# 新颜科技产品服务接入指引

##### 1.引言

1.1 编写目的

本文档是为了帮助商户更快接入新颜科技接口服务而编写，适合技术人员来阅读。

1.2 技术支持联系方式

电话：18321124941

邮箱：support@xinyan.com

QQ：3312259832

1.3 公司背景

上海新颜科技服务有限公司成立于2016年，是国内领先的互联网金融服务集团漫道集团旗下公司，总部位于上海，在北京、深圳等地设有分支机构。公司聚集了来自金融、IT和互联网行业的风控、数据、开发等领域的资深专家，拥有一支专业高效的高精尖团队。

作为一家独立的第三方大数据风控服务提供商，自成立之初，新颜便致力于通过云计算、机器学习等技术客观构建综合信用信息服务平台，为非银信贷、银行、保险、证券基金理财、社交平台、电商、共享平台等多个行业提供专业的大数据服务及风控解决方案。

成立至今，新颜已与超500家机构建立了长期战略合作关系，服务覆盖全国。以“用精准的数据为客户提供至臻的服务”为使命，秉承“以信为本，以诚为怀，求是务实，传承创新”的企业文化，新颜通过持续的产品与技术创新为行业发展赋能。

##### 2.对接流程

2.1 商务经理建立对接沟通群

商务建立对接群后双方商务会拉技术人员进群确定需要对接的产品，双方对接人员进行初步沟通。

2.2 测试环境联调

确定需要对接的产品后，新颜会提供当前最新的接口文档和Demo供参考，开发人员可以参考Demo和接口文档进行对接，走通测试环境流程，尽量实现前期的基本逻辑，做好上线准备。

2.3 上线准备

在协议流程完成后，新颜会根据协议给商户开通对应正式环境信息，客户获取信息后可以配置生产环境，将之前的测试环境配置替换为生产信息，配置说明如下：

1.获取商户号和终端号(商户号和终端号是相同的)，开通后会发送到业务联系人邮箱。

2.替换信息：商户号、终端号、私钥文件、私钥密码、公钥文件和生产地址URL。

3.生产和测试环境地址可在接口文档获取，需要变更域名 ([生产域名https://api.xinyan.com],[测试域名<https://test.xinyan.com>])。

4.生产证书使用工具生成(支持windows和linux)，运行脚本自定义密钥名称和密码，生成后将cer公钥上传到https://open.xinyan.com商户管理平台，私钥pfx和密码代码中使用，详细请见。

2.4 切量后上线观察

正式上线后，新颜会协助一起观察交易稳定情况，核对双方请求数据是否同步，核对无误后上线完成。

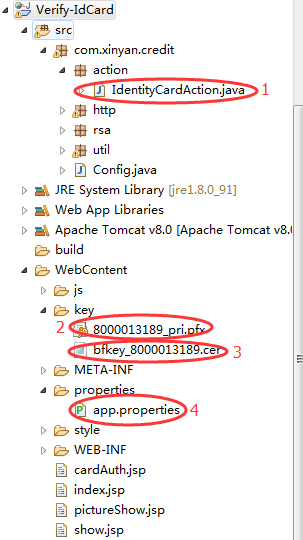
##### 3.Demo参考

3.1 说明

新颜一般会给出三种常见语言的Demo供参考，分别为java、php、c#。

3.2 java样例

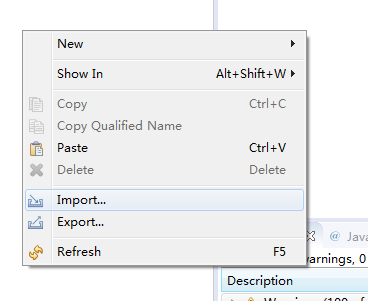
新颜发出来的一般为web工程，以实名认证为例，大概结构为：

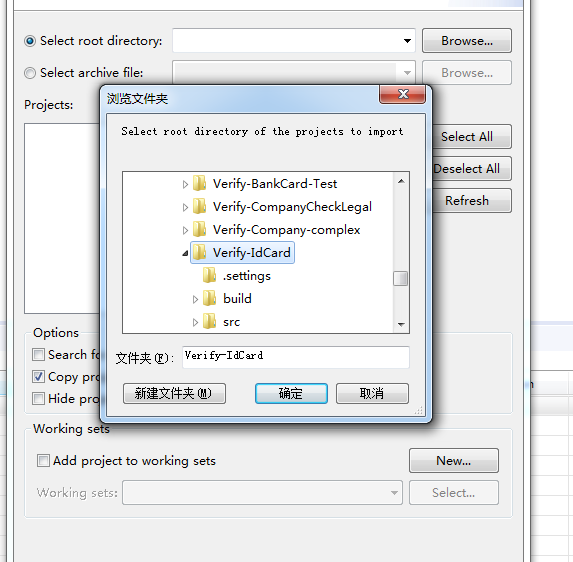


1. 关键逻辑实现类
2. 私钥文件（对业务参数进行加密）
3. 公钥文件（一般用不到，返回为明文不需要用来解密）
4. 配置文件（商户号、终端号、私钥密码、请求地址等配置）

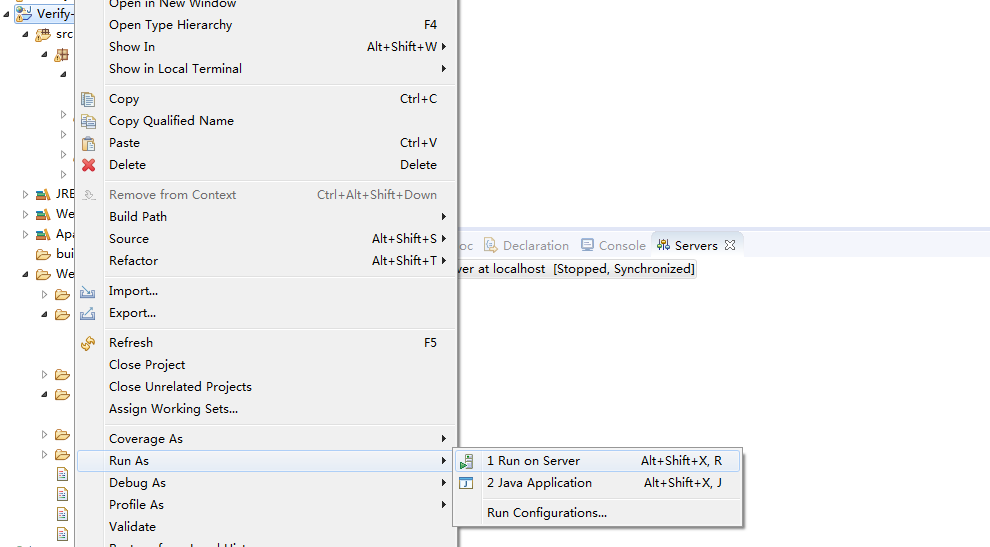
关于运行调试（这里以eclipse+tomcat为例），解压后：

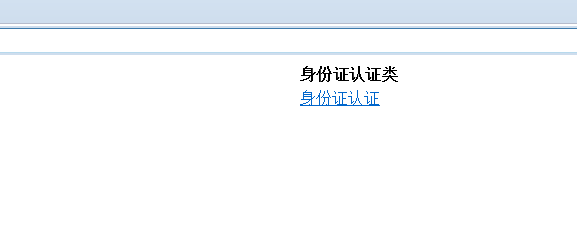
1、



2、

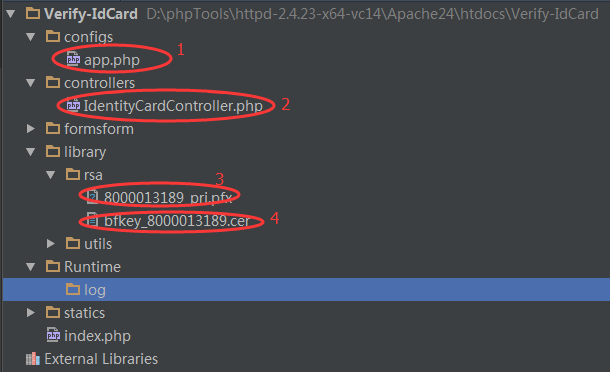
3、



4、

3.3 php样例

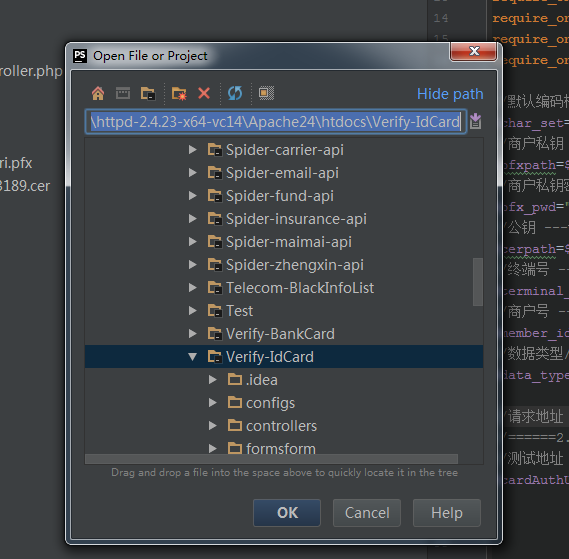
以实名认证项目为例，大概结构为：



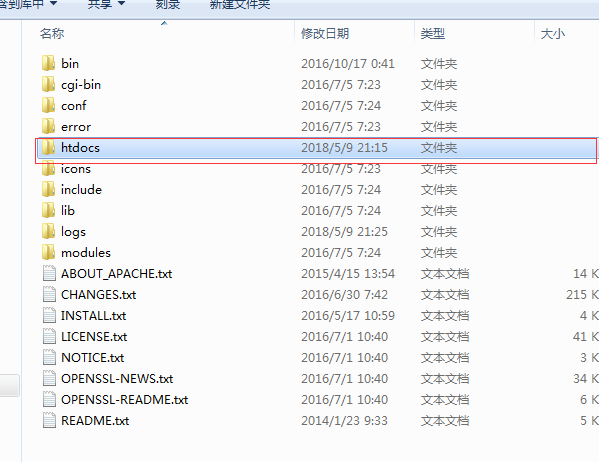
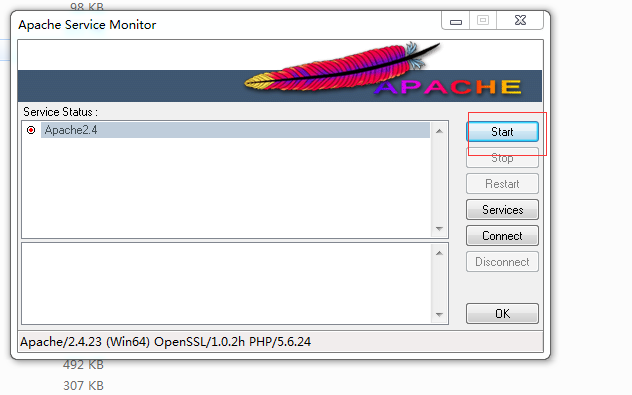
1. 配置文件（商户号、终端号、私钥密码、请求地址等配置）
2. 关键逻辑实现类
3. 私钥文件（对业务参数进行加密）
4. 公钥文件（一般用不到，返回为明文不需要用来解密）

关于运行调试（这里以PhpStorm+ httpd-2.4.23-x64-vc14为例），解压后：

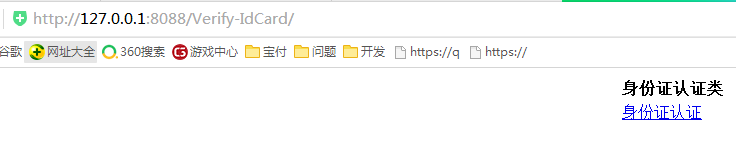
1、



2、

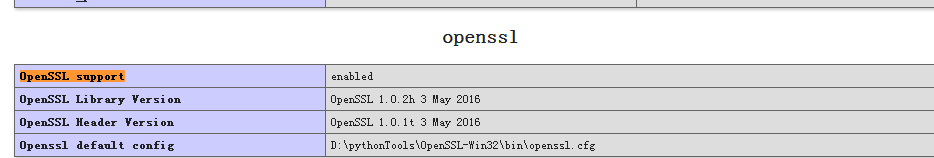
 

3、



注意：由于加密需要openssl库加密，所以需要开启openssl扩展。

通过函数phpinfo()判断，如果如果OpenSSL support是enabled就说明ssl已经开启。



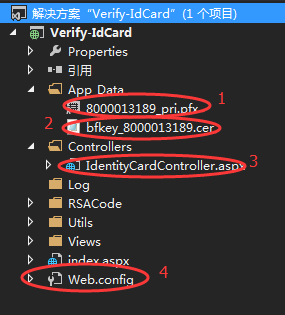
参考资料：

<https://www.cnblogs.com/imysql/p/6370141.html>

http://www.jb51.net/article/49339.htm

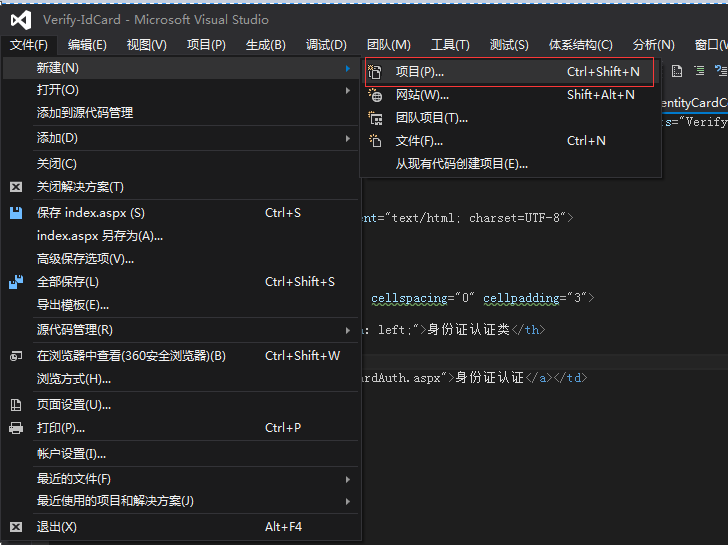
3.4 C#样例

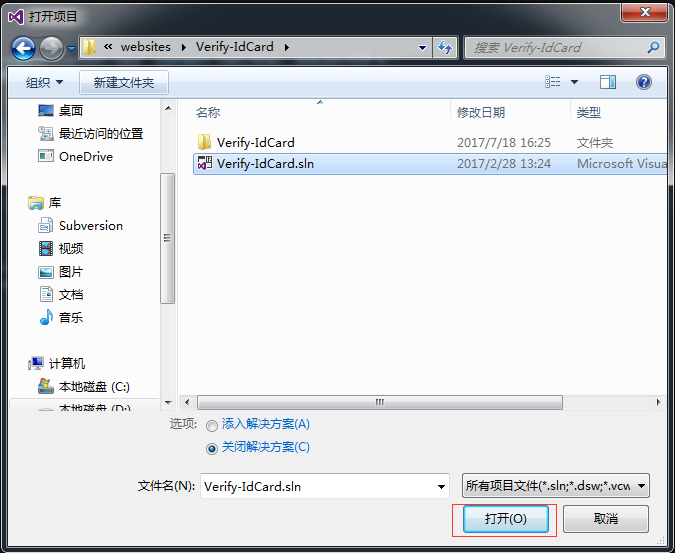
以实名认证产品为例，.Net项目结构为：



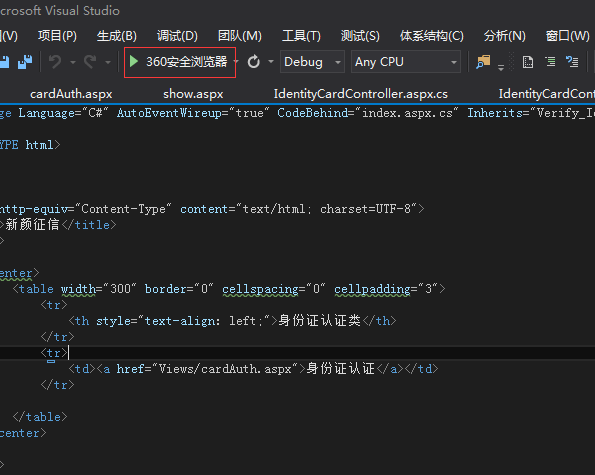
1. 私钥文件（对业务参数进行加密）
2. 公钥文件（一般用不到，返回为明文不需要用来解密）
3. 关键逻辑实现类
4. 配置文件（商户号、终端号、私钥密码、请求地址等配置）

关于运行调试（visual studio为例子），解压后：

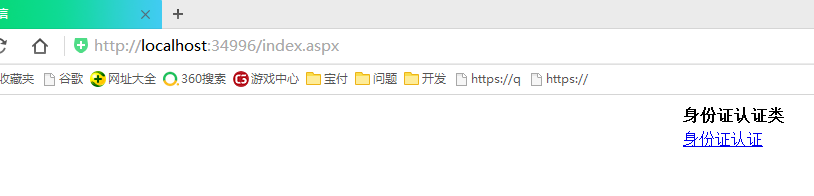
1. 



3.



4.



##### 4.关于加密

4.1 说明

新颜api接口一般对业务参数进行加密传输，一般先把业务参数组装为json,再进行base64,最后用rsa加密是算法进行加密得到加密串。特别说明新颜提供的工具一般生成私钥文件和公钥文件为pfx和cer,适用于java、php、c#语言，其他语言采用pem,详细请见[pfx转pem](#_4.6_pfx转pem).

4.2 java代码片段

*/\*\*  
 \* 私钥算法  
 \** ***@param*** *srcData 源字节  
 \** ***@param*** *privateKey 私钥  
 \** ***@param*** *mode 加密 OR 解密  
 \** ***@return*** *\*/*public static byte[] rsaByPrivateKey(byte[] srcData, PrivateKey privateKey, int mode) {  
 try {  
 Cipher cipher = Cipher.*getInstance*(RsaConst.*RSA\_CHIPER*);  
 cipher.init(mode, privateKey);  
 // 分段加密  
 int blockSize = (mode == Cipher.*ENCRYPT\_MODE*) ? RsaConst.*ENCRYPT\_KEYSIZE* : RsaConst.*DECRYPT\_KEYSIZE*;  
 byte[] decryptData = null;  
 for (int i = 0; i < srcData.length; i += blockSize) {  
 byte[] doFinal = cipher.doFinal(ArrayUtils.*subarray*(srcData, i, i + blockSize));  
 decryptData = ArrayUtils.*addAll*(decryptData, doFinal);  
 }  
 return decryptData;  
 } catch (NoSuchAlgorithmException e) {  
 *log*.error("私钥算法-不存在的解密算法:", e);  
 } catch (NoSuchPaddingException e) {  
 *log*.error("私钥算法-无效的补位算法:", e);  
 } catch (IllegalBlockSizeException e) {  
 *log*.error("私钥算法-无效的块大小:", e);  
 } catch (BadPaddingException e) {  
 *log*.error("私钥算法-补位算法异常:", e);  
 } catch (InvalidKeyException e) {  
 *log*.error("私钥算法-无效的私钥:", e);  
 }  
 return null;  
}

}

具体的细节有demo提供。

4.3 php代码片段

// 私钥加密  
**function** encryptedByPrivateKey($data\_content)  
{  
 $data\_content = base64\_encode($data\_content);  
 $encrypted = "";  
 $totalLen = strlen($data\_content);  
 $encryptPos = 0;  
 **while** ($encryptPos < $totalLen) {  
 openssl\_private\_encrypt(substr($data\_content, $encryptPos, 117), $encryptData, $this->private\_key);  
 $encrypted .= bin2hex($encryptData);  
 $encryptPos += 117;  
 }  
 **return** $encrypted;  
}

4.4 c#代码片段

public static string EncryptRSAByPfx(string src,string path,string passwd){

try

{

AsymmetricKeyParameter PrivteKey = RsaReadUtil.getPrivateKeyFromFile(path, passwd);//读取私钥

byte[] string64 = Base64.Encode(System.Text.Encoding.UTF8.GetBytes(src));//Base64编码 字符编码UF8

// Log.LogWrite("【Base64编码】" + System.Text.Encoding.UTF8.GetString(string64));

string HEX = Hex.ToHexString(RSAEDCore(string64, PrivteKey, true));//加密并转成十六进制

return HEX;

}

catch (Exception ex)

{

return "{[本地加密异常][EncryptRSAByPfx][" + ex.Message + "]}";

}

}

4.5 python加密

备注：如果使用python3.\* 开发强烈不建议用windows系统安装M2Crypto，基本不会安装成功。可以使用vmware workstation或者virtualbox安装liunx或者Mac OS系统。

**Python 2.\***

**================================**

**import** base64  
**import** M2Crypto  
**class RsaUtil**:  
  
 @staticmethod  
 **def encrypt**(digest, private\_key):  
 digest=base64.b64encode(digest)  
 result = ""  
 **while** (len(digest) > 117):  
 some = digest[0:117]  
 digest = digest[117:]  
 result += private\_key.private\_encrypt(some, M2Crypto.RSA.pkcs1\_padding).encode("hex")  
  
 result += private\_key.private\_encrypt(digest, M2Crypto.RSA.pkcs1\_padding).encode("hex")  
  
 **return** result  
  
**if** \_\_name\_\_ == "\_\_main\_\_":  
 private\_key = M2Crypto.RSA.load\_key('8000013189\_pri.pem')  
 result= RsaUtil.encrypt("123456",private\_key)  
 **print** result

Python 3.\*

import base64

import M2Crypto

class RsaUtil:

private\_key = M2Crypto.RSA.load\_key('8000013189\_test.pem')

@staticmethod

def encrypt(digest, private\_key):

digest=base64.b64encode(digest.encode('utf-8'))

result = ""

while (len(digest) > 117):

some = digest[0:117]

digest = digest[117:]

result += private\_key.private\_encrypt(some, M2Crypto.RSA.pkcs1\_padding).hex()

result += private\_key.private\_encrypt(digest, M2Crypto.RSA.pkcs1\_padding).hex()

return result

if \_\_name\_\_ == "\_\_main\_\_":

rsaUtil=RsaUtil()

cc=rsaUtil.encrypt("123456",RsaUtil.private\_key)

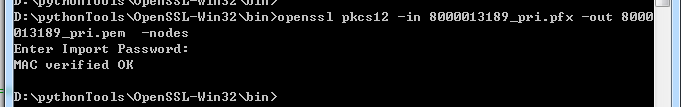
print(cc)

4.6 node js加密

**const** fs=require('fs');  
**const** crypto=require('crypto');  
**const** privateKey=fs.readFileSync('8000013189\_pri.pem','utf-8');  
console.log(privateKey);  
  
**function** encrypt(stuff) {  
 let encryptString = Buffer.from(Buffer.from(stuff).toString('base64'));  
 let result ='';  
 **while** (encryptString.length > 117) {  
 **const** chunk = encryptString.slice(0, 117);  
 encryptString = encryptString.slice(117);  
 result += crypto.privateEncrypt(privateKey, chunk).toString('hex');  
 }  
 result += crypto.privateEncrypt(privateKey, encryptString).toString('hex');  
 **return** result;  
}  
  
console.log(encrypt("123456"));

4.7 pfx转pem

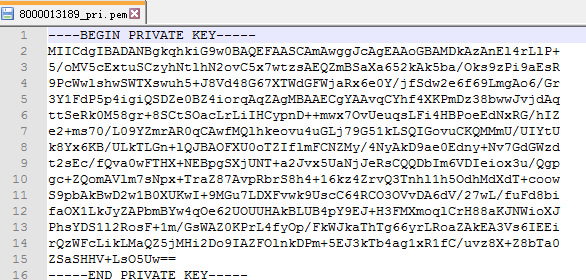
1.在openssl下使用命令：openssl pkcs12 -in 8000013189\_pri.pfx -out 8000013189\_pri.pem -nodes ，输入密码 例如：



2.需要去掉多余的字符串



变为：



只需要”----BEGIN PRIVATE KEY-----“ 到 “-----END PRIVATE KEY-----” 内容

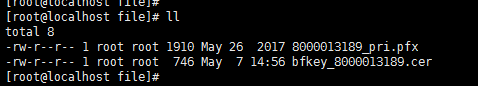
##### 5.注意事项

5.1 私钥（.pfx）文件存放路径

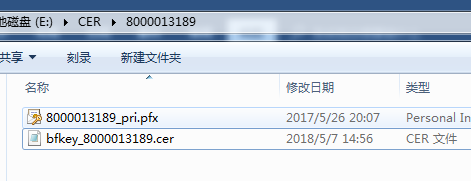
建议把私钥文件的存储路径放在独立的目录或单独的存放在文件服务器，这样的好处：1、需要更换密钥不需要发布项目或者影响其他的配置。2、对于需要用到maven打包的项目时不会因为打包改变文件导致加密失败。

例如：

Liunx系统: /data/file



Windows：E:\CER\8000013189



5.2 加密失败或者加密内容为空

1. 检查私钥文件和私钥密码是否是一对，如果不是一对不能加密出数据。

在windows下可以采用双击私钥文件输入私钥密码导入，如果能导入则为一致，如提示【输入的密码不正确】则不能加密成功，例如：



在liunx下则可以在openssl 执行命令：openssl pkcs12 -nodes -nokeys -in 8000013189\_pri.pfx -passin pass:123456 -out test.cer

错误的情况：Mac verify error: invalid password?



正确的情况：MAC verified OK



1. 报错：DerInputStream.getLength(): lengthTag=111, too big和加密数据为空

可能在maven项目中密钥文件统一放到项目src/main/resources的某个目录下，打包部署运行中出现。原因为maven打包中没有排除密钥文件，证书就会被修改。办法一：推荐把密钥文件存在单独的目录，不参与maven打包。办法二：可以采用过滤证书文件的方式，但是不推荐。例如：

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-resources-plugin</artifactId>

<configuration>

<encoding>UTF-8</encoding>

<!-- 过滤后缀为cer、pfx的证书文件 -->

<nonFilteredFileExtensions>

<nonFilteredFileExtension>cer</nonFilteredFileExtension>

<nonFilteredFileExtension>pfx</nonFilteredFileExtension>

</nonFilteredFileExtensions>

</configuration>

</plugin>

5.3 \*\*\*\*不能为空

如果在调用中出现：商户号不能为空、加密数据不能为空、终端号不能为空、加密数据类型不能为空循环出现。这样的报错是由于post提交数据的方式不对导致新颜服务端没有接收到数据随机提示错误,具体参考文档和Demo。具体提交方式为：

认证类产品、信用评估类产品提交方式为：Content-Type: application/x-www-form-urlencoded;charset=utf-8

授权查询类产品：Content-Type: application/json;charset=utf-8

参考资料：https://blog.csdn.net/tycoon1988/article/details/40080691

5.4 新颜科技接口地址和ip

新颜科技正式环境地址和测试环境地址一般的区别二级域名不一致，正式环境地址为：<https://api.xinyan.com>， 测试环境地址为：<https://test.xinyan.com>。 新颜科技不需要上报ip设置白名单。

|  |  |  |  |
| --- | --- | --- | --- |
| 环境 | 域名 | 域名对应ip | 出口ip |
| 正式环境 | https://api.xinyan.com | 103.47.136.198  222.73.110.180  116.213.96.24  106.75.138.230 | 103.47.136.99  222.73.110.131  **112.65.201.106** |
| 测试环境 | <https://test>.xinyan.com | 116.236.217.148 | 116.236.217.150 |

##### 6. Q&A

6.1 Q:新颜科技测试环境数据是真实校验么？

A:测试环境数据都是模拟的数据（除了爬虫类产品），为了前期实现业务逻辑代码走通流程。

6.2 Q:出现请求报文解析失败有哪些可能？

A:因为我们数据采用加密传输，解密失败为不能解密到传输过来的密文。一般可能的原因是：1、用测试环境密钥加密请求生产环境，生成环境没有上传解密用的公钥。2、对参数进行base64后有换行符，需要把换行符替换掉。3、加密的私钥和解密的公钥不是一对。

6.3 Q:商户号不存在有哪些情况？

A:一般为商户号与请求环境对应不上，先检查下商户号对应的相应环境，测试商户号只能请求测试环境，正式商户号只能请求正式环境 。

6.4 Q:为什么出现订单不能重复提交？

A:在同一商户号下，每笔请求的商户订单号需要保证唯一性，只能请求一次。

6.5 Q:商户号、终端号是什么？在哪里能找到？

A:商户号、终端号是新颜为了区别每个商户分配的系统参数。测试环境在Demo的配置文件中，正式环境会在签订协议后，发送到业务联系人的邮箱中。

6.6 Q:行业类别这个参数该怎么填？是否影响返回结果？

A:行业类别参数可以到接口文档上找到自己公司对应的类别，传入相应的参数，不影响返回结果。

6.7 Q:忘记登录密码可以在哪里找回？

A:可以登录新颜开放平台（https://open.xinyan.com）修改密码和查看订单。



6.8 Q:新颜科技订单记录可以在哪里查看？

A：在开发平台（https://open.xinyan.com），账户管理---->征信订单下查看。

