[文档版本更新记录 2](#_Toc22347)

[接口规则 4](#_Toc4020)

[1. 协议规则 4](#_Toc29159)

[2. 请求响应 4](#_Toc2100)

[3.TLV格式说明 4](#_Toc12555)

[文档说明 5](#_Toc19451)

[具体协议 6](#_Toc28862)

[1. 获取密钥(/v2/device/secret/get) 6](#_Toc16369)

[2. 公交刷卡机乘车消费(/v2/device/bus/consume) 8](#_Toc32600)

[数据域定义表 9](#_Toc18898)

[全局错误码 10](#_Toc24901)

[服务端提供的资料 10](#_Toc17272)

[附录1. API请求签名举例 12](#_Toc15275)

[附录2. 二维码生成规则及安全 14](#_Toc27993)

[附录3. SM2的签名说明 18](#_Toc25966)

# 文档版本更新记录

### 1.0.0

基础版本

### 1.0.1

增加附录2 二维码生成规则及安全

### 1.0.2

1. 二维码生成规则中 卡号字段的长度由10字节调整为5字节；
2. 增加二维码生成和解析，加签和验签的举例说明；
3. 添加附录3.SM2的签名说明

### 1.0.3

1.为了更好的透传数据, 修改Http的请求和响应报文结构

2. 增加TLV格式说明

3. 取消接口的请求签名认证

### 1.0.4

1. 刷卡乘车消息数据域添加，商户编号，司机编号，分公司号编号 字段

### 1.0.5

1. 调整接口的请求路径
2. 增加测试配置

### 1.0.6

1. 刷卡乘车消费中的“交易时间”调整为8直接的Long，毫秒值；

### 1.0.7

1. 增加两个接口的请求示例；

### 1.0.8

1. 补充乘车数据上传之后的返回状态为

0：成功（正常消费）

1：重复交易记录（返回之前成功扣款的流水号）

2: 二维码解析错误

3： 其他错误；

4: 二维码已过期（正常消费）

5： 二维码签名错误

正常消费表示会正常扣款，其他不会扣款;

### 1.0.9

1. 接口 公交刷卡机乘车消费(/v2/device/bus/consume) 增加请求字段 公司编号(company\_id)

# 接口规则

## 1. 协议规则

|  |  |
| --- | --- |
| 传输方式 | 采用HTTPS传输 |
| 提交方式 | 采用POST方法提交 |
| 数据格式 | 提交和返回数据都为JSON格式 |
| 字符编码 | 统一采用UTF-8字符编码 |
| 判断逻辑 | 判断交易状态 |

## 2. 请求响应

rspcd与rspmsg做为响应的非填项，不在交易定义中再做说明

请求响应

|  |  |  |
| --- | --- | --- |
| 字段名 | 描述 | 是否必填 |
| rspcd | 交易状态码,000000 全零表示为成功，其他表示为失败 | 是 |
| rspmsg | 交易状态描述,非全零时表示为错误描述 | 是 |
| body | 应交易定义有返回数据时，数据都在body结点下 | 否 |

## 3.TLV格式说明

TLV是 Tag、Length和Value的缩写，一个基本的数据元素就包括上述的三个域，Tag唯一标识数据元，Length是value域的长度，Value就是本身的数据。

我们在请求和响应的data节点中，以tlv数据格式组织数据。

Tag,Length我们采用固定的长度. Tag 2个字节，length 2个字节

格式

9F060007A0000000001010

其中9F06 是Tag值,0007是长度, A0000000001010就是9F06的值

# 文档说明

### 文档标题

公交卡接口服务

### 文档版本

1.0

### 访问基路径

HTTPS://ip:port/posservice-api

### 一个完整的请求路径(获取密钥为例)

HTTPS://ip:port/posservice-api/v2/device/secret/get

# 具体协议

## 获取密钥(/v2/device/secret/get)

请求

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **字段** | **字段类型** | **说明** | **是否必输** |
| 数据域 | data | string | Tlv转hex字符 | 是 |
| 校验域 | md5 | string | 使用md5方法对data数据域内容进行摘要，抽取md5值的中间8个字节做为其校验域内容,将字节转hex字串 | 是 |

响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **字段** | **字段类型** | **说明** | **是否必输** |
| 数据域 | data | string | Tlv转hex字符 | 是 |
| 校验域 | md5 | string | 使用md5方法对data数据域的原始内容进行摘要，抽取md5值的中间8个字节做为其校验域内容,将字节转hex字串 | 是 |

数据域内容

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序列 | 字段名 | 字段编码 | 长度 | 描述 | 格式 | 必须 |
| 1 | 商户编号 | 9F01 | N | Utf-8编码 | ANS | M |
| 2 | 设备编号 | 9F02 | N | Utf-8编码 | ANS | M |
| 3 | 公交班次号 | 9F03 | N | Utf-8编码 | ANS | M |
| 4 | 线路编号 | 9F04 | N | Utf-8 编码 | ANS | M |

响应数据域

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序列 | 字段名 | 字段编码 | 长度 | 描述 | 格式 | 必须 |
| 1 | 服务器时间戳 | 9F15 | 8 | LONG | N | M |
| 2 | 公钥列表 | 9F16 | N | 服务端下载多个日期的公钥. |  | M |

公钥列表数据域

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 序列 | 字段名 | 长度 | 描述 | 格式 | 必须 |
| 1 | 公钥个数 | 1 | 公钥个数不超过256个 | N | M |
| 2 | 公钥对应日期 | N | Utf-8编码 | ANS | M |
| 3 | 公钥长度 | 2 | 公钥的长度. Sm2公钥长度64 | N | M |
| 4 | 公钥内容 | N | 二进度数据，  2~4 字段表示一个公钥内容 |  | M |

请求报文示例:

商户编号： 1234

设备编号： 5567

公交班次号： 57912

线路编号: 101

Data 数据域内容为 9f0400033130319f03000535373931329f020004353536379f01000431323334

Md5 校验域内容为：d81d739863371aa9

注: md5的摘要对像是 tlv输出的原始字节数据，而不是转hex的字串byte

响应报表

Data数据域内容为

9f16004102000832303138303432300014313233343536373839303132333435363738393000083230313830343231001431323334353637383930313233343536373839309f15000800000162e1c74973

Md5校验域内容为: 7b1e6c3c8547f870

错误码:

|  |  |
| --- | --- |
| **错误码** | **错误描述** |

请求示例

{"data":"9F01000833373135303030319F02000C3030303030323131383434369F03000830303037373338369F0400083030303030333036","md5":"AA6D3971BE73D849"}

响应

{"rspcd":"000000","rspmsg":"succeed","body":{"data":"9f150008000001634d2613df9f1600950232303138303531310040988387b3ae02af013487998a018eca7964a64bd8d54d2e6688f16a6e9ce9a0a881611178454cad1157dbb808eea2c47a27ea59d1a6f16bf69519c486a9971753323031383035313200406ddce35144fc39ed2f864ce06b4df1f2d50abfbdacafb054e2b6f4ecbea3dbfcd32aa015fd26fc046ca219ede424f3b3d11560304f85a418d5d5419e36a80d5e","md5":"593d7dcd74ae7bd7"}}

## 公交刷卡机乘车消费(/v2/device/bus/consume)

请求

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **字段** | **字段类型** | **说明** | **是否必输** |
| 数据域 | data | string | Tlv转hex字符 | 是 |
| 校验域 | md5 | string | 使用md5方法对data数据域内容进行摘要，抽取md5值的中间8个字节做为其校验域内容,将字节转hex字串 | 是 |
| 公司编号 | company\_id | string |  | 否 |

响应

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段名称** | **字段** | **字段类型** | **说明** | **是否必输** |
| 数据域 | data | string | Tlv转hex字符 | 是 |
| 校验域 | md5 | string | 使用md5方法对data数据域的原始内容进行摘要，抽取md5值的中间8个字节做为其校验域内容,将字节转hex字串 | 是 |

请求数据域

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 字段编码 | 长度 | 描述 | 格式 | 必须 |  |
| 1 | 商户编号 | 9F01 | N | Utf-8编码 | ANS | M |  |
| 2 | 设备编号 | 9F02 | N | Utf-8编码 | ANS | M |  |
| 3 | 公交班次号 | 9F03 | N | Utf-8编码 | ANS | M |  |
| 4 | 线路编号 | 9F04 | N | Utf-8编码 | ANS | M |  |
| 5 | 司机编号 | 9F05 | N | Utf-8编码 | ANS | M |  |
| 6 | 分公司编号 | 9F06 | N | Utf-8编码 | ANS | M |  |
| 7 | 金额 | 9F11 | 4 | 单位:分 | N | M |  |
| 8 | 交易金额 | 9F12 | 4 | 单位:分 | N | M |  |
| 9 | 交易时间 | 9F13 | 8 | Long,毫秒值 | N | M |  |
| 10 | 二维码内容 | 9F14 | N | Utf-8编码 | ANS | M |  |

响应数据域

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 字段编码 | 长度 | 描述 | 格式 | 必须 |
| 1 | 交易状态 | 9F17 | 1 | Byte 类型  0：成功（正常消费）  1：重复交易记录  2: 二维码解析错误  3： 其他错误；  4: 二维码已过期（正常消费）  5： 二维码签名错误 | N | M |
| 2 | 交易流水号 | 9F18 | N | Utf-8编码 | ANS | M |

错误码

|  |  |
| --- | --- |
| **错误码** | **错误描述** |

# 数据域定义表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 序号 | 字段名 | 字段编码 | 长度 | 描述 | 格式 | 必须 |  |
| 1 | 商户编号 | 9F01 | N | Utf-8编码 | ANS | M |  |
| 2 | 设备编号 | 9F02 | N | Utf-8编码 | ANS | M |  |
| 3 | 公交班次号 | 9F03 | N | Utf-8编码 | ANS | M |  |
| 4 | 线路编号 | 9F04 | N | Utf-8编码 | ANS | M |  |
| 5 | 司机编号 | 9F05 | N | Utf-8编码 | ANS | M |  |
| 6 | 分公司编号 | 9F06 | N | Utf-8编码 | ANS | M |  |
| 7 | 金额 | 9F11 | 4 | 单位:分 | N | M |  |
| 8 | 交易金额 | 9F12 | 4 | 单位:分 | N | M |  |
| 9 | 交易时间 | 9F13 | N | 格式: yyyyMMddHHmmss | ANS | M |  |
| 10 | 二维码内容 | 9F14 | N | Utf-8编码 | ANS | M |  |
| 11 | 服务器时间戳 | 9F15 | 8 | LONG | N | M |  |
| 12 | 公钥列表 | 9F16 | N |  | N | M |  |
| 13 | 交易状态 | 9F17 | 1 | Byte | N | M |  |
| 14 | 交易流水号 | 9F18 | N |  | ANS | M |  |

请求示例

{"data":"9F01000833373135303030319F02000C3030303030323131383434369F03000830303037373338369F04000830303030303330369F0500063130303030309F06000230319F110004000000649F120004000000649F130008000001634D263EA79F140053425931370700003C5AF507A95428374351000064DAC88C372A8D72E56C67966ED7675C0C313FF10A1BDACDA143FCF7E472D1BD5DC62CB2084C8BB07F709B8AC01EB27D86029E33E5128077157B0832013C33A1","md5":"252EBEF07F22FC1E"}

请求响应

{"rspcd":"000000","rspmsg":"succeed","body":{"data":"9f180013323031383035313131313032313739333936339f17000100","md5":"d747f3cc5c26cb35"}}

# 全局错误码

全局错误码表示在每个接口都会出现，不在每个接口中声明

|  |  |
| --- | --- |
| **错误码** | **错误描述** |

|  |  |
| --- | --- |
| SERVICE10000 | 服务忙 |
|  |  |
|  |  |

# 服务端提供的资料

1. API请求接入的资料信息

测试：

商户编号(OAUTH\_APPID)： appid\_test

签名密钥：  uLOYLyw0bYliydKa9X6SQ56U

正式：

商户编号(OAUTH\_APPID)：

签名密钥：

1. 请求地址： <http://118.190.5.218:7080/posservice-api/接口?oauth_appid=appid_test>
2. 测试商户编号： mch1001

# 附录1. API请求签名举例

(以“接口1 获取卡信息”为例)

测试 appid： appid\_test

测试appsecret：  uLOYLyw0bYliydKa9X6SQ56U

请求参数：

{

　　"hard\_code":"hcode",

　　"merchant\_id":"metchant\_test",

"hard\_id":"hid",

"test":""

}

第一步，设所有发送或者接收到的数据为集合M，将集合M内非空参数值的参数按照参数名ASCII码从小到大排序（字典序），使用URL键值对的格式（即key1=value1&key2=value2…）拼接成字符串stringA。

待签名数据结合M：

{

　　"hard\_code":"hcode",

　　"hard\_id":"hid",

　　"merchant\_id":"metchant\_test",

　　"oauth\_appid":"appid\_test",

　　"oauth\_method":"MD5",

　　"oauth\_nonce":"6be645fe-f72a-4ddb-b8e8-c4a8819e0fec",

　　"oauth\_timestamp":"1522395737",

"test":""

}

将集合M内非空参数值的参数按照参数名ASCII码从小到大排序（字典序）得到：

{

"hard\_code":"hcode",

　　"hard\_id":"hid",

　　"merchant\_id":"metchant\_test",

　　"oauth\_appid":"appid\_test",

　　"oauth\_method":"MD5",

　　"oauth\_nonce":"6be645fe-f72a-4ddb-b8e8-c4a8819e0fec",

　　"oauth\_timestamp":"1522395737",

}

使用URL键值对的格式（即key1=value1&key2=value2…）拼接成字符串stringA：

hard\_code=hcode&hard\_id=hid&merchant\_id=metchant\_test&oauth\_appid=appid\_test&oauth\_method=MD5&oauth\_nonce=6be645fe-f72a-4ddb-b8e8-c4a8819e0fec&oauth\_timestamp=1522395737

第二步，在stringA最后拼接上secret得到stringSignTemp字符串，并对stringSignTemp进行MD5运算，再将得到的字符串所有字符转换为大写，得到sign值signValue。

在stringA最后拼接上secret得到stringSignTemp字符串：

hard\_code=hcode&hard\_id=hid&merchant\_id=metchant\_test&oauth\_appid=appid\_test&oauth\_method=MD5&oauth\_nonce=6be645fe-f72a-4ddb-b8e8-c4a8819e0fec&oauth\_timestamp=1522395737uLOYLyw0bYliydKa9X6SQ56U

并对stringSignTemp进行MD5运算：

md5(stringSignTemp):

fe3f991695c59193fe3e5bea8507c1b0

再将得到的字符串所有字符转换为大写，得到sign值signValue:

FE3F991695C59193FE3E5BEA8507C1B0

即，请求参数的SIGN=FE3F991695C59193FE3E5BEA8507C1B0

# 附录2. 二维码生成规则及安全

1. 二维码的组成

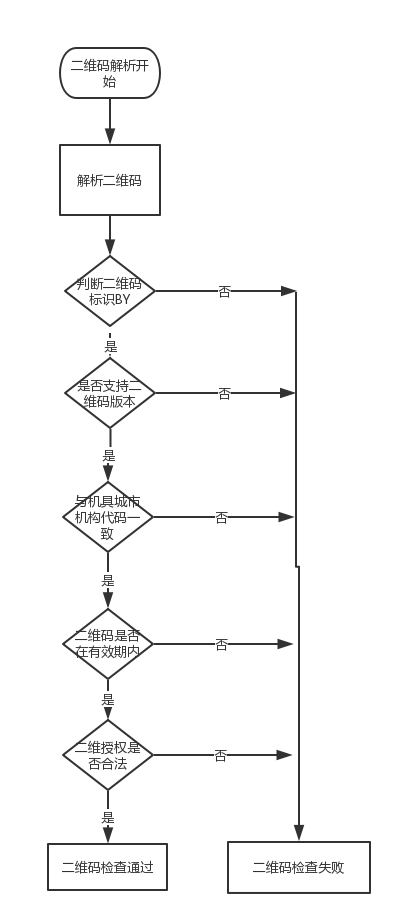
公交二维码由二维码头、联机授权域组成，公交二维码数据结构见表A。

表A-1二维码数据格式

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据域 | 序号 | 字段名 | 长度 | 描述 | 格式 | 必须 |
| 二维码头 | 1 | 二维码标识 | 2 | 二维码标识 取值BY | ANS | M |
| 2 | 二维码版本 | 1 | 二维码结构版本, 当前版本=1 | B | M |
| 授权机构授权域 | 3 | 城市机构代码 | 3 | 潍坊城市机构代码370700,实际转化为  0x37 0x07 0x00 | BCD | M |
| 4 | 二维码有效时长 | 2 | 二维码从创建开始到失效间的时间，以秒为单位 | B | M |
| 5 | 二维码生成时间 | 4 | 二维码生码的时间，单位秒,  Unix时间戳，对应1970的时间差 | B | M |
| 6 | 支付卡号 | 5 | 虚拟电子卡号 | BCD | M |
| 7 | 卡类型 | 1 | 卡类型  0:普通卡 | B | M |
| 8 | 自定义数据长度 | 1 | 自定义数据长度不超过256 | N | M |
| 9 | 自定长数据 | N |  | B | 0 |
| 签名域 | 10 | 授权签名 | 64 | 授权机构私钥对3~9项数据进行签名 | B | M |
|  |  |  | 83+ |  |  |  |

受理终端读取二维码信息进行验证，流程图A-2

图A-2二维码验码流程



二维码验证流程应符合下列要求：

a)   判断码源基本属性：检查二维码标识、版本信息；

b)   验证码源正确性：检查二维码是否在有效期内，验证机构授权签名是否正确，保证内容不被破坏；

二维码检查通过后，受理终端执行本地交易流程。交易成功后受理终端将生成交易记录，提示交易成功，并为用户提供乘车服务。

1.1　二维码的合法性

1.1.1　安全性保证

二维码的安全性保证由发卡机构私钥签名保证

1.1.2　实时性保证

二维码的实时性由‘二维码有效时长’及‘二维码生成时间’保证，二维码发码平台与受理终端均采用世界标准时间，应按以下方式保证时间同步：

a)   二维码发码平台应具备定期与世界时间服务器时间同步的机制，保证二维码的生码时间与世界标准时间保持一致；

b)   受理终端应具备定期或必要时与收单机构后台系统完成时间同步的机制，保证受理终端在验码时效性。

1. 二维码生成和解析
   1. 概述

二维码是卡的基本信息通过sm2加签，然后通过二进制转16进制字符串进行传输；

* 1. 举例的SM2的公私钥

公钥：04DC6E0D77BDC10D0BB88BD7F41D961E0ADCD61F0DAFDD1A41D61E45E8056D347BE5FDEA29CFE8494346D27A14E2B4C08FB465961E6993738F77FDCB324A7B38BD

私钥：42F2DA0C2E6CE34013D6D4051D46F86298768817F026D7471242C13416B1949C

* 1. 二维码生成

2.3.1 二维码的基本信息：

》二维码标识：BY（2字节）

》版本号：1（1字节）

》城市编号：370700（3字节）

》二维码有效时长：60（2字节）

》二维码生成时间：1523591318（4字节）

》支付卡号：3897882084（5字节）

》卡类型：0（1字节）

》自定义数据长度：0（1字节）

》自定义数据：这里没有自定义数据，是个空字符串（自定义数据长度字节）

》签名：yyy（64字节）

2.3.2 二维码生成

》将城市编号(370700)，二维码有效时长(60)，二维码生成时间()，支付卡号(1523583510)，卡类型(1523583510，需要bcd编码)，自定义数据长度(0)，自定义数据(“”)，生成待签名byte数组；

》使用SM2的私钥对byte数组进行签名，得到签名sign（参考：[附录3. SM2的签名说明](#_附录3. SM2的签名说明)）

例子数据得到的签名r的16进制字符串：12D6C20BC373322073F61636996D58995D1EF192F83963C072C2954459FA4709；

例子数据得到的签名s的16进制字符串：256EB4BF8DFB6DE24551BC6018D78CA08FBB2A5F04AC5D7BD5017050FAC12B4A

r在前，s在后得到的签名：12D6C20BC373322073F61636996D58995D1EF192F83963C072C2954459FA4709256EB4BF8DFB6DE24551BC6018D78CA08FBB2A5F04AC5D7BD5017050FAC12B4A

》将二维码标识(BY)，版本号(1)，城市编号(370700)，二维码有效时长(60)，二维码生成时间()，支付卡号(1523583510)，卡类型(1523583510，需要bcd编码)，自定义数据长度(0)，自定义数据(“”)，签名(sign)，生成二维码的byte数组；

》将二维码数组转成16进制字符串得到二维码展示字符串；

例子数据生成的二维码:425931370700003C5AD028963897882084000012D6C20BC373322073F61636996D58995D1EF192F83963C072C2954459FA4709256EB4BF8DFB6DE24551BC6018D78CA08FBB2A5F04AC5D7BD5017050FAC12B4A

2.3.3 二维码的解析

》将16进制二维码展示字符串转成byte数组；

》根据二维码的生成规则，解析出里面的二维码的基本字段；

》根据二维码的生成规则进行签名校验；

# 附录3. SM2的签名说明

SM2的签名结果是由一对签名数据组成，分别是r和s，长度都是32字节，二维码签名的时候是把r在前，s在后组成一个64字节的byte数组作为签名，所以在验签的时候，解析出二维码中的签名，需要把前32字节作为r，后32字节作为s，组成一对SM2的签名；