**内 部**

**铁路公安 机构用户同步**

**接口规范**

**二〇二二年五月**

目录

[1 接口规范 3](#_Toc90884077)

[1.1 技术要求 3](#_Toc90884078)

[1.1.1 数据传输加密机制 3](#_Toc90884079)

[1.1.2 支持扩展性 3](#_Toc90884080)

[1.1.3 容错机制 3](#_Toc90884081)

[1.2 接口规范 3](#_Toc90884082)

[1.2.1 接口设计要求 3](#_Toc90884083)

[1.2.2 接口工作模式 3](#_Toc90884084)

[1.2.3 接口错误处理 4](#_Toc90884085)

[2 接口定义 5](#_Toc90884086)

[2.1 租户开通 5](#_Toc90884087)

[2.1.1 字段详情 5](#_Toc90884088)

[2.1.2 接口详情 6](#_Toc90884089)

[2.2 租户变更 7](#_Toc90884090)

[2.2.1 字段详情 7](#_Toc90884091)

[2.2.2 接口详情 8](#_Toc90884092)

[2.3 租户注销 9](#_Toc90884093)

[2.3.1 字段详情 9](#_Toc90884094)

[2.3.2 接口详情 9](#_Toc90884095)

[2.4 子用户新增 10](#_Toc90884096)

[2.4.1 字段详情 10](#_Toc90884097)

[2.4.2 接口详情 11](#_Toc90884098)

[2.5 子用户变更 11](#_Toc90884099)

[2.5.1 字段详情 11](#_Toc90884100)

[2.5.2 接口详情 12](#_Toc90884101)

[2.6 子用户取消订购 13](#_Toc90884102)

[2.6.1 字段详情 13](#_Toc90884103)

[2.6.2 接口详情 14](#_Toc90884104)

[2.7 子用户状态变化 14](#_Toc90884105)

[2.7.1 字段详情 14](#_Toc90884106)

[2.7.2 接口详情 16](#_Toc90884107)

# 接口规范

## 技术要求

### 数据传输加密机制

由于系统数据的重要性，为了加强传输过程中数据的安全，采用HTTPS协议进行数据传输。

### 支持扩展性

服务采用RESTful的接口实现方式；数据采用JSON格式表示，并且符合相应的JSON Schema；开发语言无绑定要求。

### 容错机制

为保证数据的一致性、可靠性，数据上报服务应支持容错重传机制。

## 接口规范

### 接口设计要求

（1）接口协议均基于HTTP/HTTPS协议，采用RESTful的接口实现方式。

（2）接口调用要求具备认证机制。

（3）数据格式要求统一使用JSON格式表示，并且符合相应的JSON Schema。

（4）报文内容处理须满足事务处理一致性原则。根据JSON名称和路径进行精确定位，不应根据字段的顺序来获取字段值。报文统一采用UTF-8进行编码。单次交互报文大小限制在10MB以内，若大于10MB数据上报方需要进行拆包处理。

（5）错误处理要求保持事物处理一致性，使用统一标准的错误代码。

### 接口调用顺序

同步机构和用户前，必须先进行网关登录和用户登录，

网关登录用于确保调用方可信

用户登录用于后续机构和用户同步的数据筛选

1. 网关登录
2. 用户登录
3. 同步机构/同步用户

### 接口错误处理

接口调用失败时，需要根据HTTP状态码进行响应处理。

|  |  |  |
| --- | --- | --- |
| 错误类型 | 错误信息 | 错误处理方式 |
| 网络连接超时 | HTTP状态码: 408 | 重新传输。当出现网络问题时，调用者应延迟一段时间，如10分钟，再发起重新传输。 |
| 网络限流 | HTTP状态码: 509 | 重新传输。当出现网络问题时，调用者应延迟一段时间，如10分钟，再发起重新传输。 |

# 接口定义

## 网关登录

### 字段详情

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 名称 | 英文名 | 类型 | 最小 | 最大 | 必选 | 备注 |
| 应用标识 | appId | String | 1 | 64 | 是 | 请求方标识，由基础人员信息库提供 |
| 随机数字串 | nonce | String | 1 | 64 | 是 | 请求方产生的随机数字符串 |
| 时间戳 | timestamp | String | 1 | 64 | 是 | 时间戳, epoch毫秒数 |
| 签名 | sign | String | 1 | 500 | 是 | 由appId, nonce, timestamp和共享秘钥生成的签名，共享密钥由基础人员信息库提供。签名算法见下 |
| import javax.crypto.Mac;  import javax.crypto.SecretKey;  import javax.crypto.spec.SecretKeySpec;  import java.nio.charset.StandardCharsets;  import java.security.InvalidKeyException;  import java.security.NoSuchAlgorithmException;  import java.util.Base64;  public class HmacAuthUtil {  /\*\*  \* @param appId 请求方标识  \* @param appSecret 用户共享秘钥  \* @param nonce 随机数  \* @param timestamp 时间戳  \* @return 签名值base64字符串  \*/  public static String sign(String appId, String appSecret, String nonce, String timestamp) {  SecretKey key = new SecretKeySpec(appSecret.getBytes(StandardCharsets.UTF\_8), "HMAC");  try {  Mac mac = Mac.getInstance("HmacSHA256");  mac.init(key);  String text = appId + nonce + timestamp;  return Base64.getEncoder().encodeToString(mac.doFinal(text.getBytes(StandardCharsets.UTF\_8)));  } catch (NoSuchAlgorithmException | InvalidKeyException e) {  throw new IllegalStateException(e);  }  }  } | | | | | | |  |  |  |  |  |  |

### 返回字段详情

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 数据项名称 | 数据项英文名 | 类型 | 最小长度 | 最大长度 | 必选 | 说明 |
| 结果代码 | code | Integer |  |  | 是 | 0: 成功 |
| 结果描述 | message | String | 1 | 200 | 是 | 对请求结果的英文描述 |

### 接口详情

|  |  |
| --- | --- |
| URL | /cb\_sync /tenant/register |
| 方法 | POST |
| 参数范例 | {  "tenantId": "12345",  "tenantAccount": "test\_admin",  "tenantName": "测试租户",  "industry": "其他行业",  "contactMail": "abc@123.com",  "contactName": "张三",  "contactMobilePhone": "11122334455",  "priceInfo": "默认资费",  "licenseDevNum": 200,  "licenseEndTime": ”2021-03-05”,  "actionTime":”2021-03-05”,  "cbAcceptTime":”2021-03-05”,  "cbOperatorAccount": "test\_account",  "cbBussNum": "cb123456",  …  } |
| 正常返回值 | HTTP 200  {     "code": 0,     "message": "ok" } |
| 业务错误返回值 | 待定义 |
| 备注 |  |