# 自然人风险评分查询接口

**接口地址**:`https://tx.hz-hanghui.com:8088/hanghui-server-platform/api/v1/fx/query`

**请求方式**:`POST`

**请求数据类型**:`application/json`

**响应数据类型**:`*/*`

**接口描述**:`请求和响应报文参数均需要使用国密4进行加解密，涉及的appId、appKey、appSecret、privateKey管理员提供`

**请求示例**:

```
//国密m4加密后密文 也就是三方接口接收到的数据
{
    "appId": "",
    "bizContent":
"NI4ry+D01kDXrNf5OcmAzMGaB0td9kXfa321xmIS3qw5homFdZT4xaJu8Vqid37NDivdqj1ADzd0+v+QDKNovFSBkm
OyOQt2ObJ4HW43i0dKKHAKlI3FtZA0F9URhlqXbckbpGMD1hev3XS/b9uxRw+QLOnTzhuzgoxpiOXNA9rmQs3V7prpF
9Wpaetts2pv",
    "reqTimestamp": 1692693665800,
    "sign": "签名方式见文末加密算法"
}

//国密m4加密前 也就是三方服务解密后的数据
{
    "appId": "",
    "bizContent": {
        "sn": "",
        "name": "",
        "idNumber": "",
        "phone": ""
    },
    "reqTimestamp": 1692693665800,
    "sign": "签名方式见文末加密算法"
}
```

**请求参数**:

| 参数名称 | 参数说明 | 请求类型 | 是否必须 | 数据类型 | schema |
| --- | --- | --- | --- | --- | --- |
| bizContent | 加密内容 | body | true | | |
| sn | 设备号 | | true | string | |
| name | 姓名 | | true | string | |
| idNumber | 身份证号 | | true | string | |
| phone | 手机号 | | true | string | |
| appId | 商户唯一标识 | body | true | string | |
| reqTimestamp | 时间戳 毫秒 | body | true | long | |
| sign | 签名md5(appKey+appSecret+timestamp) 小写32位 | body | true | string | |

**响应状态:**

| 状态码 | 说明 | schema |
| --- | --- | --- |
| 200 | OK | |
| 除200以外都是异常码 | | |

**响应参数:**

| 参数名称 | 参数说明 | 类型 | schema |
| --- | --- | --- | --- |
| code | 状态码 | integer(int32) | integer(int32) |
| data | 对象 | | |
| bizContent | 加密后的数据 | string | |
| result | true/false 是否命中风险 | boolean | |
| resultInfo | 结果信息 | | |
| personFxpfjxCode | 风险状态 200 除200以外都是异常 | integer(int32) | integer(int32) |
| personFxpfjx | 风险值："001010"<br>命中结果由 6 位0｜1 字符串组成<br>分别为<br>第 1 位-其他、<br>第 2 位-前科、<br>第 3 位-涉毒、<br>第 4 位-吸毒、<br>第 5 位-在逃、<br>第 6 位-涉案(包括在逃撤销)。<br>值为 1 表示命中，0 表示未 命中 | string | |
| reqTimestamp | 时间戳毫秒 1666881956513 | long | |
| sign | 签名md5(appKey+appSecret+timestamp) 小写32位 | string | |
| msg | 返回消息 | string | |

**响应示例:**

```
//国密m4加密后密文
{
    "code": 200,
    "data": {
```

```
        "bizContent": "NI4ry+D01kDXrNf5OcmAzMGaB0td9kXfa321xmIS3qw5homFdZT4xaJu8Vqid37NDivdqj1ADzd0+v+QDKNovFSBkmOyOQt2ObJ4HW43i0dKKHAKlI3FtZA0F9URhlqXbckbpGMD1hev3XS/b9uxRw+QLOnTzhuzgoxpiOXNA9rmQs3V7prpF9Wpaetts2pv",
        "reqTimestamp": 1692693665800,
        "sign": "签名方式见文末加密算法"
    },
    "msg": ""
}

//国密m4加密前
{
    "code": 200,
    "data": {
        "bizContent": {
            "result": false,
            "resultInfo": {
                "personFxpfjxCode": 200,
                "personFxpfjx": "001010" //命中结果由 6 位 0|1 的字符串组成，分别为 第 1 位-其他、第 2 位-前科、第 3 位-涉毒、 第 4 位-吸毒、第 5 位-在逃、第 6 位-涉案(包括在逃撤销)。 值为 1 表示命中，0 表示未 命中。
            }
        },
        "reqTimestamp": 1692693665800,
        "sign": "签名方式见文末加密算法"
    },
    "msg": ""
}
```

**国密4加解密算法及签名认证**

```
// pom.xml中引入
<dependency>
    <groupId>cn.hutool</groupId>
    <artifactId>hutool-all</artifactId>
    <version>5.7.0</version>
</dependency>
<dependency>
    <groupId>org.bouncycastle</groupId>
    <artifactId>bcprov-jdk15to18</artifactId>
    <version>1.70</version>
</dependency>
```

```
import cn.hutool.core.util.StrUtil;
import cn.hutool.crypto.SecureUtil;
import cn.hutool.crypto.SmUtil;
```

```java
import cn.hutool.crypto.symmetric.SymmetricCrypto;

@Slf4j
public class Sm4Util {

    /**
     * 加密
     *
     * @param privateKey 管理员提供
     * @param str
     * @return
     */
    public static String encrypt(final String privateKey, final String str) {
      if (StrUtil.isBlank((CharSequence) str)) {
        return "";
      }

       try {
            SymmetricCrypto sm4 = new SymmetricCrypto("SM4/ECB/PKCS5Padding",
privateKey.getBytes());
            return sm4.encryptHex(str, Charset.forName("UTF-8"));
        } catch (Exception e) {
            log.error("加密失败", e);
            return null;
        }
    }

    /**
     * 解密
     *
     * @param privateKey 管理员提供
     * @param str
     * @return
     */
    public static String decrypt(final String privateKey, final String str) {
        if (StrUtil.isBlank((CharSequence) str)) {
            return null;
        }
        try {
            SymmetricCrypto sm4 = new SymmetricCrypto("SM4/ECB/PKCS5Padding",
privateKey.getBytes());
            return sm4.decryptStr(str, Charset.forName("UTF-8"));
        } catch (Exception e) {
            log.error("解密失败", e);
            return null;
        }
    }

   /**
```

```
     * 验证签名
     *
     * @param appKey
     * @param appSecret
     * @param sign
     * @param timestamp
     * @return
     */
    public static Boolean checkSign(String appKey, String appSecret, String sign, String timestamp) {
        String dbSign = SecureUtil.md5(new StringBuilder().append(appKey)
                .append(appSecret)
                .append(timestamp).toString()).toLowerCase();
        if (!dbSign.equals(sign)) {
            return Boolean.FALSE;
        }
        return Boolean.TRUE;
    }

    /**
     * 获取签名
     *
     * @param appKey
     * @param appSecret
     * @param timestamp
     * @return
     */
    public static String getSign(String appKey, String appSecret, String timestamp) {
        String sign = SecureUtil.md5(new StringBuilder().append(appKey)
                .append(appSecret)
                .append(timestamp).toString()).toLowerCase();
        return sign;
    }

}

```