# 平台接入

## 先说结论，我该怎么做？

以接入 360 玩平台 s1 服务器为例，需要修改以下几个地方：

1. 在游戏服 client 项目下的 proxy 目录里新建 wan360 目录，即：

client/proxy/wan360；

1. 拷贝 client/proxy/\_\_template 目录下的所有文件到 client/proxy/wan360 目录下；
2. 根据 360 提供的文档修改 login.php、check\_user.php 和 exchange.php 三个文件，主要是修改参数拼接的部分；
3. 将 php 代码和目录结构上传到游戏服。s1 服务器需要放在：

/data/llz/s1\_game/client/proxy 目录下；

1. 修改 Nginx 配置文件，一般是放在 /data/software/nginx-1.4.7/conf/vhosts 目录下。在 s1.conf 文件中增加 URL Rewrite 重写配置：

rewrite ^/login(.\*)$ /proxy/wan360/login.php last;

rewrite ^/check\_user(.\*)$ /proxy/wan360/check\_user.php last;

rewrite ^/exchange(.\*)$ /proxy/wan360/exchange.php last;

1. 重启 Nginx，测试游戏；

接下来，需要了解一下360、港台以及其他联运平台所提供的三个核心 API。分别是：

* login
* check\_user
* exchange

## login

这个 API 主要是用于登陆验证，登陆成功后需要我们自己将页面跳转到游戏首页。

举例：在 360 平台，这个 API 所对应的 URL 地址是：

<http://s1.ukhyfntbfsp.g.360-g.net/login>

这 URL 地址个会对应 /data/llz/s37\_game/client/login.php 页面，至于配置规则，需要参考 Nginx 的配置文件。

另外，还需要注意一点，在港台服务器上，login API 所指定的 URL 可能和 360 平台的不太相同，API 名称不相同、参数规则也不相同。甚至在某些平台上可以自己定义API 的名称，例如我们可以把 login API 命名为 signin！

## check\_user

这个 API 主要是用于检查游戏内的玩家角色是否存在？如果不存在需要返回一个“不存在”的标志，如果存在则需要返回角色数据。

举例：在 360 平台，这个 API 所对应的 URL 地址是：

[http://s1.ukhyfntbfsp.g.360-g.net/check\_user](http://s37.ukhyfntbfsp.g.360-g.net/check_user)

和 login API 一样，该 API 也会随着平台不同而不同；

## exchange

这个 API 主要是用于充值，也就是将平台币冲入到游戏内。该 API 一般是以 HTTP-POST 方式请求，而 login、check\_user 这两个 API 一般使用的都是HTTP-GET 方式。为了防止玩家刷单，需要在游戏服内对订单 Id 是否重复做出检查！

特别注意：游戏服提供的 Restful 服务，一定只能绑定内网 IP 地址！

举例：在 360 平台，这个 API 所对应的 URL 地址是：

<http://s1.ukhyfntbfsp.g.360-g.net/exchange>

和 login、check\_user 这两个 API 一样，该 API 也会随着平台不同而不同；

## 从 80 到 8002

在上一节中，我们了解到游戏服提供了 Restful 服务。这是一个基于 HTTP 协议的，有点像 WebService 的玩意儿。更具体的，我们使用的是：由 JBOSS 提供的 RestEasy 框架！关于 WebService 、Restful、RestEasy 这些名词，你可以去问 Google，这里就不赘述了……但是，这里有另外一个问题：Restful 服务绑定的是 8002 端口，而不是 80 端口！也就是说游戏服是无法直接响应 HTTP 请求的！！那么 80 端口被谁给占用了呢？

答案是：Nginx。

Nginx 作为前端 HTTP 服务器，负责显示游戏服的网站首页 index.php。所以 80 端口是必须要留给 Nginx 的！如果想要 login、check\_user 和 exchange 这三个接口能够正常转发到 8002 端口，就需要 Nginx 的反向代理。



（图1）Nginx 反向代理示意图

## 解决各平台 API 不统一的办法

之前咱们提到过，360 平台的登陆接口是 login，但是其他第三方游戏平台的登陆接口可能是 signin。另外，360 平台的登陆接口，提供了一个名为 qId 的参数。而港台那边的登陆接口提供的是 uId 参数，这两个平台的参数名也不一样！举例如下：

360 登陆验证地址：

<http://s1.ukhyfntbfsp.g.360-g.net/login?qid=1001>

其他第三方登陆验证地址：

<http://xxx.xxx/signin?uid=1001>

我们最初的办法是：在游戏服内，为不同的平台开发不同的登陆验证方式。这么做的一个问题是每接入一个新平台，我们都需要开设新的 SVN 分支，修改游戏服代码，并重新打包部署到外网。所以这个方法我们就不用了……

一个改进的方法是：把游戏服提供的接口定死，然后增加一个调停文件来处理两个接口冲突的问题。因此我们增加了 login.php 代码文件，令 Nginx 把登陆地址全部重写（URL Rewrite）到login.php 文件上。login.php 代码中将原来的参数翻译成游戏服所需要的参数，最后通过 CURL 来调用游戏服的接口。



（图2）通过 php 页面来屏蔽接口变化

注意，在 php 页面中做了一些简单的逻辑，例如验证请求参数和 MD5 码等等。