**新框架设计文档说明**

## 新架构

客户端

**获取网关ip端口**

域名网站

MongoDB·

网关2

网关1

**……**

Redis

发布订阅

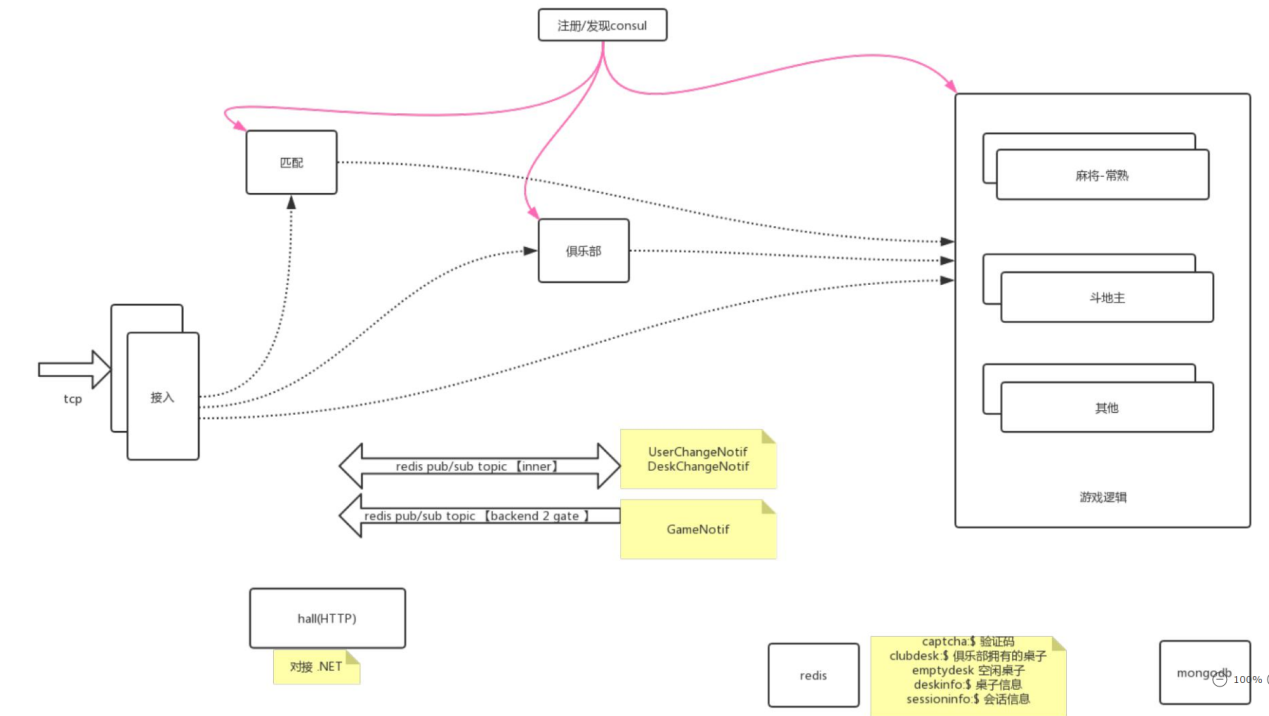
登录注册服（已经合并网关服）

Center

game server

http连.net

club server



源文件svn: https://win-0apr1ion5le:8443/svn/cyhdRepository \cy\src\cy\game\docs game.pdf

## 服务器环境

Linux centos 7.5 64位

Linux版 mongodb数据库

技术知识点:go语言，consul框架，rpc-x数据转发，Redis数据缓存，发布订阅，MongoDB数据库数据存储。

Consul用于各个服务进程之间的发现。所以需要每启动一个服务都把该服务器注册到consul服务器里面去，用于公共查询

Rpc-x用于各个服务进程之间的通信

## 协议

协议种类：HTTP，TCP两种

tcp加密方式：SSL加密方式（前端需要接入ssl库与后台进行协议动态加密联调）

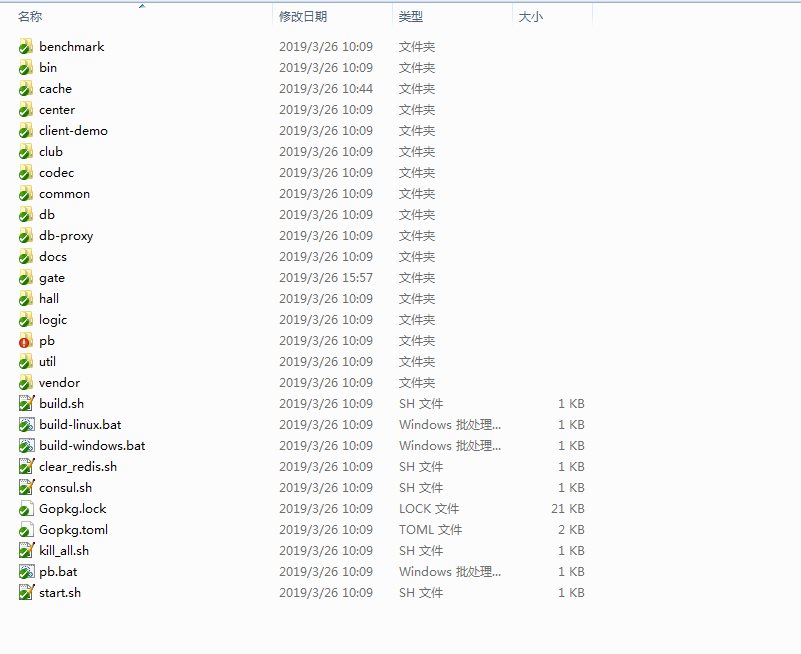
tcp协议格式：protobuffer格式进行数据

tcp协议码：

主协议码：

1. 登录网关协议：pblogin
2. 大厅主协议：pbcommon
3. 匹配场主协议：pbcenter
4. 俱乐部主协议：pbclub
5. 游戏模块主协议：pbgame\_mj\_changshu

## 服务器框架讲解



benchmark:性能测试，里面涵盖测试登录，和注册

bin:李炜加的

cache:Redis调用入口（目前涵盖：验证码，俱乐部包含桌子，空桌子可供创建房间申请，桌子信息，seesion。Redis发布入口）

center: 撮合玩家，撮合成功发到game模块去游戏

client-demo:模拟客户端调用命令，测试接口

club:俱乐部业务逻辑

codec:协议编码（涵盖json，protobuf，目前json未开发）

common: 李炜加的

db: MongoDB调用入口 （俱乐部，用户，俱乐部邮件）

db-proxy: 未用到

docs:给hall对外提供api接口文档（swagger由hall编译生成swagger.json）

gate:网关程序

hall:对外提供htttp接口，（提供.net请求接口，与.net对接）

logic:游戏逻辑模块

pb: \*.proto协议文件（inner:内部数据结构体）

util:公用方法（MongoDB保存数据格式方法，数据包压缩/解压缩，经纬度换算，临时端口，订阅通用方法，把数据保存本地磁盘/读取本地数据，）

vendor:第三方库（阿里云sdk等）

build.sh:（李炜写的）

build-windows-linux.bat:编译脚本文件

clear\_redis.sh:清理Redis命令

consul.sh:启动consul命令

Gopkg.lock: godep用的 （包管理工具：不懂请看（<https://github.com/tools/godep>））

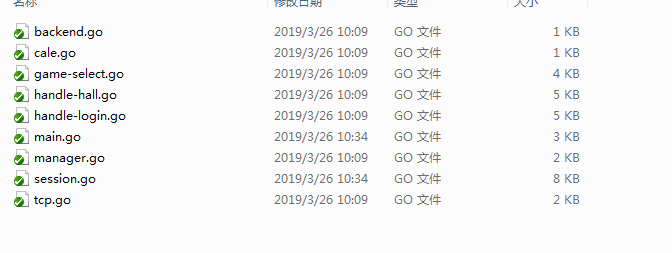
Gopkg.toml: godep 用的

Kill\_all.sh:杀掉进程命令

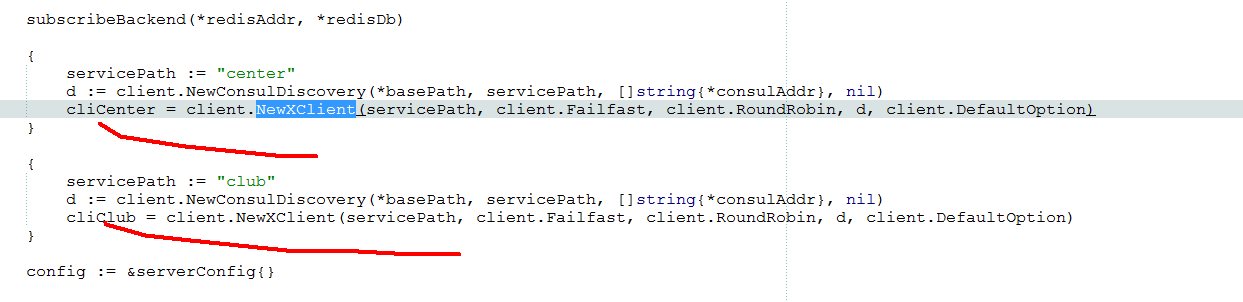
pb.bat: \*.proto用protoc.exe 生成go代码

start.sh: 启动所需进程命令

### gate网关讲解



Main.go启动网关程序：启动初始化Redis，MongoDB，redies订阅监听，center服务，club服务，tcp网络监听，logic日志 等进程。



启用rpc服务器，在rpc里面能看到center，club服务，就可以直接给这两个进程发消息

网络监听开启前加载https证书，进行加密

Tcp.go：tcp连接的监听启动，开启，关闭，调用newSession

Session.go:newSession方法 go一个监听接收消息通道，一有信息就放到通道里面，go一个hangle处理消息，通道一有数据，就会拿出来处理。如果是没登录，先处理登录，登录成功之后给后台club，game，center发布消息，由他们自己订阅。如果已经登录，走dispath方法，调用rpc发布消息给相应服务器。

handle-hall.go：处理一些大厅的业务，比如绑定手机，实名认证等

handle-login.go：处理登录心跳业务

manager.go：管理session机制

game-select.go：根据桌子ID，玩家ID选择游戏

backend.go：gate网关里面的发布订阅监听代码

### Center

Main.go:初始化日志，启动redis，初始化center Redis 启用rpc，把center注册进去rpc，启用rpc，

Handle-MatchReq.go: 匹配

Handle-CancelMatchReq.go:取消匹配

Game-select.go:匹配成功后，选择游戏服

Match.go:游戏匹配逻辑

### Club

Main.go:初始化日志，go去一个订阅，监听Redis发布的消息，启动redispool，链接数据库MongoDB，loadDB把数据库里面的俱乐部加到缓存里面来，syncdb把内存俱乐部定时存入数据库中，把club注册到rpc里面

Ack\_club\_email.go:确认邮件

Agree\_law.go: 同意俱乐部协议

Auto\_create\_desk.go： 自动建桌子

Batch\_ack.go: 批量确认邮件

Cache\_club.go:club信息本地缓存

Cache\_user.go：用户信息本地缓存

Create\_club.go:创建

Dbsync.go:同步mongodb

Deal\_identity.go:身份变更

Def.go:const定义

Exit\_club.go:退出

Invite\_join.go:邀请加入

Join\_club.go:加入

Notif.go:邮件通知

Query\_club.go:通过ID查询

Query\_club\_member.go:查询成员

Query\_email.go:查询邮件

Remove\_club.go:解散

Sub\_club.go:订阅俱乐部变化信息

Transfer\_master.go:转群主

Update\_club.go:更新俱乐部信息

### Hall

Main.go：启动发布站点

Api.go:对外提供接口

Web.go:方便doc生成，将struct定义放到非mian包

怎么生存doc \swagger文件夹里面的文件的：

vendor\github.com\swaggo\swag\cmd\swag go install默认生存到bin目录 (go build)

F:\cy\src\cy\game\docs cmd 执行swag.exe init

### Logic

游戏模块

麻将

地方a

扑克

## 环境部署

Linux环境部署：centos6.5以上版本

- 内核参数修改

- [ ] echo "\* - nofile 1048576" >> /etc/security/limits.conf

- [ ] echo "fs.file-max = 1048576" >> /etc/sysctl.conf

Consul平台部署：

- [安装](www.consul.io)

- 运行 nohup ./consul agent -dev

- [参数介绍](https://www.consul.io/docs/agent/options.html)

MongoDB部署

- [安装、运行](https://docs.mongodb.com/manual/administration/install-on-linux/)

- [备份、还原](<https://docs.mongodb.com/manual/tutorial/backup-and-restore-tools/>)

-[客户端连接]（<https://nosqlbooster.com/downloads>）

Windows安装MongoDB请看<http://baijiahao.baidu.com/s?id=1601512248926547477&wfr=spider&for=pc>

Redis部署

- [安装、运行](<https://redis.io/download>)

<http://www.runoob.com/redis/redis-install.html> windows部署方案

各个进程模块部署

启动服务器

## 数据库表结构

* 用户表 pbcommon.UserInfo

Wxid 唯一索引

userid 唯一索引

mobile 索引

* 用户ID表 userid

提供给用户表的userid自增用

* 俱乐部ID 表

ID

Inuse 是否被使用

* 俱乐部信息表

见db/mgo/club struct定义

* 俱乐部邮件ID表 emailid

提供给clubemail id自增用

* 俱乐部邮件表

见db/mgo/email ClubEmail struct定义

* 玩家俱乐部邮件表

见db/mgo/email userEmail struct定义