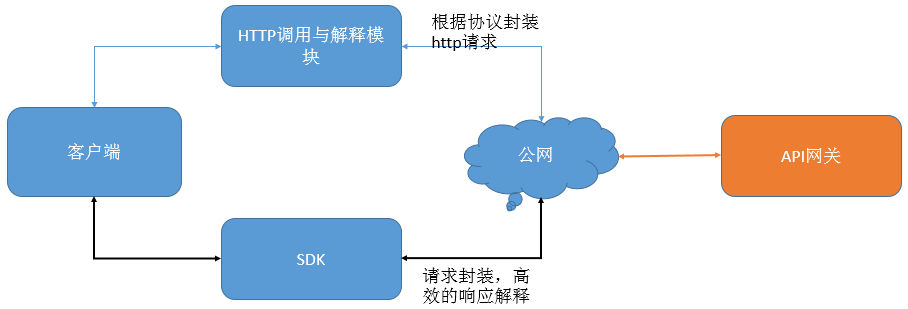
API调用方法详解

开放平台的API是基于HTTP协议来调用的，开发者可以直接使用已经支持语言的SDK—java/php(php只做参考)，也可以自己来封装HTTP请求进行调用，以下主要是针对自行封装HTTP请求进行API调用的原理进行详细解说

调用流程

根据协议：填充参数 > 生成秘钥 > 进行加密 > 生成签名 > 拼装HTTP请求 > 发起HTTP请求 > 得到HTTP响应 > 解释



调用入口

调用API的服务URL地址，目前提供了4个环境供使用：开发环境，测试环境，预发环境，生产环境。

1.开发环境：供开发人员编写代码，自我测试使用

2.测试环境：供开发和测试人员进行调测使用

3.预发环境：应用上线之前的正式模拟环境

4.正式环境：应用上线之后使用的环境，会进行流量控制

|  |  |  |
| --- | --- | --- |
| 调用环境 | 服务地址(HTTP) | 服务地址(HTTPS) |
| 开发环境 | http://openapi.dev.pajk.cn/api/v1/ | https://openapi.dev.pajk.cn/api/v1/ |
| 测试环境 | [http://openapi.test.pajk.cn](http://openapi.test.pajk.cn/)/api/v1/ | https://openapi.test.pajk.cn/api/v1/ |
| 预发环境 | [http://openapi.pre.jk.cn/](http://openapi.pre.jk.cn/api/v1/)api/v1/ | https://openapi-pre.jk.cn/api/v1/ |
| 生产环境 | http://openapi.jk.cn/api/v1/ | https://openapi.jk.cn/api/v1/ |

公共参数

调用任何一个API都必须传入的参数，目前支持的公共参数有：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必须 | 参数描述 | |
| apiId | String | 是 | 接口的唯一标识 |
| partnerId | String | 是 | 合作伙伴ID |
| key | String | 否（http必须，https依据配置而定） | 用于加密的秘钥 |
| apiGroup | String | 是 | API分组唯一ID |
| apiName | String | 是 | 接口名称 |
| version | String | 是 | API的版本标识(默认0.1.0) |

业务参数

API调用除了必须包含公共参数外，如果API本身有业务级的参数也必须传入，参数依次是arg1，arg2，arg3等

加密和签名算法

为了防止API调用过程中被黑客恶意纂改，调用任何一个API前都需要对数据进行加密和签名（https除外），服务会根据请求参数，对签名进行验证，签名不合法的请求将会被拒绝。目前平台支持的加密算法是3DES的分段加密算法，签名算法是hash\_hmac('sha1', $str, $key, true)。

PHP加密算法示例

1 var $key;//秘钥

2 var $iv;//填充向量

3

4 //初始化

5 function Crypt3Des($key, $iv){

6 $this->key = $key;

7 $this->iv = $iv;

8 }

9

10 //加密

11 function encrypt($input){

12 $size = mcrypt\_get\_block\_size(MCRYPT\_3DES,MCRYPT\_MODE\_CBC);

13 $input = $this->pkcs5\_pad($input, $size);

14 $key = str\_pad($this->key,24,'0');

15 $td = mcrypt\_module\_open(MCRYPT\_3DES, '', MCRYPT\_MODE\_CBC, '');

16 if( $this->iv == '' )

17 {

18 $iv = @mcrypt\_create\_iv (mcrypt\_enc\_get\_iv\_size($td), MCRYPT\_RAND);

19 }

20 else

21 {

22 $iv = $this->iv;

23 }

24 @mcrypt\_generic\_init($td, $key, $iv);

25 $data = mcrypt\_generic($td, $input);

26 mcrypt\_generic\_deinit($td);

27 mcrypt\_module\_close($td);

28 $data = bin2hex($data);

29 return $data;

30 }

31

32 //填充加密快的大小

33 function pkcs5\_pad ($text, $blocksize) {

34 $pad = $blocksize - (strlen($text) % $blocksize);

35 return $text . str\_repeat(chr($pad), $pad);

36 }

php签名算法示例

//签名

1 public function signHmac($str,$key)

2 {

3 $signature = '';

4 if(function\_exists('hash\_hmac')){

5 $signature = hash\_hmac('sha1', $str, $key, true);

6 }else{

7 $blocksize = '64';

8 $hashfunc = 'sha1';

9 if(strlen($key) > $blocksize){

10 $key = pack('H\*',$hashfunc($key));

11 }

12 $key = str\_pad($key, $blocksize, chr(0X00));

13 $ipad = str\_repeat(chr(0X36), $blocksize);

14 $opad = str\_repeat(chr(0X5c), $blocksize);

15 $hmac = pack(

16 'H\*', $hashfunc(

17 ($key ^ $opad) . pack(

18 'H\*', $hashfunc(

19 ($key ^ $ipad) . $str

20 )

21 )

22 )

23 );

24 $signature = $hmac;

25 }

26 return $signature;

27 }

调用示例

以调用openapi.dev.pajk.cn为例

加密算法：

1.按照http post请求方式拼装请求参数，参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必须 | 参数描述 |
| \_\_o\_s | String | 是 | 公共参数中的apiId |
| \_\_o\_v | String | 是 | 公共参数中的version |
| arg1 | String | 否 | 所有的业务参数，形式如下arg1=a&arg2=b |
| \_\_o\_r | String | 是 | 随机数 |

下面以PHP进行加密和签名算法的详解：

1.公共参数如下

appId=c88399a45eb54ecc872972400d880200#OPENAPI\_DUBBO\_DEV

partnerId =openApi\_middleware

key=600959f7120cf956fd0f6bb85f291b83

apiGroup= api\_middleware

apiName =getOpenApiInfo

version=0.1.0

2.业务参数为：

arg1= {"apiId":1,"apiName":"ApiInfo","dateTime":"2016-12-02 12:22:23","serialNum":2}

3.依据参数规则，拼装出如下待加密参数

\_\_o\_s=c88399a45eb54ecc872972400d880200#OPENAPI\_DUBBO\_DEV&\_\_o\_v=0.1.0&arg1={"apiId":1,"apiName":"ApiInfo","dateTime":"2016-12-02 12:22:23","serialNum":2}&\_\_o\_r=

297335.97035396

4.根据key生成加密秘钥

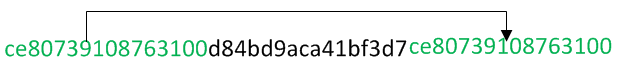
key=600959f7120cf956fd0f6bb85f291b83

对key进行md5散列

注：php中默认是32位的16进制的字符，Java中直接调用API生成16位字节

md5(600959f7120cf956fd0f6bb85f291b83)= ce80739108763100d84bd9aca41bf3d7

将前16位的字符复制到后16位，变成48位的字符，如下



将上面48位的字符串转成24位的字节序列

pack('H48', "ce80739108763100d84bd9aca41bf3d7ce80739108763100")

5.生成16位的空向量，以供后面加密使用

pack('H16','0000000000000000')

6.进行加密

使用第3步的待加密参数进行加密操作：

encrypt(“\_\_o\_s=c88399a45eb54ecc872972400d880200#OPENAPI\_DUBBO\_DEV&\_\_o\_v=0.1.0&arg1={"apiId":1,"apiName":"ApiInfo","dateTime":"2016-12-02 12:22:23","serialNum":2}

&\_\_o\_r=297335.97035396”)

生成加密串为：

ea50a7c13fbec20cd251a91ce80755cb6508a55a8403df496a51db99ccc3ee7cda7cca093246f499abb8e65b9aaac5e988428937181101463487cd3fb127ffb83b394d0ae33cca3fee6a5e1924c2f407b683c7cc3b5d246a2e99bb04bfe349c968921e97a42e73007f4df10bff808eb3a4af9174a73e3ba16fb25c5352a20ae084850e0445167b0f108b15277bd6baa4f540678022db245092a55b1b7f081ddeb9a60e2e347dd8322d2aabdd9d533658

7.生成签名

使用第6步生成的加密串，并在该加密串后面增加一个随机数变成如下：

ea50a7c13fbec20cd251a91ce80755cb6508a55a8403df496a51db99ccc3ee7cda7cca093246f499abb8e65b9aaac5e988428937181101463487cd3fb127ffb83b394d0ae33cca3fee6a5e1924c2f407b683c7cc3b5d246a2e99bb04bfe349c968921e97a42e73007f4df10bff808eb3a4af9174a73e3ba16fb25c5352a20ae084850e0445167b0f108b15277bd6baa4f540678022db245092a55b1b7f081ddeb9a60e2e347dd8322d2aabdd9d533658297335.97035396

调用签名函数进行签名

signHmac(“ea50a7c13fbec20cd251a91ce80755cb6508a55a8403df496a51db99ccc3ee7cda7cca093246f499abb8e65b9aaac5e988428937181101463487cd3fb127ffb83b394d0ae33cca3fee6a5e1924c2f407b683c7cc3b5d246a2e99bb04bfe349c968921e97a42e73007f4df10bff808eb3a4af9174a73e3ba16fb25c5352a20ae084850e0445167b0f108b15277bd6baa4f540678022db245092a55b1b7f081ddeb9a60e2e347dd8322d2aabdd9d533658297335.97035396”, “600959f7120cf956fd0f6bb85f291b83”)

生成的16进制的签名串如下：

a03ccfad4205458e378129de0547f2450ca414af

8.拼装请求参数

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必须 | 参数描述 |
| p | String | 是 | partnerId |
| v | String | 是 | API的版本标识(默认0.1.0) |
| s | String | 是 | 第6步的加密随机数 |
| h | String | 是 | 第7步的签名值 |

另：在http请求体中放入第6步的数据

q= ea50a7c13fbec20cd251a91ce80755cb6508a55a8403df496a51db99ccc3ee7cda7cca093246f499abb8e65b9aaac5e988428937181101463487cd3fb127ffb83b394d0ae33cca3fee6a5e1924c2f407b683c7cc3b5d246a2e99bb04bfe349c968921e97a42e73007f4df10bff808eb3a4af9174a73e3ba16fb25c5352a20ae084850e0445167b0f108b15277bd6baa4f540678022db245092a55b1b7f081ddeb9a60e2e347dd8322d2aabdd9d533658

上面的参数拼装出来的请求URL如下

http://openapi.dev.pajk.cn/api/v1/api\_middleware/getOpenApiInfo?p=openApi\_middleware&v=0.1.0&s=297335.97035396&h=a03ccfad4205458e378129de0547f2450ca414af

到此加密签名逻辑到发起请求就已经结束了，下面在简要阐述一下获取数据后的解密逻辑

解密示例代码如下

1 var $key;//秘钥

2 var $iv;//填充向量

3

4 //初始化

5 function Crypt3Des($key, $iv){

6 $this->key = $key;

7 $this->iv = $iv;

8 }

9

10 //填充加密快的大小

11 function pkcs5\_pad ($text, $blocksize) {

12 $pad = $blocksize - (strlen($text) % $blocksize);

13 return $text . str\_repeat(chr($pad), $pad);

14 }

15

16 //解密

17 function decrypt($encrypted){

18 $encrypted = pack('H\*', $encrypted);

19 $key = str\_pad($this->key,24,'0');

20 $td = mcrypt\_module\_open(MCRYPT\_3DES,'',MCRYPT\_MODE\_CBC,'');

21 if( $this->iv == '' )

22 {

23 $iv = @mcrypt\_create\_iv (mcrypt\_enc\_get\_iv\_size($td), MCRYPT\_RAND);

24 }

25 else

26 {

27 $iv = $this->iv;

28 }

29 $ks = mcrypt\_enc\_get\_key\_size($td);

30 @mcrypt\_generic\_init($td, $key, $iv);

31 $decrypted = mdecrypt\_generic($td, $encrypted);

32 mcrypt\_generic\_deinit($td);

33 mcrypt\_module\_close($td);

34 $y=$this->pkcs5\_unpad($decrypted);

35 return $y;

36 }

9.首先要进行签名验证

返回数据如下：

s=0.5702584&h=eb950292794ed78341193e5fff4bb044bae0d731&d=

进行签名验证，步骤如下

9.1取出s和d，将s拼接在d的后面生成如下待签名串、

.5702584

调用签名函数

signHmac(.5702584)

生成的签名值如下

eb950292794ed78341193e5fff4bb044bae0d731

9.2取出h值和9.1步中生成的签名值进行对比，一致就进行第10步

10.进行数据解密

这里需要使用到上面加密部分第4步生成的加密秘钥，以及第5步的填充向量

取出返回中的d，调用解密函数decrypt，进行解密

decrypt()

解密完成后在进行url的解码，就获得最后的数据，数据如下

[apiId] => 1047800729 [apiName] => openApi\_service\_result-695762193 [dateTime] => 1481858995920 [serialNum] => 846468660

https方式

下面简单讲一下不进行加密的https方式，如果需要加密，则只需按照上面的1~7步进行

按照http post请求方式拼装请求URL，参数如下

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必须 | 参数描述 |
| p | String | 是 | partnerId |
| v | String | 是 | API的版本标识(默认0.1.0) |
| s | String | 是 | 随机数 |

1.公共参数如下

appId=c88399a45eb54ecc872972400d880200#OPENAPI\_DUBBO\_DEV

partnerId =openApi\_middleware

key=600959f7120cf956fd0f6bb85f291b83

apiGroup= api\_middleware

apiName =getOpenApiInfo

version=0.1.0

2.拼装请求URL

根据上面的参数拼装请求URL，如下

https://openapi.dev.pajk.cn/api/v1/api\_middleware/getOpenApiInfo?p=openApi\_middleware&v=0.1.0&s=939264.07161134

3.http请求body中需要放入如下数据

|  |  |  |  |
| --- | --- | --- | --- |
| 参数名称 | 参数类型 | 是否必须 | 参数描述 |
| \_\_o\_s | String | 是 | 公共参数中的apiId |
| \_\_o\_v | String | 是 | 公共参数中的version |
| arg1 | String | 否 | 所有的业务参数，形式如下arg1=a&arg2=b |
| \_\_o\_r | String | 是 | 随机数 |

\_\_o\_s= c88399a45eb54ecc872972400d880200#OPENAPI\_DUBBO\_DEV&\_\_o\_v=0.1.0&arg1=

{"apiId":1,"apiName":"ApiInfo","dateTime":"2016-12-02 12:22:23","serialNum":2}&\_\_o\_r=

939264.07161134

并进行urlCode生成如下字符串

q=\_\_o\_s%3Dc88399a45eb54ecc872972400d880200%23OPENAPI\_DUBBO\_DEV%26\_\_o\_v%3D0.1.0%26arg1%3D%7B%22apiId%22%3A1%2C%22apiName%22%3A%22ApiInfo%22%2C%22dateTime%22%3A%222016-12-02+12%3A22%3A23%22%2C%22serialNum%22%3A2%7D%26\_\_o\_r%3D939264.07161134

到此https发送请求已经结束，下面进行一下数据解析的阐述

4.进行数据解析

第3步请求响应的数据如下

[message] => [object] => \_\_o\_o=%7B%22apiId%22%3A-1379517594%2C%22apiName%22%3A%22openApi\_service\_result-963329660%22%2C%22dateTime%22%3A1481880856801%2C%22class%22%3A%22com.pajk.saturn.openapi.dubbodemo.api.ApiInfo%22%2C%22serialNum%22%3A1043364681%7D&\_\_o\_s=1481880856124&\_\_o\_v=0.1.0&\_\_o\_r=1481880856124.262 [tips] => [code] => 0 [version] => 0.1.0 )

调用parse\_str函数获得如下数据：

[\_\_o\_o] => {"apiId":-1379517594,"apiName":"openApi\_service\_result-963329660","dateTime":1481880856801,"class":"com.pajk.saturn.openapi.dubbodemo.api.ApiInfo","serialNum":1043364681} [\_\_o\_s] => 1481880856124 [\_\_o\_v] => 0.1.0 [\_\_o\_r] => 1481880856124.262 )

其中的\_\_o\_o就是想要获得的数据