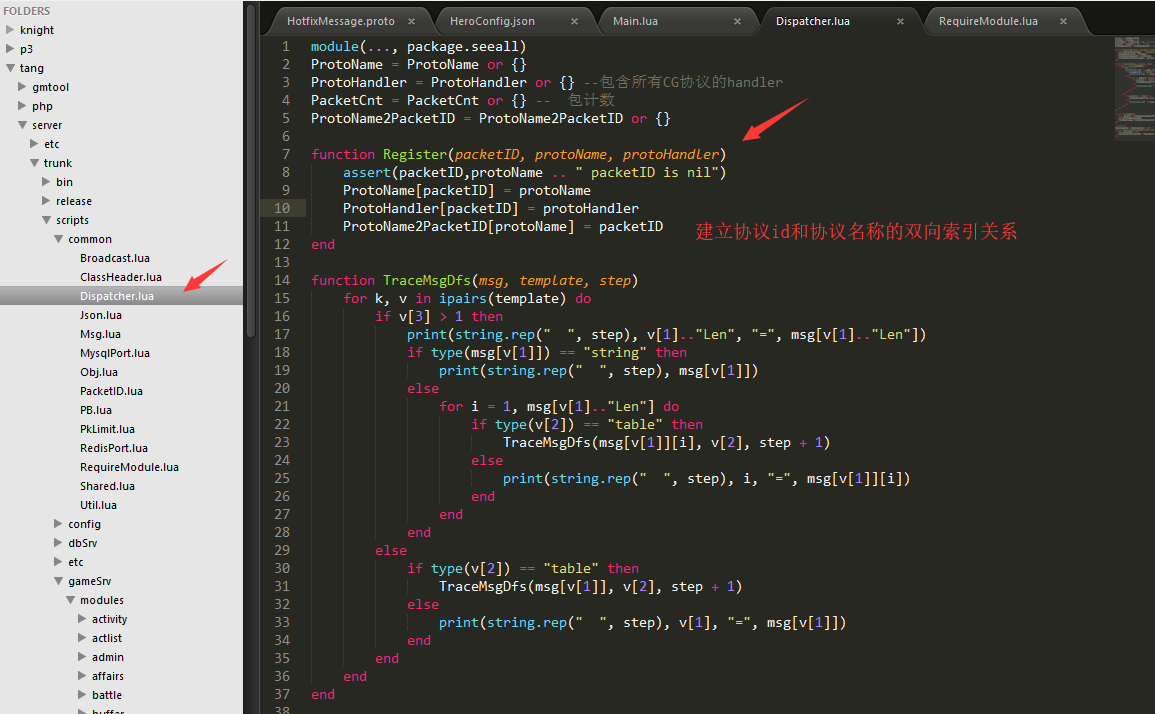
2019.04.11

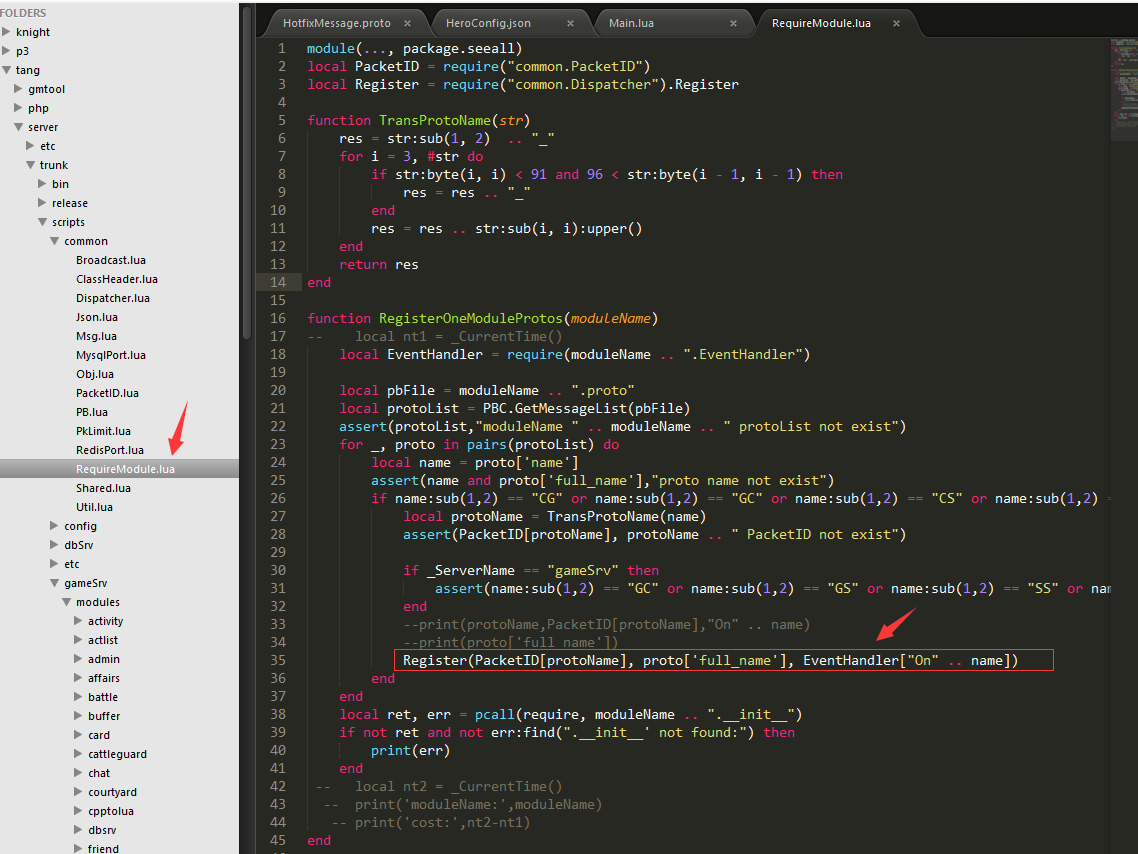
1. 为将sproto转换为protobuf，考虑使用协议id来索引protobuf中的协议名称，而传输时只传输协议id（可参考原来的异度之门项目或者tang项目），收到协议id后再去查找protobuf中对应的协议名称。

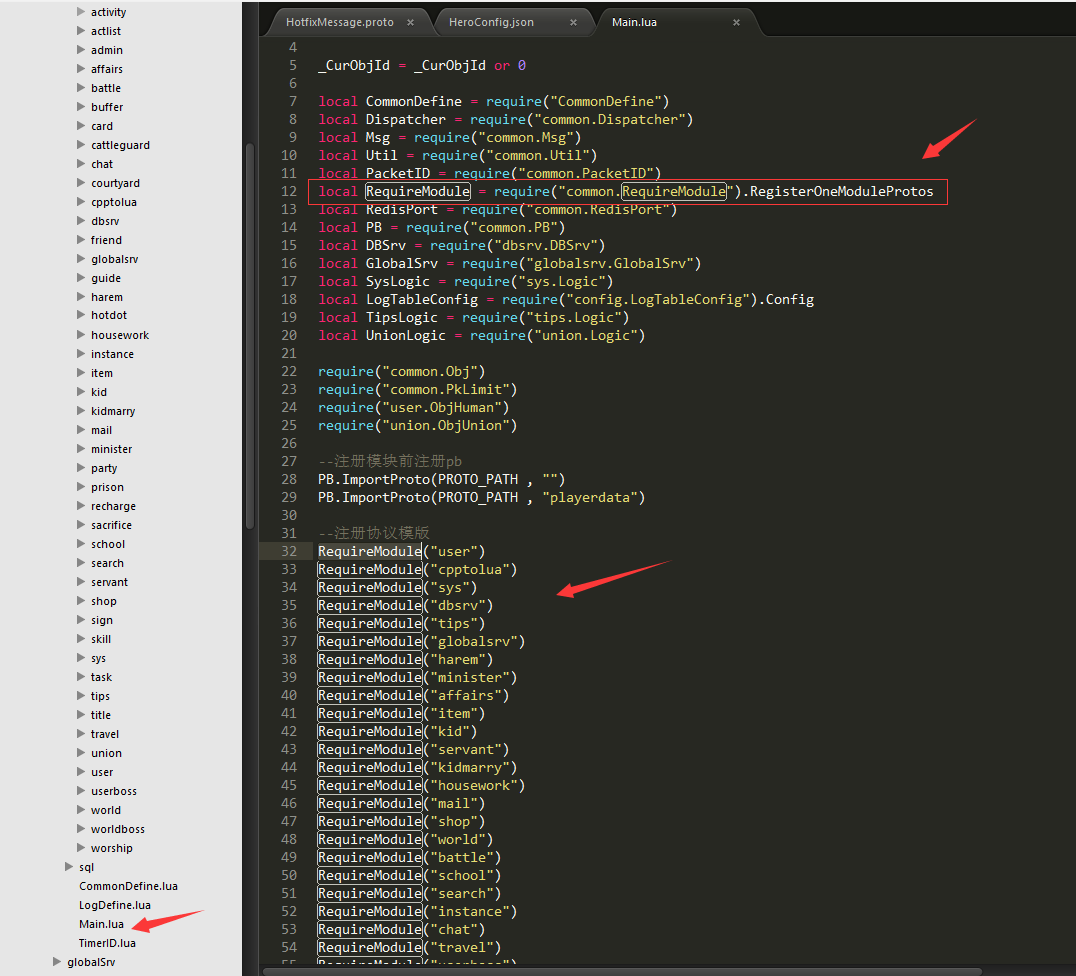
通过阅读tang项目中的模拟客户端连接发包的工具gmtool，发现其传输的包头中确实只包含包长和协议id，并不直接包含协议名称，而lua层面有建立协议id和协议名称的双向索引关系，方便在lua层面解包。而现在的挂机类项目中的处理则是把协议id和协议名称单独摘出，放到proto.config文件中，猜测实际处理时应该还是在内存中建立协议id和协议名称的双向索引关系。

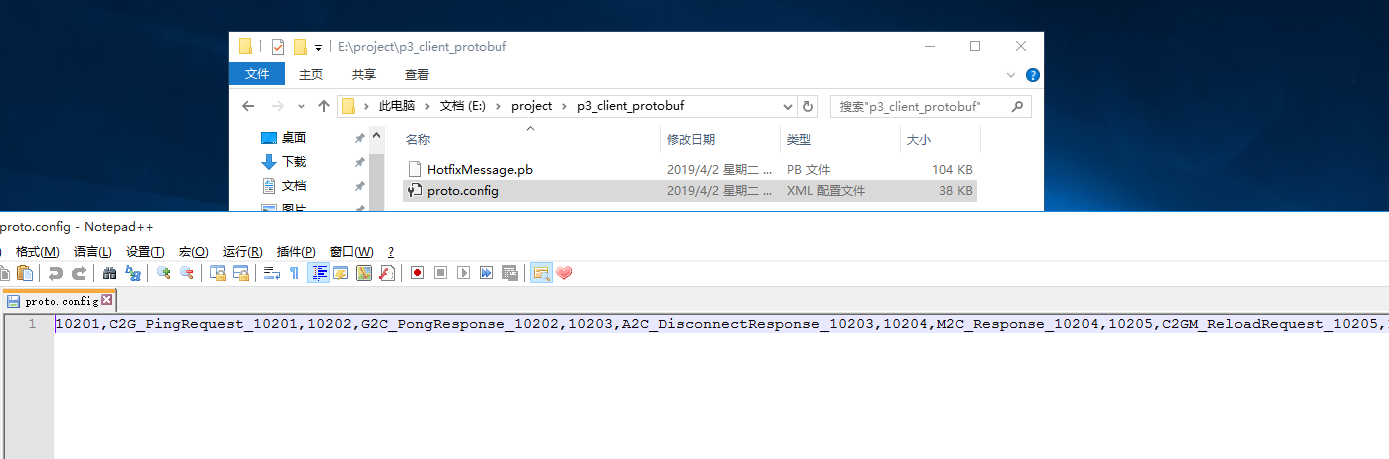












1. 考虑到做过的游戏都曾遇到过被刷包工具通过不停刷包找bug或者刷广告的问题，本项目的协议中必须要有一个客户端会话id，客户端连接服务端时服务端发送初始会话id值给客户端，后续客户端每次发包时自增然后做md5发给服务端，服务端进行校验，如果校验失败则直接断开客户端连接。

2019.04.12

通过阅读ggApp的代码，发现ggApp/tools/proto/protobuf/下的关于协议的部分刚好可以借鉴，而且common.proto中的消息头（MessageHeader）刚好包含了之前想要的会话id（session）。

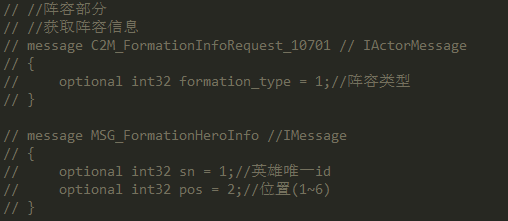
2019.04.22

因为阅读ggApp的代码目前还是有些吃力，而且暂时没有能跑起来的宏观例子，所以打算放弃编写一个基于skynet的挂机类项目的高，大，全的框架，而是先按目前能想到的编写一个简单的挂机类框架，有一些最基础的游戏模块，能跑起来就行，可能这是更务实的一条路。反正后续如果有更好的思路，持续迭代进行改进，并不会花费太多时间和精力，总比总是在想如何做出一个高，大，全的框架而止步不前好多了。记得之前云风说过，现在有什么就做什么，后面有更好的再按更好的做，大意是这样。学习之路的螺旋上升法则，确实是一个无法逾越的通用规则。

2019.04.24

借助ggApp中的python脚本，协议id与协议映射关系的对应lua文件已经可以生成，目前需要规划的是如何处理同一个service中的指令分发（多个游戏功能，比如背包，商店，副本等等）。

终于解决了前几天遇到的一个py脚本转换协议id与协议映射关系的bug，原因是因为协议中包含了注释项，例如：



使用的正则表达式字符串是：patten = "message\s+(\w+)\_(\w+)\_(\d+)\s+//"

注释部分是不应该被解析的内容，在正则表达式字符串头部加入 ^ 表示匹配行开头，但是貌似不生效，具体可能跟py的搜索机制有关，后来通过在正则表达式字符串头部加入 \n解决了这个问题。最终正确的正则表达式字符串为：

patten = "\nmessage\s+(\w+)\_(\w+)\_(\d+)\s+//"

2019.08.30

因为妖怪项目新加了好几个新模块的大单子，部分模块比较复杂，耗费了比较多的事件，做完这些模块人也变得懒散了，这中间荒废了不少时间。今天决定把后续的项目计划衔接上来。

前两周尝试把tang项目的部分接口整理出来，从.proto文件生成.pb文件后尝试实现登录流程，一直卡在解包那边，最终通过对比测试，发现是服务端使用的netpack.filter无法从">HHc"..bufSize格式的数据中正确解包，改为">HHs2"后就可以了。但是tang的PC客户端只能对">HHc"..bufSize格式的数据正确解包。

登录流程初步调通后，后续计划开始接入游戏的后续流程，就涉及到玩家数据的读取和保存这块。初步考虑独立创建一个db服务，玩家登录后从db读取数据缓存到内存中，然后进行逻辑操作，逻辑操作修改内存中的数据后立即将修改后的数据发送给db服务进行保存。后续可进行优化。一个优化的思路是考虑加一个redis中间层，将读取和保存数据都缓存在redis中，这样缓存的数据更可靠（游戏服务挂掉redis中的数据依然存在），落地也可以使用redis的队列pop，定时依次进行。