# 首信易支付接口规范

## 二级商户入网接口

版本说明

| 时间 | 版本号 | 作者 | 说明 |
| --- | --- | --- | --- |
| 2019-5-30 | v1.0 | 技术支持 | 初始版本 |
| 2019-6-17 | V1.1 | 技术支持 | 新增二级商户结算配置，二级商户计费信息配置，二级商户结算配置查询接口 |

# 1.引言

本部分对首信易支付综合接口进行详细地描述，通过该文档可以对本接口有个全面地了解，使商户技术人员尽快掌握本接口，并能够在此基础上进行开发。

## 1.1 概述

本部分详细介绍了首信易支付综合接口进行了详细描述。接口采用 https+数据签名的方式来保证商户与交易平台间的身份验证、中间信息传递的完整性，以便进行电子商务安全当中非常重要的交易身份辨识、不可抵赖、防止篡改等功能。首信易人民币系统采用的是 CFCA 证书和 AES 加密算法，保证数据安全

## 1.2 使用对象

首信易支付商户的网上应用开发人员、维护人员和管理人员，他们应具备以下基本知识:

1. 了解上述系统上的网站设置和网页制作方法;

2. 了解HTML语言以及CGI(CommonGatewayInterface)或ASP(ActiveServerPages)的开发方法或JAVA、PHP、.NET 等开发语言;

3. 了解信息安全的基本概念。

## 1.3 需求栏

| 标记 | 含义 |
| --- | --- |
| M | 必填 |
| C | 有条件的 |
| O | 可选 |

## 1.4 数据提交和接收的方式

首信易支付所有的数据提交和接收的方式皆是以 post 方式提交和接收，并且数据皆以 aes 加密（请求 json）后的格式来提交到接口和接收异步回调，我们的异步回调是以流的方式返回的（aes 加密后格式）。
请求头：ContentType = "application/vnd.5upay-v3.0+json";

应答机制（订单异步返回）是指当贵公司系统收到首信易的支付成功或者取消数据通知（服务器点对点通讯形式）时，必须输出“SUCCESS”首信易支付收到 “SUCCESS”，便认为贵公司已收到；否则将继续发送通知，以递增的时间间隔再次重发 4 次；时间间隔分别为 20/30/40/50秒，总时长 140 秒，以确保订单通知成功。

## 1.5 技术支持

一般事务咨询:请访问首信易支付网站或联系首信易支付客服

技术支持邮箱：tech-support@payeasenet.com

技术支持热线：(010)82652626-6903/6955/6956/6957/6907

技术支持时间：9:30-18:30（工作日）

# 2.首信易支付二级商户入网提交接口（商户-->首信易支付，必做接口）

## 2.1 功能描述

首信易支付二级商户入网接口用于平台类商户为平台商品或服务的真实经营者进行开户注册。

## 2.2 接口地址

请求地址：
https://apis.5upay.com/declaration/submerchant/subMerchantDeclare

## 2.3 请求参数列表

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| merchantId | 商户编号 | varchar(9) | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 | M |
| requestId | 订单号 | varchar(50) | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 | M |
| operationType | 操作类型 | 固定值 | CREATE 创建<br>MODIFY 更新 | M |
| subMerchantId | 二级商户编号 | varchar(9) | 修改时需要商户传递，创建时传空 | C |
| signedName | 签约名称 | varchar(150) | 二级商户签约名称，请上传企业工商注册名称 | M |
| email | 邮箱 | varchar(50) | 二级商户邮箱 | M |

| | | | | |
|---|---|---|---|---|
| phone | 手机号 | varchar(11) | 二级商户手机号 | M |
| businessScope | 经营范围 | 固定值 | AIR_TOUR 航空旅游<br>PHYSICAL_COMMODITY 实物商品<br>VIRTUAL_PRODUCT 虚拟商品<br>PUBLIC_UTILITIES 公共事业<br>COMMUNITY_SERVICES 社区服务<br>INSURANCE 保险<br>EXAM_ORIENTED_EDUCATION 考试教育<br>OTHER 其他 | M |
| authUserId | 实名认证流水号 ID | varchar(32) | 商户调用实名认证接口后返回的流水号(serialNumber参数)，法人证件类型是身份证时，入网必须传递此参数。 | C |
| authEnterpriseId | 工商认证流水号 ID | varchar(32) | 商户调用企业认证接口后返回的流水号(serialNumber 参数) | C |
| hasAgreeProtocol | 是否同意电子协议 | 固定值 | TRUE 是<br>FALSE 否 | M |
| declarationSubMerchantCertificate | 二级商户资质信息 | Json | 见表 2.3.1 | M |
| declarationSubMerchantRiskDocs | 二级商户信息风险评级题目信息 | Json 数组 | 见表 2.3.2 | M |
| hmac | 参数签名 | varchar(500) | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 | M |
| **本接口资质信息上传需要配合 ftp 使用，ftp 连接方式见表 7** | | | | |

## 2.3.1 请求参数（declarationSubMerchantCertificate，二级商户资质信息）

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| registerRole | 签约身份 | 固定值 | ENTERPRISE_LEGAL_PERSON 企业法人<br>INDIVIDUAL_BUSINESS 个体工商户<br>NATURAL_PERSON 自然人 | M |
| businessLice | 营业执照照 | varchar(255) | 营业执照照片上传到 ftp 的路径，签 | C |

| | | | | |
|---|---|---|---|---|
| nsePath | 片路径 | | 约身份为自然人时非必填 | |
| openAccountPath | 开户许可证照片路径 | varchar(255) | 开户许可证照片上传到 ftp 的路径 | O |
| legalIdCardProsPath | 法人证件正面照路径 | varchar(255) | 法人证件正面照上传到 ftp 的路径 | M |
| legalIdCardConsPath | 法人证件反面照路径 | varchar(255) | 法人证件反面照上传到 ftp 的路径 | M |
| settleBankCardPath | 结算卡照片路径 | varchar(255) | 结算卡照片上传到 ftp 的路径，签约身份为自然人时必填 | C |
| otherCerPath | 其他资质文件路径 | varchar(255) | 其他资质文件上传到 ftp 的路径 | O |
| name | 法人姓名 | varchar(50) | 法人姓名 | M |
| idType | 法人证件类型 | 固定值 | IDCARD 身份证<br>PASSPORT 护照<br>OFFICERPASS 军官证<br>ARMEDPOLICE 武警证<br>HM_VISITORPASS 港澳居民往来内地通行证<br>RESIDENCEBOOKLET 户口簿<br>T_VISITORPASS 台湾居民来往大陆通行证<br>FOREIGNESTAYFOREVER 外国人永久居留证<br>OTHER 其它证件号<br>（签约身份为自然人时必须选身份证） | M |
| idNo | 法人证件号 | varchar(255) | 法人证件号 | M |
| effectiveDateStart | 法人证件有效期开始日期 | varchar(10) | 法人证件有效期开始日期<br>yyyy-MM-dd | M |
| effectiveDateEnd | 法人证件有效期结束日期 | varchar(10) | 法人证件有效期结束日期<br>yyyy-MM-dd | M |

### 2.3.2 请求参数（declarationSubMerchantRiskDocs，二级商户信息风险评级题目信息）

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| docCode | 题目名称编码 | 固定值(四个全部需要) | REGISTER_AMOUNT 注册资金<br>STAFF_SIZE 公司员工规模<br>OFFICE_SPACE 办公面积 | C |
| answer | 答案 | 固定值 | **传递参数：A B C**<br><br>**各项题目答案：**<br>**注册资金：**<br>A.人民币 500 万以上，或等值外币<br>B.人民币 100 万-500 万(含)，或等值外币<br>C.人民币 100 万以下或等值外币及自然人商户<br>**公司员工规模：**<br>A.100 人以上<br>B.50-~100 人()含<br>C.50 人以下<br>**办公面积：**<br>A.500 平米以上<br>B.200-500 平米(含)<br>C.200 平米以下 | C |

## 2.4 示例

```
{
    "authEnterpriseId": "2c9348a334gd78bd0162s5g7edce000c",
    "authUserId": "2c9348a36a1aees0016a66e83sc80017",
    "businessScope": "AIR_TOUR",
    "declarationSubMerchantCertificate": {
        "businessLicensePath": "/transfer/details/111.png",
        "effectiveDateEnd": "2019-10-10",
        "effectiveDateStart": "2018-10-10",
        "idNo": "4105211991XXXXXXXX",
```

```
            "idType": "OFFICERPASS",
            "legalIdCardConsPath": "/transfer/details/444.png",
            "legalIdCardProsPath": "/transfer/details/333.png",
            "name": "姓名",
            "openAccountPath": "/transfer/details/222.png",
            "otherCerPath": "/transfer/details/555.png",
            "registerRole": "ENTERPRISE_LEGAL_PERSON"
        },
        "declarationSubMerchantRiskDocs": [{
            "answer": "A",
            "docCode": "REGISTER_AMOUNT"
        }, {
            "answer": "A",
            "docCode": "STAFF_SIZE"
        }, {
            "answer": "A",
            "docCode": "OFFICE_SPACE"
        }
],
        "email": "qjytest002@qq.com",
        "hasAgreeProtocol": "TRUE",
        "hmac":
"N7tuOHP/93OXjFjQJpY88Cv+lLaT2xzGkm1n3mYYNuVpGOMMRzR/LtpDB58prkjROOEQy
B8QnJraKJi8N1LwMHrMg4RVYN1UFWQaw22WKCxW/dr1rxrjVAdNXoBZxpNVMoZH9Up
Y74rTXFm73rYGxxLdTBieT3w712ofhQief2Co/sX16B2EDGdsNqVeA7ew0NLDMhFHDRNf1
Dbptzm60/SXhNoCn9rcJmuYTadF4DIiD0VTEsZY0m6ClBWey/Z7F0ywSXGbL6/YQczLkXrdL
o6abhyCXY5jRebGbfNp750uzrfdBGdFv09b009ORz492JDok4hcpoWWElpC5ZOxug==",
        "merchantId": "896666748",
        "operationType": "CREATE",
        "phone": "17600101156",
        "requestId": "1559204294738",
        "signedName": "北京三快科技有限公司",
        "subMerchantId": ""
}
```


## 2.5 请求同步返回参数列表

| 参数名称 | 参数中文名 | 参数说明 |
| --- | --- | --- |

| | 称 | |
|---|---|---|
| merchantId | 商户编号 | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 |
| requestId | 订单号 | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 |
| subMerchantId | 二级商户编号 | 又首信易返回的二级商户编号 |
| status | 响应状态 | 成功 SUCCESS,<br>失败 FAILED,<br>错误 ERROR, |
| errorMessage | 错误信息 | 错误信息 |
| hmac | 参数签名 | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 |

## 2.6 示例

```
{
    "merchantId": "890000593",
    "requestId": "1527746602492",
    "subMerchantId": "890000593",
    "hmac": "959e138ad2e290328e8d59fd124c618f",
    "status": "SUCCESS"
}
```

# 3.首信易支付二级商户入网查询接口（商户-->首信易支付，必做接口）

## 3.1 功能描述

商户通过二级商户入网查询接口查询首信易支付平台的二级商户入网状态。

## 3.2 接口地址

请求地址：
https://apis.5upay.com/declaration/submerchant/subMerchantQuery

## 3.3 请求参数列表

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| merchantId | 商户编号 | varchar(9) | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 | M |
| subMerchantId | 二级商户编号 | varchar(9) | 由首信易创建的二级商户编号。 | M |
| hmac | 参数签名 | varchar(500) | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 | M |

## 3.4 示例

```
{
    "merchantId": "890000593",
    "subMerchantId": "890000593",
    "hmac": null
}
```

## 3.5 入网查询返回参数列表

| 参数名称 | 参数中文名称 | 参数说明 |
|---|---|---|
| merchantId | 商户编号 | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 |
| subMerchantId | 二级商户编号 | 由首信易创建的二级商户编号。 |
| subMerchantReviewStatus | 二级商户审核状态 | INIT 待审核<br>SUCCESS 审核通过<br>FAIL 审核不通过<br>REFUSE 审核拒绝 |
| subMerchantReviewRemark | 二级商户审核备注信息 | 审核意见 |

| | | |
|---|---|---|
| status | 响应状态 | 错误 ERROR,<br>成功 SUCCESS,<br>失败 FAILED; |
| errorMessage | 错误信息 | 错误信息 |
| hmac | 参数签名 | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 |

### 3.6 示例

```
{
    "subMerchantId": "890000593",
    "subMerchantReviewRemark": "审核通过",
    "subMerchantReviewStatus": "SUCCESS",
    "merchantId": "890000593",
    "status": "SUCCESS"
}
```

# 4.二级商户结算配置

## 4.1 概述

商户给二级商户进行配置结算信息额接口，能够对公和对私，可以选择定期结算和自助结算。

## 4.2 接口地址

请求地址：
https://apis.5upay.com/declaration/submerchant/settlementProfile/order

## 4.3 请求参数列表

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| merchantId | 商户编号 | varchar(9) | 商户在首信易系统的唯一身份标识， | M |

| | | | | |
|---|---|---|---|---|
| | | | 商户完成首信易系统注册后可登录商户后台商户服务查看。 | |
| requestId | 订单号 | varchar(50) | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 | M |
| subMerchantId | 二级商户编号 | varchar(9) | 由首信易创建的二级商户编号。 | M |
| settleMode | 结算模式 | 固定值 | SETTLE 手动结算<br>REGULAR_SETTLE 定期结算<br>WITHDRAW 提现 | M |
| cycleType | 周期类型 | 固定值 | BYWEEK 周结<br>BYMONTH 月结<br>结算模式为提现时可不传此参数 | C |
| cycleData | 结算日期 | varchar(55) | 周结：1-7<br>月结：1-28<br>以上两种方式皆可传多项数据，中间用英文逗号分隔，例：1,3,6(如是周结该结算方式为每周一、周三、周六进行结算，如是月结该计算方式为每月 1 号、3 号、6 号进行结算)<br>结算模式为提现时可不传此参数 | C |
| minAmount | 起结金额 | varchar(6) | 设置起结金额，分为单位，最小为 1 | M |
| recName | 收款方账户名称 | varchar(100) | 收款方账户名称 | M |
| recBankAccountNo | 收款方账户号 | varchar(19) | 收款方账户号 | M |
| recIdCardNum | 收款人身份证号 | varchar(18) | 收款人身份证号，账户为个人时必传 | C |
| recBankPhoneNum | 银行预留手机号 | varchar(11) | 收款方银行预留手机号，对公账户可传 | C |
| recBankCode | 收款账户银行代码 | varchar(10) | 收款方银行代码，见编码表.xlsx | M |
| recBranchBankName | 开户行支行名称 | varchar(50) | 收款方支行名称 | M |
| recProvinceCode | 收款方省编码 | varchar(2) | 收款方省编码，见编码表.xlsx | M |
| recCityCode | 收款方市编码 | varchar(4) | 收款方市编码，见编码表.xlsx | M |
| recAccountType | 收款人账户类型 | 固定值 | TO_PUBLIC 对公<br>TO_PRIVATE 对私 | M |
| hmac | 参数签名 | varchar(500) | 获取 hmac 的方法见表 8，参数顺序按 | M |

| | | | | |
|---|---|---|---|---|
| | | | 照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 | |

## 4.4 示例

| |
|---|
| |

## 4.5 返回参数列表

| 参数名称 | 参数中文名称 | 参数说明 |
|---|---|---|
| merchantId | 商户编号 | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 |
| requestId | 订单号 | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 |
| subMerchantId | 二级商户编号 | 由首信易创建的二级商户编号。 |
| status | 响应状态 | 错误 ERROR,<br>成功 SUCCESS,<br>失败 FAILED; |
| serialNumber | 订单流水号 | 二级商户结算配置订单流水号 |
| errorMessage | 错误信息 | 错误信息 |
| hmac | 参数签名 | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 |

## 4.6 示例

| |
|---|
| |

# 5.二级商户计费信息配置

## 5.1 概述

商户给二级商户进行配置计费信息的接口，能够内扣和外口，计费类型可以单笔固定金额和百分比金额。

## 5.2 接口地址

请求地址：
https://apis.5upay.com/declaration/submerchant/billing/order

## 5.3 请求参数列表

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| merchantId | 商户编号 | varchar(9) | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 | M |
| requestId | 订单号 | varchar(50) | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 | M |
| subMerchantId | 二级商户编号 | varchar(9) | 由首信易创建的二级商户编号。 | M |
| billingMode | 计费模式 | 固定值 | REAL_TIME_OUTSIDE 外扣<br>REAL_TIME_INSIDE 内扣 | M |
| feeType | 收费类型 | 固定值 | SINGLE 单笔固定金额<br>ORDER_AMOUNT_PERCENTAGE 订单金额百分比 | M |
| feeValue | 收费值 | long | 单笔固定金额：商户自行配置金额<br>订单金额百分比：0-10000 | O |
| minFeeAmount | 最小收费金额 | long | 分为单位，最小收费金额，按照订单金额百分比扣费时可传此参数 | O |
| maxFeeAmount | 最大收费金额 | long | 分为单位，最大收费金额，按照订单金额百分比扣费时可传此参数 | O |
| hmac | 参数签名 | varchar(500) | 获取 hmac 的方法见表 8，参数顺序按 | M |

| | | | | |
|---|---|---|---|---|
| | | | 照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 | |

## 5.4 示例

| |
|---|

## 5.5 返回参数列表

| 参数名称 | 参数中文名称 | 参数说明 |
|---|---|---|
| merchantId | 商户编号 | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 |
| requestId | 订单号 | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 |
| subMerchantId | 二级商户编号 | 由首信易创建的二级商户编号。 |
| status | 响应状态 | 错误 ERROR,<br>成功 SUCCESS,<br>失败 FAILED; |
| serialNumber | 订单流水号 | 二级商户计费配置订单流水号 |
| errorMessage | 错误信息 | 错误信息 |
| hmac | 参数签名 | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 |

## 5.6 示例

| |
|---|

# 6.二级商户结算配置查询

## 6.1 概述

此接口能够查询到二级商户的结算和计费相关配置信息。

## 6.2 接口地址

请求地址：
https://apis.5upay.com/declaration/submerchant/billingRule/query

## 6.3 请求参数列表

| 参数名称 | 参数中文名称 | 类型& 长度 | 参数说明 | 是否必填 |
|---|---|---|---|---|
| merchantId | 商户编号 | varchar(9) | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 | M |
| requestId | 订单号 | varchar(50) | 订单号为商户自行拟定，提交的订单号必须在自身平台交易中唯一。商户平台不能以相同的订单号再次提交。 | M |
| subMerchantId | 二级商户编号 | varchar(9) | 由首信易创建的二级商户编号。 | M |
| hmac | 参数签名 | varchar(500) | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 | M |

## 6.4 示例

```
{
    "merchantId": "890000593",
    "requestId":"123412341234"
    "subMerchantId": "890000593",
    "hmac": null
}
```

## 6.5 返回参数列表

| 参数名称 | 参数中文名称 | 参数说明 |
|---|---|---|
| merchantId | 商户编号 | 商户在首信易系统的唯一身份标识，商户完成首信易系统注册后可登录商户后台商户服务查看。 |
| serialNumber | 订单流水号 | 查询订单流水号。 |
| errorMessage | 错误信息 | 错误信息 |
| recName | 收款账户名称 | 收款账户名称 |
| recIdCardNum | 收款人身份证号码 | 收款人身份证号码。 |
| recAccountType | 收款方银行代码 | TO_PUBLIC 对公<br>TO_PRIVATE 对私 |
| recBankCode | 收款方银行代码 | 收款方银行代码，见编码表.xlsx |
| recBranchBankName | 开户银行分行名称 | 开户银行分行名称 |
| recBankAccountNo | 收款方银行账户号 | 收款方银行账户号 |
| recProvinceCode | 收款方行省编码 | 收款方行省编码，见编码表.xlsx |
| recCityCode | 收款方行市编码 | 收款方行市编码，见编码表.xlsx |
| settleMode | 结算模式 | SETTLE 手动结算<br>REGULAR_SETTLE 定期结算<br>WITHDRAW 提现 |
| minAmount | 最小结算金额 | 最小结算金额 |
| cycleType | 周期类型 | BYWEEK 周结<br>BYMONTH 月结 |
| cycleData | 结算日期 | 周结：1-7<br>月结：1-28<br>以上两种方式皆可传多项数据，中间用英文逗号分隔，例：1,3,6(如是周结该结算方式为每周一、周三、周六进行结算，如是月结该计算方式为每月 1 号、3 号、6 号进行结算) |
| billingRule | Json | 计费信息，见表 6.5.1 |

| | | |
|---|---|---|
| errorMessage | 错误信息 | 错误信息 |
| hmac | 参数签名 | 获取 hmac 的方法见表 8，参数顺序按照键名首字母从 a-z 拼接对应的键值中间用#号隔开。 |

### 6.5.1 返回参数（billingRule，计费信息）

| 参数名称 | 参数中文名称 | 参数说明 |
|---|---|---|
| billingMode | 计费模式 | REAL_TIME_OUTSIDE 外扣<br>REAL_TIME_INSIDE 内扣 |
| feeType | 收费类型 | SINGLE 单笔固定金额<br>ORDER_AMOUNT_PERCENTAGE 订单金额百分比 |
| feeValue | 收费值 | 单笔固定金额：商户自行配置金额<br>订单金额百分比：0-100 |
| minFeeAmount | 最小收费金额 | 最小收费金额，按照订单金额百分比扣费时可传此参数 |
| maxFeeAmount | 最大收费金额 | 最大收费金额，按照订单金额百分比扣费时可传此参数 |

## 6.6 示例

| |
|---|
| |

# 7.首信易支付资质信息上传 sftp 相关信息

## 7.1 概述

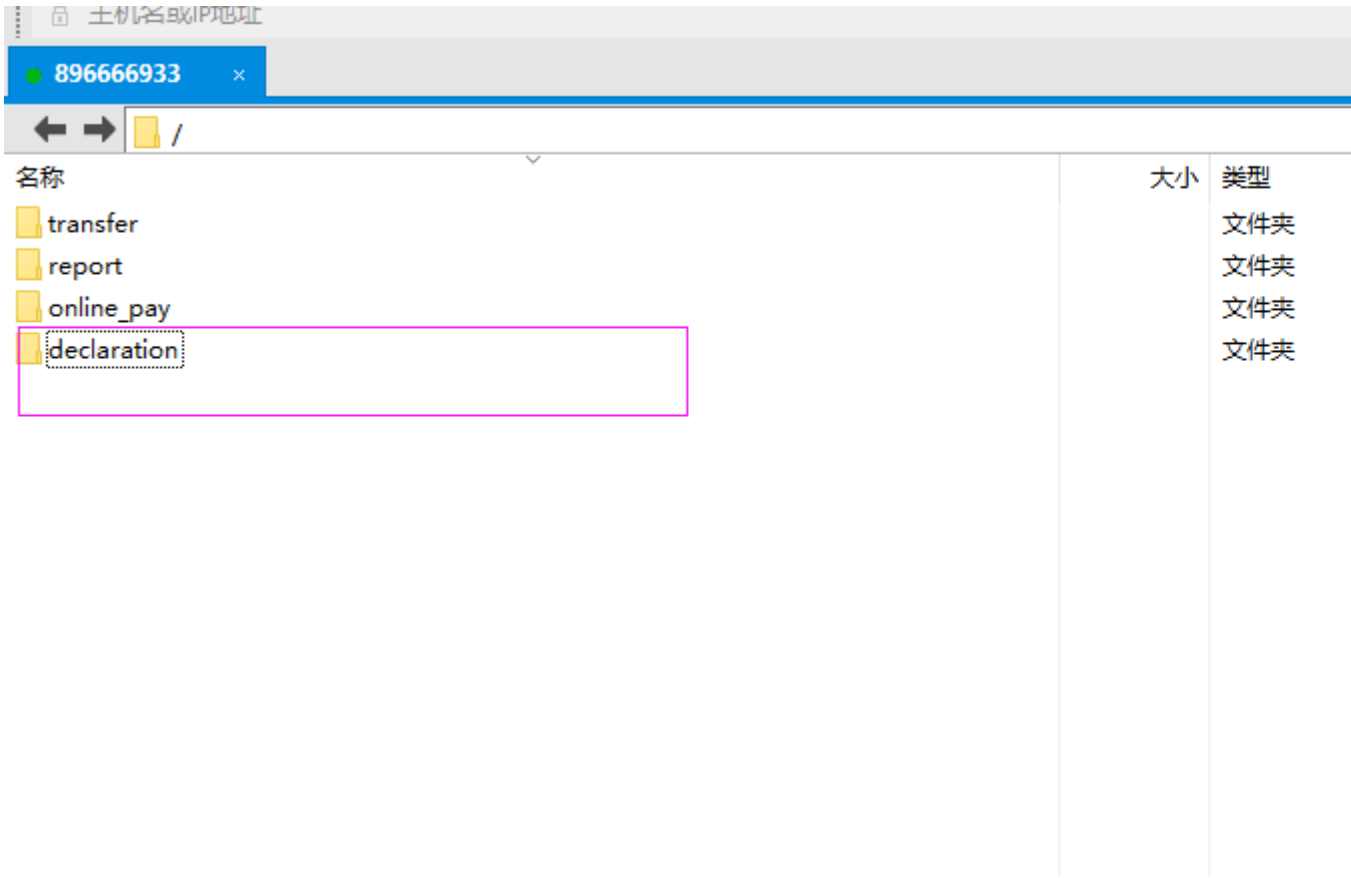
商户上传资质等信息需要上传至指定 sftp，以下是 sftp 相关链接方式以及路径信息，商户需要先从首信易方面获取到登陆密钥文件。

## 7.2 连接方式

主机：merchant-sftp.5upay.com
端口：2822
协议：SFTP
用户名：商编
密钥：密钥文件
密码：不需要密码
示例：


## 7.3 文件目录

目录:/declaration/




商户可以在 declaration 下面创建子文件夹用来区分各个不同的二级商户。

# 8.首信易支付各加密/解密算法以及排序手法的相关说明

## 请求加密详细说明

为了数据安全，首信易支付接口用了两种加密方法、两种签名方法以及一种排序方法，分别是：CFCA 公钥证书加密、AES 加密、SHA1 签名、CFCA 私钥证书签名和键名首字母排序，首信易的三种语言 demo 分别都提供了各个方法，具体用法：

**键名首字母排序**：该方法的主要作用是将原数据 json 按照键名首字母进行从 a-z 的顺序进行排序，并按照排好的顺序进行拼接键值，各键值中间用#号隔开，例：

排序前：{

"merchantId": "890000595",

"orderAmount": "1",

```
        "orderCurrency": "CNY",
        "requestId": "1556592332569",
        "notifyUrl": "https://api.5upay.com/sdk/onlinepay/notify",
        "callbackUrl": "https://api.5upay.com/sdk/callback",
        "remark": "备注",
        "paymentModeCode": "",
        "productDetails": [{
            "name": "",
            "quantity": 1,
            "amount": 1,
            "receiver": "",
            "description": ""
        }],
        "payer": {
            "name": "付款人姓名",
            "idType": "IDCARD",
            "idNum": "132201198701280426",
            "bankCardNum": "6222021001116245702",
            "phoneNum": "13901234567",
            "email": "cc@cc.com",
            "nationality": null
        },
    "hmac": null
}
```


排序后：

https://api.5upay.com/sdk/callback#890000595#https://api.5upay.com/sdk/onlinepay/notify#1#CNY#6222021001116245702#cc@cc.com#132201198701280426#IDCARD#付款人姓名#13901234567#1#1#备注#1556592332569#

**PS:'#'符号尾加首不加**

**SHA1 签名**：该签名方法用于对数据进行初步摘要。例：

签名前：

https://api.5upay.com/sdk/callback#890000595#https://api.5upay.com/sdk/onlinepay/notify#1#CNY#6222021001116245702#cc@cc.com#132201198701280426#IDCARD#付款人姓名#13901234567#1#1#备注#1556592332569#

签名后：wkJpxZ9o+vuGhQn2pieT3S5GvSo=

**CFCA 私钥签名**：商户使用自身 CFCA 私钥证书（**商户与首信易方面业务人员申请**）进行签名，该签名优点是不可被破解，请求接口后首信易支付使用商户在商户后台上传的公钥进行验签，防止信息篡改。例：

签名前：wkJpxZ9o+vuGhQn2pieT3S5GvSo=

签名后：
SjhLrzimZKXWCz1m1L5npw/rBs6GdTOMSDlg5D49TWdgQUY9+eC2j14TnhopGSk7euAWX7c
WQaMCP096znzKeRFaYp0rsghkQSquzRwFQiVWh/gePJSVdnl5raUHGPW4r0gfRLuKbtMCe2pgi
dMrGM1WTOGjHj9kM/1qaecDO1PXHSDJWNIuypXbsMg1bvkltg1qlfOJgCRJ6IDJSr3vnW4eWV
EZJDqkcYZAU7lZnnq419XJnbXGqefBYW4pBrVlFgyKkP7PG25JGu5b03IuxgKTmp5qeXB0BHZ
zgaVRfrRyNqvRtK2qqCB7/+QI7O1c7wexsF6uR/ekGKArWRCDCw==

**AES 加密**：该加密可通过相同的密钥进行解密，可自行生成 16 位随机数密钥进行加密，首信易支付接收到后通过相同的密钥进行解密，AES 加密作用于加密已经含有 hmac 参数的原数据，例：

加密前：{

```
"merchantId": "890000595",
"orderAmount": "1",
"orderCurrency": "CNY",
"requestId": "1556592332569",
"notifyUrl": "https://api.5upay.com/sdk/onlinepay/notify",
"callbackUrl": "https://api.5upay.com/sdk/callback",
"remark": "备注",
"paymentModeCode": "",
"productDetails": [{
    "name": "",
    "quantity": 1,
    "amount": 1,
    "receiver": "",
    "description": ""
}],
"payer": {
    "name": "付款人姓名",
    "idType": "IDCARD",
    "idNum": "132201198701280426",
    "bankCardNum": "6222021001116245702",
    "phoneNum": "13901234567",
    "email": "cc@cc.com",
    "nationality": null
},
"hmac": "
```

SjhLrzimZKXWCz1m1L5npw/rBs6GdTOMSDlg5D49TWdgQUY9+eC2j14TnhopGSk7euAWX7c
WQaMCP096znzKeRFaYp0rsghkQSquzRwFQiVWh/gePJSVdnl5raUHGPW4r0gfRLuKbtMCe2pgi
dMrGM1WTOGjHj9kM/1qaecDO1PXHSDJWNIuypXbsMg1bvkltg1qlfOJgCRJ6IDJSr3vnW4eWV
EZJDqkcYZAU7lZnnq419XJnbXGqefBYW4pBrVlFgyKkP7PG25JGu5b03IuxgKTmp5qeXB0BHZ
zgaVRfrRyNqvRtK2qqCB7/+QI7O1c7wexsF6uR/ekGKArWRCDCw=="

}

加密后：
PoE9VSppnI/ixnoi7j2/LE0o28NGVdREh6TfZXXPS0Z0xZTs8Iy9FOoL2lDYOOGfoNNb3YRGtQ1y1ggUykbBaMRh8ipd7njD8XgH0suhx0nyKcSuUS6vkCV3rI+0cJyjENozstB7vhAv31LxOoWyhrBCim/9mnyMomi6jEZS0xdQizUu4TcgJsQL2wXCecHdoD6C6PzLS+oSpizoBC6OtMMuD3aPR/tU5dhofU/r1Bof8CuWulV4sCNr8X5EsduZs3uu3dnLsLEJNDjqplN7dGYga492DMg5KHqm8yAOGkiLAyR2jEspHz0ICVdB7RSS6RTDrtqAVzEIw2zrAYyoTMbKOPkh720eFjkRc3iBw5TLfry9ZP4/sBtCzeOvcKw5rxGWiV6RIzxmftUFfsZLAA45DAil6+qryNU7cvS6UupxFiJLykkg88HvZRCGXuY4RxMhTxMT4Vt+emA3RkYHr4U5Z+jUmvx1AjzQs6VBEoqmiFtM5vwHVkkF+e6JYV3d8BVeWyWCcYWYR0h9WYOdvnozsLv5NAQZtFYLbON7krN9abvqUa1Nwn+f8f+kQpyQAqhhLbqx/r2V9jvaIWXYyUzo0oLIH8eL2WvWBPDXMeipo0c0erwxqL3cGI1jxsAvrpAHPoGxYaeRCdWWIWDG2l6hHwiER/pZKp16k0JSOsJVBGapgWvbPFAd/kfoP6SHuRG/dNjgahiS+tK/QUrV+X8D16Q7uixBJ15/Gwg/b2QSC88Dvs9t14UmYqtqVYFZAo4u5O718r4yA20nk8bZCd9Zwv+Hy6IEg1MzzZEYoOoT9oXCvrQEb6SY5JzCymLPOLq07Mi1a5IZ1c9KGWY2gi6UaXFEHGVCLt8ITlHhG5V8NCo8+uMB3l4vFKNAMRC21tc1Z7dFdD29vnU+2sa2oa86JGMDLbe+MXgVEpWsmFouGhcq1uU+xDSgTKlhm87TW0waUihBX/VnmCPrpV0YQgIN/RZMnWAHEf78cWx4UMXEqi+eqJer27CuKG8L2k6sCImskWdIBvVXRpHrEsDqZnzsBDQuQ5f0kPjljqXjhk/vLzYy+READjUwVpKDhwkvPd2JT+j+UZBg+shYia0UNpPc0GdvFCgz0unqZUIhkEf9dHXQBzu+pyIMOFspxvf9O41Otj+Y/cDCxeqIP+KlxJrxLrxr6g868VpWKffKEMivuMZMo528QV0/xUyivzUBxep4OCq6x8zB6poGRaap50PAvbMPEZ9FDJdrSpMkZcGJ5c9w+ET/Pu2xBEv1Pk7ykfyv2FPgM0vfh+xvjhLvAa0YPXvM7xbeB35LQ2iA54T/voza91sNhMowoR7ZcS0xK6+bp+VDCJfAK6kP4dwvCg==

**CFCA 公钥加密：**该加密是不对称加密，商户通过首信易提供的公钥进行 CFCA 公钥证书加密，加密方法在 demo 内有提供，加密的数据是 AES 密钥，防止 AES 密钥泄露。

# 返回解密相关说明

为了数据安全，首信易支付接口用了两种解密方法、一种签名方法、一种验签方法以及一种排序方法，分别是：CFCA 私钥证书解密、AES 解密、SHA1 签名、CFCA 公钥验签和键名首字母排序，首信易的三种语言 demo 分别都提供了各个方法，具体用法：

**CFCA 私钥解密：**该方法的用处是将商户上传到商户后台的 CFCA 公钥加密的 encryptKey 进行解密，得到十六位的 aes 密钥。
返回的请求头 encryptKey:
OsYlS3vP1I+m5csUDTSbi801evXvMDsMejLHtqIqbtA5QTdPTjvCE2q4NLFek53Sqo0HxjcHxxgbX5QMEM7y0VMmeUmaLJ7BK/RR3Zc5xtHkWE/AbO8ErG7oXgORZ6JVakeRlDG8sSVTT2Duvy38RsXUQC4Vj4VHgxsLq398vfYQZgf9O8FPfSc+m7g8MGhbbuqMSbOWBH1lDKnCkpl93XSiWFYHICt7aO/P9avhqlSVJDQg2KpwMld6Mfag5hgz/LTE7DfwyueH6VxIU7rrUkelWZSbO3ULUg

7G5okRAFvXJPqfTnkRnQ+qTsK3QOpb98XlaR+tjuYqzTWs+/mUVg==

解密后得到的 aes 密钥：w4deov41ogHO7eFi

**AES 解密：**该方法的用处是将首信易方面用随机生成的 aes 密钥加密的数据进行解密，得到原数据。
返回的请求体 data:
IsMowPkfHQd/x4w7uq3PQpCrCSir9e1B4075S03gQ0svH2LtOcPogsODNGPdrf5YL9AMvslQOj3iGfS8pkfxxsyYienxAWrwNpu0b49LveB8CvJXBaYOETIbRTYtJ2NaEvJp6vySUo+L0vQZuOb6hxLALr8nU/zk4cZiS2KvGtP1tTR1If64Xfut1qNITtQv

解密后：
{"redirectUrl":"https://payment.5upay.com/receipt/redirect/index/2c9553496a684fe9016a6c53228d7cf1/2c9553496a684fe9016a6c5322af7cf3","merchantId":"890000595","requestId":"1556595531274","hmac":"PhbbgXjh6641/cQ6qfy5Dq10h/2TEH1XJiRLKAmtCDUy/hR0K+KRUvJ3bskYVATF3aDrHPBUz+RZjkWjBUgEd9E/7jrHVjAt/WHKlcxhwwlId1svUcY3oUvJPuh28fHTC8mZ6uOBFLQ5NVy+sT6A6m2g5OWJ//LQMU05WO77mIt62C60qqFRpdXkcfGdkJapyatUFKIJB8S5EKOeGkDusj6wMWSIR+Uhrgrzx/pv6BtUmRv2F1syxRK5BjdLWp/bVUl4bxKjCo/JKW8cRmz2ou/ZbchL3uQxJhjwzJoITEJ1PZmUN5B1yJurtQXR3C62MLCeXFVNFaOP6qC9VbQ8Ew==","paymentOrderId":"2c9553496a684fe9016a6c53228d7cf1","status":"REDIRECT"}

**键名首字母排序：**该方法的主要作用是将去掉了 hmac 的原数据 json 按照键名首字母进行从 a-z 的顺序进行排序，并按照排好的顺序进行拼接键值，各键值中间用#号隔开，例：
排序前：
{
"redirectUrl":
"https://payment.5upay.com/receipt/redirect/index/2c9553496a684fe9016a6c53228d7cf1/2c9553496a684fe9016a6c5322af7cf3",
"merchantId": "890000595",
"requestId": "1556595531274",
"paymentOrderId": "2c9553496a684fe9016a6c53228d7cf1",
"status": "REDIRECT"
}
排序后：
890000595#2c9553496a684fe9016a6c53228d7cf1#https://payment.5upay.com/receipt/redirect/index/2c9553496a684fe9016a6c53228d7cf1/2c9553496a684fe9016a6c5322af7cf3#1556595531274#REDIRECT#

**SHA1 签名：**该签名方法用于对数据进行初步摘要。例：
签名前：

890000595#2c9553496a684fe9016a6c53228d7cf1#https://payment.5upay.com/receipt/redirect/index/2c9553496a684fe9016a6c53228d7cf1/2c9553496a684fe9016a6c5322af7cf3#1556595531274#REDIRECT#
签名后：UmfW5oHZqRwSY5XKdINdfh57FK4=

**CFCA 公钥验签：**该方法的主要作用是商户方面使用首信易提供的统一 CFCA 公钥对首信易方面返回的 hmac 数据进行验签。

# 键名首字母自动排序

## PHP

```
/**
        *参数排序
        * @return string
        */
       protected function buildJson($para=null)
       {
            $vars = $para?":get_object_vars($this);
            unset($vars['response_hmac_fields']);
            $data = array();
            foreach($vars as $k=>$var){
                 if(is_scalar($var) && $var !== '' && $var !== null){
                      $data[$k] = $var;
                 }else if(is_object($var) && $var instanceof AbstractModel){
                      $data[$k] =array_filter((array) $var);
                 }else if(is_array($var)){
                      $data[$k] =array_filter($var);
                 }
                 if(empty($data[$k])){
                      unset($data[$k]);
                 }}
            ksort($encrypt_str);
$hmacSource = '';
foreach($encrypt_str as $key => $value){
       if (is_array($value)) {
            ksort($value);
            foreach ($value as $key2 => $value2) {
```

```
                if (is_object($value2)) {
                    $value2 = array_filter((array)$value2);
                    ksort($value2);
                    foreach ($value2 as $oKey => $oValue) {
                        $oValue .= '#';
                        $hmacSource .= trim($oValue);

                    }
                } else if(is_array($value2)){
                    ksort($value2);
                    foreach ($value2 as $key3 => $value3) {
                        if (is_object($value3)) {
                            $value3 = array_filter((array)$value3);
                            ksort($value3);
                            foreach ($value3 as $oKey => $oValue) {
                                $oValue .= '#';
                                $hmacSource .= trim($oValue);
                            }
                        } else{
                            $value3 .= '#';
                            $hmacSource .= trim($value3);
                        }
                    }
                } else{
                    $value2 .= '#';
                    $hmacSource .= trim($value2);
                }
            }
        } else {
            $value .= '#';
            $hmacSource .= trim($value);
        }
}
            return $hmacSource;
        }
```

## JAVA

见 sdk。

## C#

var a = JObject.Parse(order.Json); //将 a 中的 json 解码

```
string hm = string.Join("", GetValue.Getvalue1(a));
public static List<string> Getvalue1(object inp)
        {
            var res = new List<string>();
            var t = new SortedDictionary<string, JToken>();
            switch (inp.GetType().Name)
            {
                case "JValue":
                    res.Add(string.Format("{0}", (inp as JValue).Value));
                    break;
                case "JObject":
                    foreach (var x in (inp as JObject)) t.Add(x.Key, x.Value);
                    foreach (var x in t)
                    {
                            res.AddRange(Getvalue1(x.Value));
                            if (x.Value is JObject)
                        {}
                        else if (x.Value is JArray)
                        { }
                        else
                        {
                            if (String.IsNullOrEmpty(x.Value.ToString()))
                            {}
                            else
                            { res.Add("#");}
                        } }
                   break;
                case "JArray":
                    foreach (var x in (inp as JArray)) res.AddRange(Getvalue1(x));
                    break;
            }
            return res;
        }
```

## SHA1 签名

## PHP

```
 $sourceHmac=sha1($hmacSource, true);
```

## JAVA

```
public static byte[] encryptSHA(byte[] data)
      throws Exception
   {
      MessageDigest sha = MessageDigest.getInstance("SHA");
      sha.update(data);
      return sha.digest();
   }
```

## C#

```
 public static string EncryptToSHA1(string str)
        {
            SHA1CryptoServiceProvider sha1 = new SHA1CryptoServiceProvider();
            byte[] str1 = Encoding.UTF8.GetBytes(str);
            byte[] str2 = sha1.ComputeHash(str1);
            sha1.Clear();
            (sha1 as IDisposable).Dispose();
            return Convert.ToBase64String(str2);
        }
```

# AES 加密

## PHP

```
/**
      * AES 加密方法
      * @param string $str
      * @return string
      */

      $screct_key = $rands;
      $str = trim($str);
      $str = $this->addPKCS7Padding($str);
      $iv = mcrypt_create_iv(mcrypt_get_iv_size(MCRYPT_RIJNDAEL_128,MCRYPT_MODE_E
CB),MCRYPT_RAND);
      $encrypt_str =  mcrypt_encrypt(MCRYPT_RIJNDAEL_128, $screct_key, $str, MCRYPT_MO
DE_ECB, $iv);
      $date = base64_encode($encrypt_str);
   /**
    * 填充算法
    * @param string $source
```

```
 * @return string
 */
function addPKCS7Padding($source){
    $source = trim($source);
    $block = mcrypt_get_block_size('rijndael-128', 'ecb');
    $pad = $block - (strlen($source) % $block);
    if ($pad <= $block) {
        $char = chr($pad);
        $source .= str_repeat($char, $pad);
    }
    return $source;
}
```

## JAVA

```
/**
     * 对字符串进行加密，根据指定的密码
     *
     * @param content 需要加密的字符串
     * @return 返回加密完成之后的十六进制字符串
     * */
    public static String encrypt(String content, String key) {
        try {
            Cipher cipher = Cipher.getInstance("AES");// 创建密码器
            byte[] byteContent = content.getBytes(CHARSTER_ENCODING);
            cipher.init(Cipher.ENCRYPT_MODE, genKey(key));// 初始化
            byte[] result = cipher.doFinal(byteContent);
            return HexUtils.toHexStr(result); // 加密
        } catch (Exception e) {
            LOGGER.error(e.getMessage(), e);
        }
        return null;
    }

    /**
     * 对字符串进行解密，根据指定的密码
     *
     * @param content 需要加密的字符串
     * @return 返回解密之后的字符串
     * */
    public static String decrypt(String content, String key) {
```

```
        try {
            byte[] decryptFrom = HexUtils.toByte(content);
            Cipher cipher = Cipher.getInstance("AES");// 创建密码器
            cipher.init(Cipher.DECRYPT_MODE, genKey(key));// 初始化
            byte[] result = cipher.doFinal(decryptFrom);
            return new String(result); // 加密
        } catch (Exception e) {
            LOGGER.error(e.getMessage(), e);
        }
        return null;
    }

    /**
     * 根据密码获得一个 AES 类型的 key
     *
     * @param sourceKey 密码
     * @return SecretKeySpec 返回生成的 key
     * */
    private static SecretKeySpec genKey(String sourceKey) {
        byte[] enCodeFormat = {0};
        try {
            KeyGenerator kgen = KeyGenerator.getInstance("AES");
            SecureRandom secureRandom = SecureRandom.getInstance("SHA1PRNG");
            secureRandom.setSeed(sourceKey.getBytes());
            kgen.init(128, secureRandom);
            SecretKey secretKey = kgen.generateKey();
            enCodeFormat = secretKey.getEncoded();
        } catch (Exception e) {
            LOGGER.error(e.getMessage(), e);
        }
        return new SecretKeySpec(enCodeFormat, "AES");
    }


    /**
     * 加密
     *
     * @return
     */
    public static byte[] encrypt(byte[] data, byte[] key) {
        if(key.length!=16){
```

```
            throw new RuntimeException("Invalid AES key length (must be 16 bytes)");
        }
        try {
            SecretKeySpec secretKey = new SecretKeySpec(key, "AES");
            byte[] enCodeFormat = secretKey.getEncoded();
            SecretKeySpec seckey = new SecretKeySpec(enCodeFormat,"AES");
            Cipher cipher = Cipher.getInstance(CipherConfigure.AES_ALGORITHM);// 创建密码器
            cipher.init(Cipher.ENCRYPT_MODE, seckey);// 初始化
            byte[] result = cipher.doFinal(data);
            return result; // 加密
        } catch (Exception e){
            throw new RuntimeException("encrypt fail!", e);
        }
    }
```

## C#

```
public static string Encrypt(string plainText, string aesKey)
        {
            byte[] keyArray = UTF8Encoding.UTF8.GetBytes(aesKey);
            byte[] toEncryptArray = UTF8Encoding.UTF8.GetBytes(plainText);
            RijndaelManaged rDel = new RijndaelManaged();
            rDel.Key = keyArray;
            rDel.Mode = CipherMode.ECB;
            rDel.Padding = PaddingMode.PKCS7;
            ICryptoTransform cTransform = rDel.CreateEncryptor();
            byte[] resultArray = cTransform.TransformFinalBlock(toEncryptArray, 0, toEncryptArray.Length);
            return Convert.ToBase64String(resultArray, 0, resultArray.Length);
        }
```

# AES 解密

## PHP

```
/*
     * AES 解密
     *
     */
    $date = base64_decode($data['data']);
```

```
$screct_key = $decrypted;
$iv = mcrypt_create_iv(mcrypt_get_iv_size(MCRYPT_RIJNDAEL_128,MCRYPT_MODE_ECB),MCRYPT_RAND);
$encrypt_str =  mcrypt_decrypt(MCRYPT_RIJNDAEL_128, $screct_key, $date, MCRYPT_MODE_ECB, $iv);
$encrypt_str = preg_replace('/[\x00-\x1F]/','', $encrypt_str);
$encrypt_str = json_decode($encrypt_str,true);‘’
```

## JAVA

```
/**
     * 解密
     * @return
     */
    public static byte[] decrypt(byte[] data, byte[] key) {
        if(key.length!=16){
            throw new RuntimeException("Invalid AES key length (must be 16 bytes)");
        }
        try {
            SecretKeySpec secretKey = new SecretKeySpec(key, "AES");
            byte[] enCodeFormat = secretKey.getEncoded();
            SecretKeySpec seckey = new SecretKeySpec(enCodeFormat, "AES");
            Cipher cipher = Cipher.getInstance(CipherConfigure.AES_ALGORITHM);// 创建密码器
            cipher.init(Cipher.DECRYPT_MODE, seckey);// 初始化
            byte[] result = cipher.doFinal(data);
            return result; // 加密
        } catch (Exception e){
            throw new RuntimeException("decrypt fail!", e);
        }
    }
```

## C#

```
public static string Decrypt(string prestr, string key)
        {
            if (string.IsNullOrEmpty(prestr)) return null;
            Byte[] toEncryptArray = Convert.FromBase64String(prestr);
            RijndaelManaged rm = new RijndaelManaged
            {
                Key = Encoding.UTF8.GetBytes(key),
                Mode = CipherMode.ECB,
```

```
                Padding = PaddingMode.PKCS7
            };
            ICryptoTransform cTransform = rm.CreateDecryptor();
            Byte[] resultArray = cTransform.TransformFinalBlock(toEncryptArray, 0, toEncryptArray.Length);
            return Encoding.UTF8.GetString(resultArray);
        }
```

## CFCA 公钥加密

### PHP

```
 function rsaPublicEncode($public_key,$rands){

        $encryptKey=file_get_contents($public_key);
        $pem = chunk_split(base64_encode($encryptKey),64,"\n");//转换为 pem 格式的公钥
        $public_key = "-----BEGIN CERTIFICATE-----\n".$pem."-----END CERTIFICATE-----\n";
        $pu_key =  openssl_pkey_get_public($public_key);
        openssl_public_encrypt($rands,$encrypted,$pu_key);
        $encryptKey=base64_encode($encrypted);
        return $encryptKey;
    }
```

### JAVA

```
    public static byte[] encryptByPublicKeyF(byte[] data, String publicKey)
      throws Exception
  {
      byte[] keyBytes = decryptBASE64(publicKey);
      X509EncodedKeySpec x509KeySpec = new X509EncodedKeySpec(keyBytes);
      KeyFactory keyFactory = KeyFactory.getInstance("RSA");
      Key publicK = keyFactory.generatePublic(x509KeySpec);

      Cipher cipher = Cipher.getInstance(keyFactory.getAlgorithm());
      cipher.init(1, publicK);
      int inputLen = data.length;
      ByteArrayOutputStream out = new ByteArrayOutputStream();
      int offSet = 0;

      int i = 0;
```

```
    while (inputLen - offSet > 0)
    {
       byte[] cache;
       if (inputLen - offSet > MAX_ENCRYPT_BLOCK)
          cache = cipher.doFinal(data, offSet, MAX_ENCRYPT_BLOCK);
       else {
          cache = cipher.doFinal(data, offSet, inputLen - offSet);
       }
       out.write(cache, 0, cache.length);
       ++i;
       offSet = i * MAX_ENCRYPT_BLOCK;
    }
    byte[] encryptedData = out.toByteArray();
    out.close();
    return encryptedData;
  }
```

## C#

```
    public static string CFCAencryption(string publicKeyPath, string data){
                X509Certificate2 pubcrt = new X509Certificate2(publicKeyPath);
                string keyPublic2 = pubcrt.PublicKey.Key.ToXmlString(false);
                string rsadata = RSAEncrypt(keyPublic2, data);
                return rsadata;
                }
```

# CFCA 私钥解密

## PHP

```
 function rsaPrivateDecode($data,$private_key,$password){
            $prikey=file_get_contents($private_key);
            $encryptKey =$data['encryptKey'];
            $results=array();
            openssl_pkcs12_read($prikey,$results,$password);
            $private_key=$results['pkey'];
            $pi_key =    openssl_pkey_get_public($private_key);
            openssl_private_decrypt(base64_decode($encryptKey),$decrypted,$private_key);
            return $decrypted;
    }
```

## JAVA

```
    public static byte[] decryptByPrivateKeyF(byte[] encryptedData, String privateKey)
    throws Exception
  {
    byte[] keyBytes = decryptBASE64(privateKey);
    PKCS8EncodedKeySpec pkcs8KeySpec = new PKCS8EncodedKeySpec(keyBytes);
    KeyFactory keyFactory = KeyFactory.getInstance("RSA");
    Key privateK = keyFactory.generatePrivate(pkcs8KeySpec);
    Cipher cipher = Cipher.getInstance(keyFactory.getAlgorithm());
    cipher.init(2, privateK);
    int inputLen = encryptedData.length;
    ByteArrayOutputStream out = new ByteArrayOutputStream();
    int offSet = 0;

    int i = 0;

    while (inputLen - offSet > 0)
    {
      byte[] cache;
      if (inputLen - offSet > MAX_DECRYPT_BLOCK)
        cache = cipher.doFinal(encryptedData, offSet, MAX_DECRYPT_BLOCK);
      else {
        cache = cipher.doFinal(encryptedData, offSet, inputLen - offSet);
      }
      out.write(cache, 0, cache.length);
      ++i;
      offSet = i * MAX_DECRYPT_BLOCK;
    }
    byte[] decryptedData = out.toByteArray();
    out.close();
    return decryptedData;
  }
```

## C#

```
    public static string CFCADencryption(string prviateKeyPath, string data, string pfxPassword)
        {
            X509Certificate2 pubcrt = new X509Certificate2(prviateKeyPath, pfxPassword,
X509KeyStorageFlags.Exportable | X509KeyStorageFlags.PersistKeySet);
            string keyprivate = pubcrt.PrivateKey.ToXmlString(true);
            string rsadata = RSADecrypt(keyprivate, data);
```

```
        return rsadata;
    }
```

## CFCA 私钥签名

### PHP

```
function rsaPrivateSign($data,$path,$password){
    $pubKey = file_get_contents($path);
    $results=array();
    openssl_pkcs12_read($pubKey,$results,$password);
    $private_key=$results['pkey'];
    $pi_key =  openssl_pkey_get_private($private_key);//这个函数可用来判断私钥是否是可用的，可用返回资源 id Resource id
    openssl_sign($data, $signature,$private_key,"md5");
    $signature=base64_encode($signature);
    return $signature;
}
```

### JAVA

```
public static String sign(byte[] data, String privateKey)
    throws Exception
{
    byte[] keyBytes = decryptBASE64(privateKey);

    PKCS8EncodedKeySpec pkcs8EncodedKeySpec = new PKCS8EncodedKeySpec(keyBytes);

    KeyFactory keyFactory = KeyFactory.getInstance("RSA");

    PrivateKey privateKey2 = keyFactory.generatePrivate(pkcs8EncodedKeySpec);

    Signature signature = Signature.getInstance("MD5withRSA");
    signature.initSign(privateKey2);
    signature.update(data);

    return encryptBASE64(signature.sign());
}
```

### C#

```
    public static string privateSign(string privateKeyPath,string data ,string pfxPassword)
```

```
        {
            X509Certificate2 objx5092;
            if (string.IsNullOrWhiteSpace(pfxPassword))
            {
                objx5092 = new X509Certificate2(privateKeyPath);
            }
            else
            {
                objx5092 = new X509Certificate2(privateKeyPath, pfxPassword);
            }
            RSACryptoServiceProvider rsa = objx5092.PrivateKey as
RSACryptoServiceProvider;
            byte[] dataencod = Convert.FromBase64String(data);
            byte[] reslut = rsa.SignData(dataencod, "MD5");//为证书采用 MD5withRSA 签名
            return Convert.ToBase64String(reslut);
    }
```

## CFCA 公钥验签

### PHP

```
 function rsaPubilcSign($data,$path,$hmac){
            $public_key=file_get_contents($path);
            $pem1 = chunk_split(base64_encode($public_key),64,"\n");
            $pem1 = "-----BEGIN CERTIFICATE-----\n".$pem1."-----END CERTIFICATE-----\n";
            $pi_key =  openssl_pkey_get_public($pem1);
            $result=openssl_verify($data,base64_decode($hmac),$pem1,OPENSSL_ALGO_MD5);
            return $result;

      }
```

### JAVA

```
     public static boolean verify(byte[] data, String sign, X509Certificate cert)
    throws Exception
  {
    PublicKey publicKey = cert.getPublicKey();

    Signature signature = Signature.getInstance(cert.getSigAlgName());
    signature.initVerify(publicKey);
    signature.update(data);
```

```
    return signature.verify(decryptBASE64(sign));
}
```

C#

```
public static bool VerifySign(string data, string publicKeyPath, string sign)
        {
            byte[] messagebytes = Convert.FromBase64String(data);
            byte[] messagesign = Convert.FromBase64String(sign);
            X509Certificate2 x509 = new X509Certificate2(publicKeyPath);

            RSACryptoServiceProvider oRSA = new RSACryptoServiceProvider();
            oRSA.FromXmlString(x509.PublicKey.Key.ToXmlString(false));

            bool bVerify = oRSA.VerifyData(messagebytes, "MD5", messagesign);

            return bVerify;
        }
```

# 9.附录

| | |
|---|---|
| 首信易支付平台： | 首信易支付 |
| 首信易支付英文支付通道： | Easy-pay online platform |

注：1、890000593 为示例商户编号，商户不可用于测试
2、商户用于测试的商户编号及测试 CFCA 证书请与首信易支付工作人员联系获取
3、商户在使用正式商户编号时，请提前与首信易支付工作人员联系申请 CFCA 证书
4、商户测试时，请务必测试 CFCA 加密环节