年我涉猎的领域有些“广泛”，并且有那么一点“跳跃”：从上半年的终端（游戏）开发到下半年[golang](http://tonybai.com/tag/go)、[docker](http://tonybai.com/tag/docker)以及目前将要提及的[微信公众平台 接口](http://mp.weixin.qq.com/)开发，似乎有些远离了老本行C以及技术管理的内容。但在这个转型以及创新驱动的时代，这显然是顺势而为。寻求与新兴领域的主动接轨，在实打实的实践 中，扩大了自己的视野，并可以进一步甄别发现适合自己的领域。

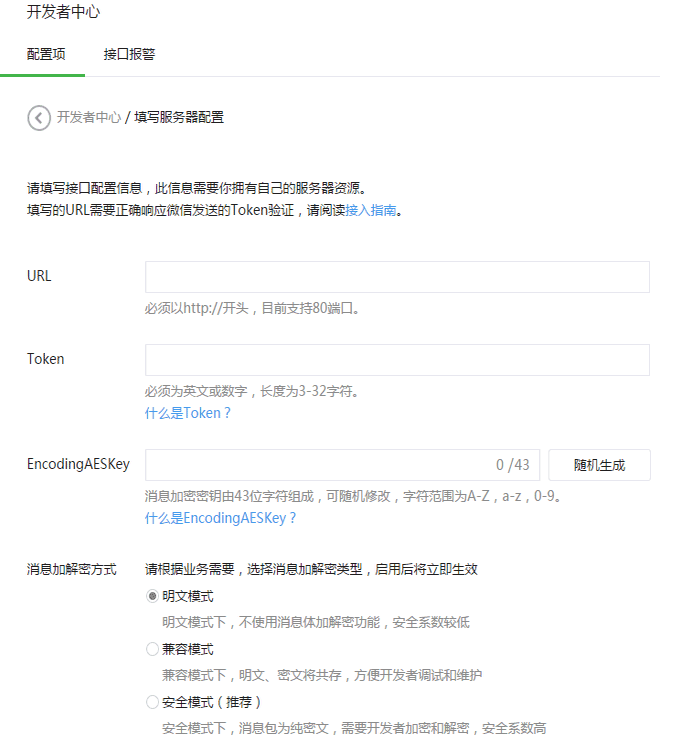
移动互联网时代，微信平台一枝独秀，是社交领域的巨人，但其诞生也才不到4年。微信平台的发展前景十分广阔，企鹅公司将其打造为人与人、人与物、物与物的统一、万能入口之雄心不变，因此围绕微信平台广大开发者依旧有诸多机会。

微信公众平台接口应该算是微信平台首批对外开放的接口吧。公众平台相对成熟，但其业务模式依旧在演进和创新。公众平台接口的开发并非不难，上手几个月就可 以写成一本诸如“微信公众平台应用开发实践”的事情就发生在你我眼前，因此这里后续有关微信公众平台接口开发的文章也都是一些入门级的，我个人也是边学 习，边实践，边记录，边分享，就像上半年写[Cocos2d-x](http://tonybai.com/tag/Cocos2d-x)文章那样。

**一、公众号申请（可选**）

本着“再小的个体，也有自己的品牌”的微信公众平台产品哲学，只要你是合法自然人类，你就可以到https://mp.weixin.qq.com/上申请一个公众号，一般对于个体而言，只能申请订阅号。

对于具有开发能力的订阅号拥有者，你可以在订阅号的“开发者中心”，启用开发者账号。并且“一旦启用并设置服务器配置后，用户发给公众号的消息以及开发者需要的事件推送，将被微信转发到该URL中”。



不过此时即便你填写相关信息并提交，你也不会通过验证。这正是本篇要告诉你的事情，如何写程序实现微信公众平台的接入验证，后续道来。

**二、测试号申请（可选）**

正式的订阅号申请有些繁琐，需要提交个人信息，需要审核，不会立即生效。并且未认证的订阅号所能使用的功能接口有限（只能使用普通消息接口），而认证又需 要一笔费用（现价300rmb/次）。对于学习者而言，也许真的没有必要。于是我们在学习开发的过程中可以申请测试号来替代真正的公众号。

测试号是一种体验账号，有效期一年，具有各种功能接口体验权限。测试号可以在http://mp.weixin.qq.com/debug/cgi-bin/sandbox?t=sandbox/login下申请，申请时有一个类似公众号开发者配置的页面需要你填写服务器配置。同样，你需要进行接入验证（后续道来）。

一旦申请成功，可以用终端微信app扫一扫测试号的QR，关注微信平台测试号，用于后续平台接口开发测试。

以上公众号和测试号**二选一**。

**三、公众号服务器**

为何需要公众号服务器，这就要谈及微信公众平台的架构了。

很多人觉得微信公众平台的业务模式有些类似于若干年前火爆一时的短信增值业务模式-SP/CP模式：

【终端用户】 <—-短信—> 【移动运营商移动增值业务网关】 <—-> 【SP/CP服务器】

微信公众号时代，这个业务模式变成了：

【终端用户】 <—-微信消息—> 【微信公众平台】 <—–> 【公众号服务器】

短信变成了微信，SP/CP变成了公众号。微信公众平台将终端用户发给公众号的信息转发至公众号配置的公众号服务器URL，公众号服务器做业务处理后，将响应信息通过微信公众平台再发给终端用户。因此我们需要实现公众号业务逻辑的公众号服务器。

本文标题里所说的“**接入验证**”，指的就是微信公众平台对公众号服务器提供服务的URL有效性的验证。我们在填写开发者中心的“接口配置信 息”并提交时，微信公众平台会向配置的公众号服务器的URL发送验证Request，只有公众号服务存在，且按要求返回包含特定信息的Response， 我们才能真正通过微信公众平台的验证，“接口配置信息”才真正生效。

因此我们需要一台放在公网的主机。如果采用Golang开发公众号服务的话，这样的主机只能是独立的VPS，像国内[新浪提供的app engine主机](http://sae.sina.com.cn/)不能运行Golang，无法满足要求（当然如果你使用其他语言开发的话，比如[PHP](http://php.net/)，那么可用的主机范围就很广泛了）。

这里建议申请一个[亚马逊免费EC2主机](https://aws.amazon.com/cn)(t2.micro型，免费一年，学习够用)用作学习测试使用或者购买像[Linode](https://www.linode.com/?r=ec42f83592f2ddfc8f487c0428a9b74fa1b2984b)或[DigitalOcean](https://www.digitalocean.com/?refcode=bff6eed92687)的VPS。关于如何申请亚马逊主机可以咨询谷歌和度娘，这里不赘述。

注意：[Amazon EC2](http://aws.amazon.com/cn/ec2/)实例默认采用的时动态IP，instance重启后IP会发生变化。因此可申请分配一个Elastic IP，并绑定在你的EC2实例上，目前绑定instance的Elastic IP是免费的，这个IP在instance重启后不会变更。当你EC2主机到期后，记得释放这个IP，否则就收费了。

**四、接入验证**逻辑

前面提到过，无论是公众号还是测试号，当你提交配置URL时会收到提交失败的信息，这是微信公众平台接入验证失败所致。在公众平台开发者文档中，关于URL验证逻辑如下：

开发者提交信息(包括URL、Token)后，微信服务器将发送Http Get请求到填写的URL上，GET请求携带四个参数：signature、timestamp、nonce和echostr。公众号服务程序应该按如下要求进行接入验证：

1. 将token、timestamp、nonce三个参数进行字典序排序  
2. 将三个参数字符串拼接成一个字符串进行sha1加密  
3. 将加密后获得的字符串与signature对比，如果一致，说明该请求来源于微信  
4. 如果请求来自于微信，则原样返回echostr参数内容

以上完成后，接入验证就会生效，开发者配置提交就会成功。

列出Http抓包分析后的文本，理解起来就更容易些:

微信服务器发出的验证Request如下：

GET /?signature=d01007dcff994c555bc51d22e154956ccdc61ec5&timestamp=1418970951&nonce=484765335&echostr=qwe1235 HTTP/1.0\r\n  
User-Agent: Mozilla/4.0\r\n  
Accept: \*/\*\r\n  
Host: wechat.tonybai.com\r\n  
Pragma: no-cache\r\n  
Content-Length: 0\r\n

应答返回如下：  
HTTP/1.0 200 OK\r\n  
Date: Fri, 19 Dec 2014 06:35:59 GMT\r\n  
Content-Length: nn\r\n  
Content-Type: text/plain; charset=utf-8\r\n

qwe1235

**五、参考实现**

环境：AWS t2.micro ubuntu 14.04 x86\_64 Server  
     [go 1.4](http://tonybai.com/2014/11/04/some-changes-in-go-1-4/)

Go语言标准库提供了一个强大的http server，我们直接利用这个server来处理微信平台的Url验证请求。另外微信平台发给公众平台服务器的http request都是请求到"/"下的，这样我们的service无需设置太多http route。

//urlvalidation.go  
package main

import (  
        "crypto/sha1"  
        "fmt"  
        "io"  
        "log"  
        "net/http"  
        "sort"  
        "strings"  
)

const (  
        token = "wechat4go"  
)

func makeSignature(timestamp, nonce string) string {  
        sl := []string{token, timestamp, nonce}  
        sort.Strings(sl)  
        s := sha1.New()  
        io.WriteString(s, strings.Join(sl, ""))  
        return fmt.Sprintf("%x", s.Sum(nil))  
}

func validateUrl(w http.ResponseWriter, r \*http.Request) bool {  
        timestamp := strings.Join(r.Form["timestamp"], "")  
        nonce := strings.Join(r.Form["nonce"], "")  
        signatureGen := makeSignature(timestamp, nonce)

        signatureIn := strings.Join(r.Form["signature"], "")  
        if signatureGen != signatureIn {  
                return false  
        }  
        echostr := strings.Join(r.Form["echostr"], "")  
        fmt.Fprintf(w, echostr)  
        return true  
}

func procRequest(w http.ResponseWriter, r \*http.Request) {  
        r.ParseForm()  
        if !validateUrl(w, r) {  
                log.Println("Wechat Service: this http request is not from Wechat platform!")  
                return  
        }  
        log.Println("Wechat Service: validateUrl Ok!")  
}

func main() {  
        log.Println("Wechat Service: Start!")  
        http.HandleFunc("/", procRequest)  
        err := http.ListenAndServe(":80", nil)  
        if err != nil {  
                log.Fatal("Wechat Service: ListenAndServe failed, ", err)  
        }  
        log.Println("Wechat Service: Stop!")  
}

编译这个go源码，执行urlvalidation。

$> urlvalidation  
2014/12/18 17:48:10 Wechat Service: Start!  
2014/12/18 17:48:10 Wechat Service: ListenAndServe failed, listen tcp :80: bind: permission denied

程序提示没有权限绑定80端口。80端口只有管理员权限才能绑定，因此我们需要通过sudo方式执行validation。

$ sudo ./urlvalidation  
2014/12/18 09:56:29 Wechat Service: Start!

接下来我们回到订阅号开发者中心配置页面或测试号服务器配置页面，点击提交。在我们的公众号服务器后台可以看到如下日志：

2014/12/18 09:56:52 Wechat Service: validateUrl Ok!

同时你的提交也会显示成功，Url已经验证通过，你将正式成为开发者。

如果我们随意构造一个http get 请求发给validate程序，比如：

curl -s http://wechat.tonybai.com（比如我的URL为http://wechat.tonybai.com）

那么我们将看到validation输出如下错误日志：

2014/12/18 10:02:07 Wechat Service: this http request is not from Wechat platform!

以上源码文件在[这里](https://github.com/bigwhite/experiments/tree/master/wechat_examples)可以下载。

处于安全考虑，后续订阅号平台均需要对收到的http request进行验证，以确保请求来源于微信公众平台。

文章地址：https://tonybai.com/2014/12/18/access-validation-for-wechat-public-platform-dev-in-golang/