兴业银行

INDUSTRIAL BANK CO.,LTD.

收付直通车商户通知接口

目录

一 接入准备

1.1 兴业账号

网上商户只要有兴业账号（企业账号、个人借记卡号），即可和兴业银行签订收付直通车合作协议，在商户网站上实现使用各大银行账户（包括借记卡、信用卡、对公账户）对客户购买的订单进行支付。

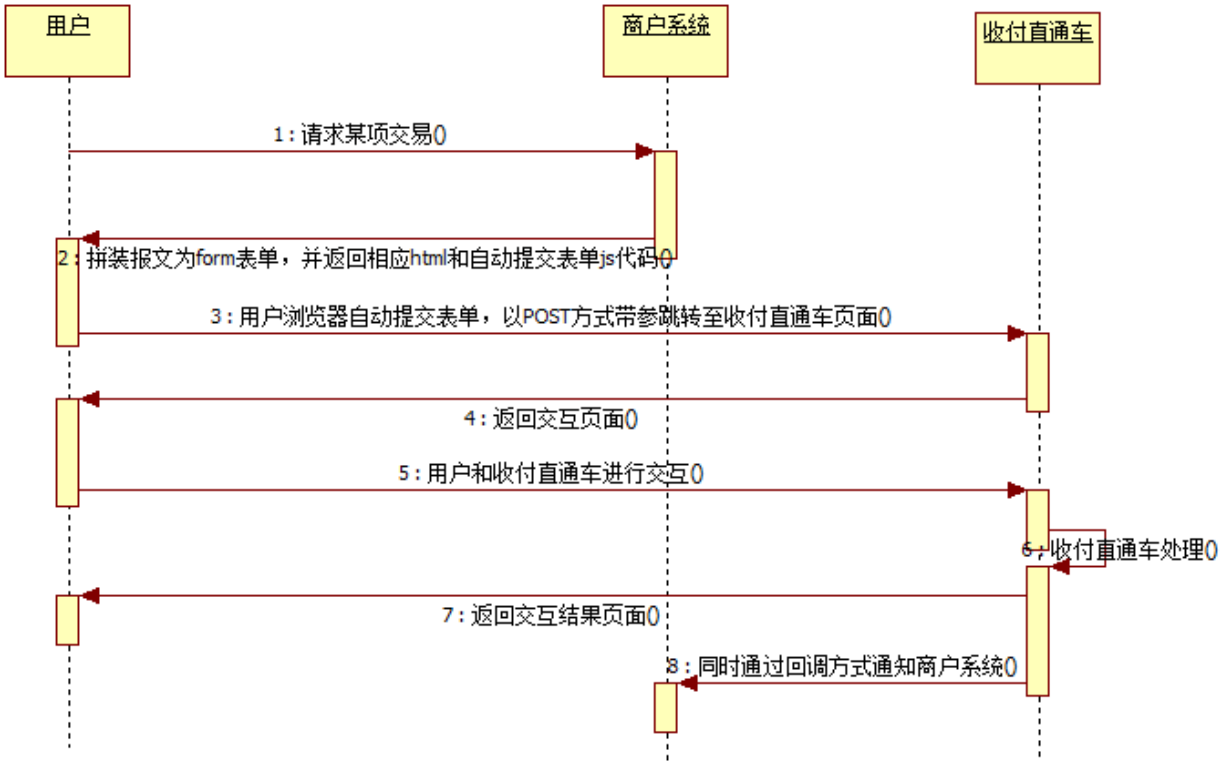
兴业银行收付直通车基于 https 协议，向签约商户提供 WEB 服务。

1.2 通讯模式

收付直通车与商户或用户间的通讯方式均为 HTTPS 协议方式，商户均采用 POST 方法发送数据，收付直通车返回给商户的数据为 JSON 格式字符串。对于下载类业务，业务成功时返回的数据类型为 ZIP 格式压缩包，业务失败时返回的类型为 JSON 格式字符串，可以根据 HTTPS 头部的 Content-Type 判断。根据是否需要用户参与交互，分为两种通讯模式：有用户参与通讯模式和无用户参与通讯模式。

1.2.1 有用户参与通讯模式

当需要用户与收付直通车进行交互时，使用这种通讯方式。商户系统将报文数据拼装好后，发送至用户浏览器，用户浏览器将商户拼装的数据以 POST 方式（即提交 method=post 类型的 form 表单）跳转至收付直通车页面进行继续操作。流程图如下：



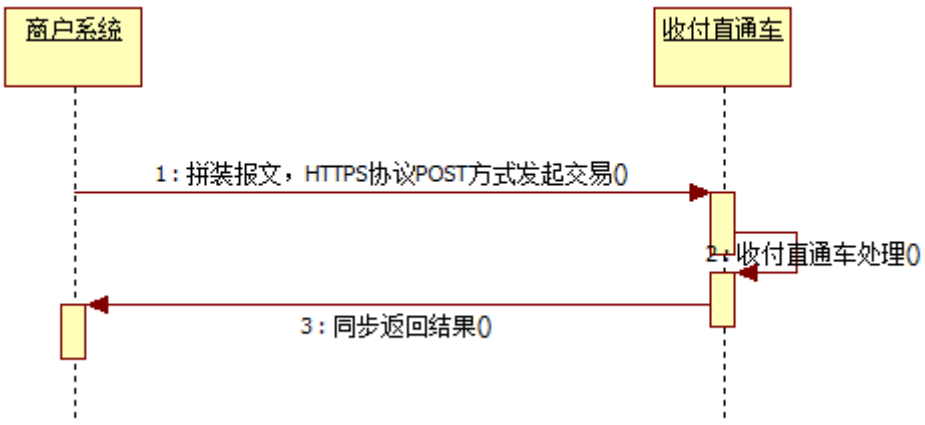
第 2 步中，html 和 js 代码示例（以网关支付为例）：

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>收付直通车跳转示例</title>
</head>
<form                  id="epayredirect"                  name="epayredirect"
action="https://3gtest.cib.com.cn:37031/acquire/cashier.do" method="post">
<input type="hidden" name="order_no" value="DD2015082788899WG"/>
<input type="hidden" name="order_amount" value="1.00"/>
<input type="hidden" name="order_title" value="示例订单"/>
<input type="hidden" name="order_desc" value="这笔订单是示例订单"/>
<input type="hidden" name="order_ip" value="59.83.203.110"/>
<input type="hidden" name="appid" value="Q0000001"/>
<input type="hidden" name="service" value="cib.epay.acquire.cashier.netPay"/>
<input type="hidden" name="ver" value="01"/>
<input type="hidden" name="sub_mrch" value="示例商城"/>
<input type="hidden" name="cur" value="CNY"/>
<input type="hidden" name="order_time" value="20150827134530"/>
<input type="hidden" name="timestamp" value="20150827134530"/>
<input                        type="hidden"                        name="mac"
value="00000000000000000000000000000000000000000"/>
<input type="submit" value="submit" style="display:none;">
</form>
<script>document.forms["epayredirect"].submit();</script>
<body></body>
</html>
```

其中各个<input>中为需要发送的数据项，注意页面需要使用 UTF-8 编码，上述 MAC 值仅为示例。

1.2.2 无用户参与通讯模式

这种通讯方式下不需要用户进行参与，商户系统服务器直接与收付直通车服务器进行 HTTPS 协议 POST 方式通讯。流程图如下：



这种方式下，除文件下载类交易有可能返回 ZIP 格式压缩包外，其余均为返回 JSON 格式字符串。

1.3 接入与测试

1.3.1 环境地址

支付类型	环境类型	URL
快捷支付	生产环境	https://pay.cib.com.cn/acquire/easypay.do
	测试环境	https:// 220.250.30.210:37031/acquire/easypay.do
网关支付	生产环境	https://pay.cib.com.cn/acquire/cashier.do
	测试环境	https:// 220.250.30.210:37031/acquire/cashier.do
托收支付	生产环境	https://pay.cib.com.cn/acquire/easypay.do
	测试环境	https:// 220.250.30.210:37031/acquire/easypay.do
代付地址	生产环境	https://pay.cib.com.cn/payment/api
	测试环境	https:// 220.250.30.210:37031/payment/api

注意事项：

1. 商户秘钥可通过兴业银行收付直通车商户平台设置，请勿与测试环境密钥一致。
2. 商户秘钥可设置为 8-32 位，建议在 10 位以上。
3. 生产上建议定期更换商户秘钥，例如一年一换。
4. 测试环境应监管要求网络互访需要对接运营人员开通网络白名单。
5. 兴业银行返回的应答或通知消息可能会由于升级增加参数，请验证应答签名或处理应答报文时注意允许这种情况。

1.3.2 SDK 开发包

收付直通车目前提供 JAVA 和 PHP 语言两种 SDK 开发包，商户可以参考 SDK 开发包进行开发联调，以加快开发效率。商户也可以根据需要直接修改 SDK 中的源代码以符合自己系统的用途。

需要特别注意的是，SDK 提供的接口对传入参数没有作检查和过滤，商户在传入参数前，需要自行对传入的参数进行检查和过滤。特别是生成跳转页面代码的接口中，请务必对传入参数进行安全性检查，防止出现 XSS 攻击等安全问题。

SDK 相关工程由对接运营人员提供。

1.3.2.1 JAVA

JAVA 语言 SDK 开发包为 J2EE 工程，支持 JAVA 版本 1.6+，可以使用 Eclipse 直接导入工程使用。建议使用前先阅读 SDK 包中 Readme.txt 文件。

调用 API 的示例，详细示例请参见 Example 类中说明：

```
//设置参数
Configure.setAppid("A0000093");                    //商户 ID，appid
```

```
Configure.setCommKey("85EDE15FE3654471B3117D9F2BC9F3D4");    //商户秘钥
Configure.setSub_mrch("SDK 测试商城");          //二级商户名称
Configure.setNeedChkSign(true);                //应答是否验签，开发时可以设成 false
进行调试


// 调用快捷支付 API
String result_json = EPay.epPay("DD20150825KJ", "100.00", "示例订单标题", "示例订
单内容", "6222801234567888953");
```

其中所有对外交易接口均在 com.cib.epay.sdk.EPay 类中。

签名和验签接口在 com.cib.epay.sdk.util.Signature 类中。

epay_notify.jsp 页面为商户回调示例页面，商户需要添加自己的业务逻辑处理代码。

epay_redirect.jsp 该页面为跳转接口示例，即使用有用户参与通讯模式的示例。仅供参考，商户可以根据需要自行实现。同时该页里面也需要加上商户自己的业务逻辑，如订单状态变更为支付中等。

所有示例在 com.cib.epay.sdk.Example 类中，该类请不要包含在商户工程中。

example.jsp 为需要用户交互的接口示例，商户可以在浏览器中打开该页面查看示例。同样，该文件不要包含在商户工程中。

需要注意的是：

1. 该工程为 UTF-8 编码，如果商户系统不是该编码，需要自行进行修改或编码转换；
2. 工程中使用的 HTTPS 通讯方式为 HttpsURLConnection，如果商户想使用其它方式，如 HttpClient 等，可以自行实现一个 IRequestService 接口，并在 Configure 类中修改 httpsRequestClassName 变量为你实现的类的全名；
3. 调用任何 API 之前，请先设置 Configure 参数类，具体设置方法可以参考 Example 类中的示例；
4. com.cib.epay.sdk.Example 类和 example.jsp 页面为示例，请不要包含在商户系统中。
5. 日期时间必须为北京时间，注意操作系统中的时区和时间设置。如果收付直通车提示“订单已过期”等类似提示，请检查系统时间是否正确。
6. 若提示{"errcode":"EPAY_29098","errmsg":"[EPAY_29098]应答消息验签失败，交易未决"}，可在开发调试时设置 Configure.setNeedChkSign(false)，不验证应答签名，待调试正常后再改回 true。生产上建议设置为 true。

1.3.2.2 PHP

PHP 版本 SDK 支持 PHP 版本 5.3+，可以使用 Eclipse for PHP 直接导入工程中。建议使用前先阅读 SDK 包中 Readme.txt 文件。

调用 API 的示例，详细示例请参见 example.php 中说明：

```
require_once('lib/epay_core.class.php');
$epay = new EPay($epay_config);    //使用参数实例化 EPay 类
$result_json = $epay -> epPay("DD20150825KJ", "100.00", "示例订单标题", "示例订单
内容", "6222801234567888953");            //调用快捷支付 API
```

包含的文件如下：

1. epay.config.php：参数示例。PHP 版本 SDK 中使用的参数为一个数组，在实例化接口类时，需

要传入该数组。具体字段请打开该文件查看。

2. lib/epay_core.class.php：收付直通车全部接口类，实例化该类时请传入参数数组，示例参数数组在 epay.config.php 文件中。
3. lib/epay_util.class.php：工具类，epay_core.class.php 中包含该文件。
4. lib/ca-bundle.crt：可信 CA 证书。商户可以自行互联网上下载该证书列表。也可以直接使用该文件。
5. epay_redirect.php：该页面为跳转接口示例，即使用有用户参与通讯模式的示例。仅供参考，商户可以根据需要自行实现。同时该页里面也需要加上商户自己的业务逻辑，如订单状态变更为支付中等。
6. epay_notify.php：该页面为商户回调示例页面，商户需要添加自己的业务逻辑处理代码。
7. example.php：所有 API 调用示例，请直接打开该文件源代码进行查看。该文件请不要包含在商户工程中。
8. index.php：需要用户交互的接口示例，商户可以在浏览器中打开该页面查看示例。同样，该文件不要包含在商户工程中。

需要注意的是：

1. 本 SDK 需要使用 php_curl 扩展作为 http 通讯使用，因此，请确保商户系统运行的 PHP 环境中包含 php_curl 扩展。Windows 下可以去掉 php.ini 中 extension=php_curl.dll 这行前面的注释并重启服务，即可启用 php_curl 扩展。
2. 若你的环境无法启用 php_curl 扩展，需要自行修改 epay_util.class.php 文件中的 getHttpPostResponse 方法。
3. epay_util.class.php 文件中上方 4 个常量为客户端失败时（如网络异常等），返回的错误提示，非服务端返回内容。因此，你可以自己修改为需要的内容，也可以直接使用原有文件中的内容。
4. 调试时，可以去掉 epay_util.class.php 文件 getHttpPostResponse 方法中 var_dump(curl_error($curl));和 var_dump(curl_getinfo($curl));两行前面的注释，来显示 curl 通讯过程中的信息。
5. epay.config.php 中的参数为示例参数，商户需要修改为自己的配置。商户系统可以根据需要在其它地方初始化这些参数，只要在实例化 EPay 类时，传入配置参数数组即可。
6. 日期时间必须为北京时间，注意操作系统或 php.ini 中的时区和时间设置。如果收付直通车提示“订单已过期”等类似提示，请检查系统时间是否正确。
7. 该工程为 UTF-8 编码，如果你的系统不是该编码，你需要自行修改或进行编码转换。
8. 若提示{"errcode":"EPAY_29098","errmsg":"[EPAY_29098]应答消息验签失败，交易未决"}，可在开发调试时设置$epay_config['needChkSign'] = false，不验证应答签名，待调试正常后再改回 true。

生产上建议设置为 true。

1.4 商户秘钥

代付部分使用商户秘钥包含两种：用于 MAC 校验算法使用的商户秘钥，一般为 32 位十六进制字符串，如下所示：8D0565EF811F4C61806A6301C9DF8726

另外一种，用于 sha1WithRSAEncryption 签名算法使用的客户证书文件，一般为 pfx 文件。

注意事项：

1. 为确保订单信息真实有效不被篡改，每个商户均应在兴业银行收付直通车商户平台中设定商户商户秘钥。
2. 商户秘钥作为商户与银行间的信任关键，双方都应对其保密。
3. 商户秘钥内容只能由数字和字母组成，其中字母区分大小写。
4. 为避免商户秘钥泄漏，生产环境上请勿使用与测试环境相同的商户秘钥。
5. 商户秘钥应定期更换，至少应每年更换一次。
6. 对于客户证书文件，收付直通车服务器上没有保存副本，商户若遗失，只能进行重置操作。

1.4.1 签名/加密算法

目前收付直通车代付部分使用两种算法：SHA1WithRSAEncryption 签名、MAC 校验（SHA1）；鉴于安全考虑，后续将逐步淘汰掉 SHA1-MAC 校验方法。

1.4.1.1 SHA1 方式 MAC 校验

该算法适用于 sign_type=SHA1 时计算 mac 参数的值。以以下待发送参数，商户秘钥 84A8251FECD84E599B7B1037579E6A4D 为例，具体说明计算步骤：

```
timestamp=20150708111509
appid=P0000014
service=cib.epay.payment.receiptFile
ver=01
trans_date=20150623
rcpt_type=1
```

1. 拼装消息内容

将键值对以 key=value 形式组装成一个数组，将该数据进行排序后使用&进行拼装，最后&上商户秘钥。

此步需要注意的是：

编码需要使用 UTF-8；

参数中不能出现类似&等特殊字符

此时不需要对参数键、值等进行 URLEncode 操作。

```
appid=P0000014&rcpt_type=1&service=cib.epay.payment.receiptFile&timestamp=2015
0708111509&trans_date=20150623&ver=01&84A8251FECD84E599B7B1037579E6A4D
```

2. 计算 SHA-1 值

使用 hash 算法 SHA-1 计算上述字符串的 hash 值，得到 16 进制形式的字符串（共 40 个字母），并全部转换为英文字母大写形式。如:30F2B0CA895BB25881CEE20CB32E5EEADCB9E18D

3. 上步得到的字符串即作为发送参数中 mac 参数的值。

注意：若参数为空，可不发送（不加入到拼装字符串中），也无需加入计算 mac；若要发送，相应的也需加入去计算 mac。

1.4.1.2 SHA1WithRSAEncryption 签名验签

该算法适用于 sign_type=RSA 时计算 mac 参数的值。

具体做法可以参考示例代码或 SDK 中相应代码：

1. 参考 1.5.1.1 中第 1 步，将参数以 key=value 形式组装成一个数组，将该数据进行排序。注意，此时不需要再&上商户秘钥。注意编码必须为 UTF-8。
2. 从客户证书文件中获取商户 RSA 私钥。行方提供的 PFX 文件默认密码为 123456，别名(alias)为 appsvr_client。
3. 使用标准的 SHA1WithRSAEncryption 签名算法对该字符串进行签名，得到的值（base64 格式字符串）即为 mac 参数的值。

1.4.2 算法代码示例

1.4.2.1 SHA1 示例

【JAVA 示例】

```
/**
     * 生成用于 MAC 计算的参数字符串。<br>
     *
     * @return 模式为 key=value&key=value
     */
    public static String generateParamStr(Map<String, String> params) {
        // 取所有非空字段内容（除 mac 以外），塞入列表
        List<String> paramList = new ArrayList<String>();
        for (String key : params.keySet()) {
            if (“mac”.equals(key)) {
```

```
                continue;
            }
            String val = params.get(key);
            paramList.add(key + "=" + val);
        }
        // 防护
        if (paramList.size() == 0) {
            return null;
        }
        // 对列表进行排序
        Collections.sort(paramList);
        // 以&符分割拼装成字符串
        StringBuilder sb = new StringBuilder();
        sb.append(paramList.get(0));
        for (int i = 1; i < paramList.size(); i++) {
            sb.append("&").append(paramList.get(i));
        }
        return sb.toString();
    }

/**
     * 将 byte 数组转换为 16 进制格式的字符串
     *
     * @param bytes 待转换数组
     * @return 16 进制格式的字符串
     */
    public static String bytesToHexStr(byte[] bytes) {
        StringBuffer sb = new StringBuffer(bytes.length * 2);
        for (int i = 0; i < bytes.length; i++) {
            sb.append(hexChar[(bytes[i] & 0xf0) >>> 4]);
            sb.append(hexChar[bytes[i] & 0x0f]);
        }
        return sb.toString();
    }

private static char[] hexChar = { '0', '1', '2', '3', '4', '5', '6', '7', '8',
'9', 'A', 'B', 'C', 'D', 'E', 'F' };

/**
     * SHA 摘要算法，输入内容将被 UTF-8 编码
     *
     * @param content 输入明文
     * @return 内容摘要，40 位 16 进制字符串
     */
    public static String encryptBySHA(String content) {
        if (content == null)
            return null;

        try {
            MessageDigest md = MessageDigest.getInstance("SHA-1");
            byte[] output = md.digest(content.getBytes("UTF-8"));
            returnbytesToHexStr(output);
        } catch (NoSuchAlgorithmException e) {
            log.error("无法加载 SHA 算法。", e);
        } catch (UnsupportedEncodingException e) {
            log.error("无法将输入字符串转换到 utf-8 编码。", e);
        }
```

```
        return null;
    }
```

【.Net 示例】

```
/**
 * 生成用于 MAC 计算的参数字符串。
* @return 模式为 key=value&key=value
 */
public static string generateParamStr(Dictionary<string,string> _params) {
    // 取所有非空字段内容（除 mac 以外），塞入列表
    ArrayList paramsList = new ArrayList();
    foreach (KeyValuePair<string,string> key in _params){
        if (key.Key.Equals("mac")) {
            continue;
        }
        paramsList.Add(key.Key + "=" + key.Value);
    }
// 防护
    if (paramsList.Count <= 0){
        return null;
    }
 // 对列表进行排序
    paramsList.Sort();

    // 以&符分割拼装成字符串
    string sb="";
    for (int i = 0; i < paramsList.Count; i++){
         sb += paramsList[i].ToString() + "&";
    }
    sb = sb.TrimEnd('&');
    return sb;
}

/**
 * SHA 摘要算法，输入内容将被 UTF-8 编码
 *
 * @param content 输入明文
 * @return 内容摘要，位进制字符串
 */
public static string encryptBySHA(string content) {
    if (content == null)
        return null;
    try {
        SHA1CryptoServiceProvider sha = new SHA1CryptoServiceProvider();
        byte[] str2 = sha.ComputeHash(Encoding.UTF8.GetBytes(content));
        sha.Clear();
        (sha as IDisposable).Dispose();
StringBuilder sb = new StringBuilder();
for (int i = 0; i < str2.Length; i++) {
sb.Append(str2[i].ToString("X2"));
        }
        return sb.ToString();
        } catch{
        //log.error("无法加载 SHA 算法。", e);
```

```
        throw;
    }
}
```

【PHP 示例】

```
/**
 * 根据参数数组生成 MAC 码。
* @return MAC 码
 */
public static function Signature($param_array, $commkey) {
     ksort($param_array);
     reset($param_array);
     $signstr = '';
     foreach ($param_array as $k => $v) {
         if(strcasecmp($k, 'mac') == 0) continue;
         $signstr .= "{$k}={$v}&";
     }
     $signstr .= $commkey;
     return strtoupper(sha1($signstr));
  }
```

1.4.2.2 SHA1WithRSAEncryption 示例

【JAVA 示例】

```
/**
     * 生成用于 MAC 计算的参数字符串。<br>
     *
     * @return 模式为 key=value&key=value
     */
    public static String generateParamStr(Map<String, String> params) {

        // 取所有非空字段内容（除 mac 以外），塞入列表
        List<String> paramList = new ArrayList<String>();
        for (String key : params.keySet()) {
            if (“mac”.equals(key)) {
                continue;
            }
            String val = params.get(key);
            paramList.add(key + "=" + val);
        }
        // 防护
        if (paramList.size() == 0) {
            return null;
        }
        // 对列表进行排序
        Collections.sort(paramList);
        // 以&符分割拼装成字符串
        StringBuilder sb = new StringBuilder();
        sb.append(paramList.get(0));
        for (int i = 1; i < paramList.size(); i++) {
            sb.append("&").append(paramList.get(i));
```

```
        }
        return sb.toString();
    }

/**
  * 生成签名 MAC 字符串
  * @param       参数列表（包含 mac 参数）
* @param     证书路径（带文件名）
* @param     证书密码
  * @return      MAC 字符串
     */
public static String generateMAC(Map<String, String> params, String certPath,
String certPwd) {
     try {
     PrivateKey privKey = readPrivateKey(certPath, certPwd);
             java.security.Signature              signature              =
java.security.Signature.getInstance("SHA1WithRSA");
             signature.initSign(privKey);
             signature.update(generateParamStr(params).getBytes());
             byte[] signed = signature.sign();

             // 计算 base64encode(signed)，无换行。如无法使用，请自行替换为其它
BASE64 类库。
             @SuppressWarnings("restriction")
             String mac = new sun.misc.BASE64Encoder().encode(signed)
          .replaceAll(System.getProperty("line.separator"), "");
             return mac;
     } catch (Exception e) {
     // 读取证书等出错
     return "SIGNATURE_RSA_CERT_ERROR";
     }
}
```

【PHP 示例】

```
/**
 * 根据参数数组生成 MAC 码（sha1WithRSAEncryption 方式）。
* @return MAC 码
 */
public static function Signature($param_array, $commkey) {
     ksort($param_array);
     reset($param_array);
     $signstr = '';
     foreach ($param_array as $k => $v) {
         if(strcasecmp($k, 'mac') == 0) continue;
         $signstr .= "{$k}={$v}&";
     }
     $signstr = substr($signstr, 0, strlen($signstr) - 1);
     if (false !== ($keystore = file_get_contents($cert)) &&
              openssl_pkcs12_read($keystore, $cert_info, $cert_pwd) &&
              openssl_sign($signstr,      $sign,      $cert_info['pkey'],
'sha1WithRSAEncryption')) {
              return base64_encode($sign);
     } else{
     return 'SIGNATURE_RSA_CERT_ERROR';
     }
  }
```

1.5 注意事项

1. 输入数据都通过 key-value 的参数列表形式，使用 http post 提交到服务接入地址。

2. 数据中不能包含"&"、"="、"?"等特殊字符。

3. 通讯报文使用 UTF-8 编码。

4. 商户提交数据中的空格将被认为是有效字符被接收，请提交时注意对多余空格的控制。

5. 测试环境中快捷认证的手机号请输入格式正确的手机号，测试环境并不会真正的发送短信验证码，短信验证码请输入任意 6 位数字即可。

6. 测试环境中，根据邮路不同，快捷支付返回的支付结果可能是随机结果，如果需要成功支付的结果，可以尝试支付多次。

7. 快捷支付身份认证，每张银行卡每天连续认证失败 6 次后当天将无法继续进行快捷支付身份认证交易，请第二天再尝试或更换银行卡。

8. 兴业银行返回的应答或通知消息可能会由于升级增加参数，请验证应答签名或处理应答报文时注意允许这种情况。

9. 鉴于安全考虑，后续将逐步淘汰掉 SHA1-MAC 校验方法。

10. 判断交易是否正确，应以正常返回交易状态（trans_status）进行判断；除非特别说明，否则异常返回都应当成“支付中”、“处理中”的中间状态来进行处理。

二 通知接口

通知接口为收付直通车向商户系统发送支付成功、退款成功等事件通知。在对应事件发生后，由收付直通车系统发起。每个商户可以设置一个通知回调 URL，当对应事件发生时，根据事件不同，收付直通车会通过 HTTP 协议的 GET 或（和）POST 方法，将参数发给商户系统。

商户回调 URL 可以联系对应业务人员进行添加或修改操作，也可以通过企业管理员账号，自行在商户平台上进行配置，配置后 5 分钟内生效。

2.1 支付成功通知

2.1.1 功能概述

方向：银行→商户

方法： POST

由兴业银行收付直通车发往商户监听地址，属于事件通知，无需返回。网关支付成功时，商户合约中若有配置服务端通知，则发送通知，否则不发送通知；网关通知最多通知 3 次，收到正常响应（HTTP 200）之后不再通知。

2.1.2 接口说明

参数名称	类型	必输	示例值	描述
event	varchar(60)	√	NOTIFY_ACQUIRE_SUCCESS	事件类型： 成功通知事件：NOTIFY_ACQUIRE_SUCCESS
timestamp	char(14)	√	20140831212747	时间戳
以下为支付成功通知内容				
appid	char(8)	√	A0000012	兴业银行分配给商户的 AppID
order_no	varchar(32)	√	20141023163955812	已支付的订单号
order_amount	dec(13,2)	√	10.50	订单金额，单位为元，即：十位整数，两位小数
order_time	char(14)	√	20141023163955	商户端订单生成时间 yyyyMMddHHmmss
pay_time	char(14)	√	20141210094045	付款时间 yyyyMMddHHmmss
sno	char(16)	√	2014102300233000	支付网关流水号
以下为签名相关内容				

mac	varchar(40)	√		消息校验码
sign_type	varchar(4)		RSA	RSA

2.1.3 接口示例

支付成功通知根据配置包含前台和（或）后台两种通知。前台通知为用户点击支付成功页面上的链接，将参数带至商户服务器上的某个 URL 的方式（HTTP GET 方式）将支付成功消息通知至商户；后台通知为收付直通车服务器直接向商户服务器发送一带参数的 HTTP POST 请求的方式通知商户。目前，快捷支付支付成功通知为后台通知；网关支付支付成功通知根据后台配置，可以包含前台通知和后台通知两种。

【业务成功结果】

```
appid=Q0001388
event=NOTIFY_ACQUIRE_SUCCESS
mac=K4b4SGFEir6vbQ89okEj%2FuyK%2FsfcKPEPcFF4swV%2B9U6ORrmaRBwXd4FAiD0z1Aako5Js
j9dTU6rei3anm7hXPdXzUmCI2HjJJ3QUWJLLcUTxJi7zq9cALiD2NQTMaWBW7L6ZX0nu%2B4dIPvhu
g58olGMZbk3MctdKfaPwMnxi4uBN9Q7Cq%2FxWwLwZlvmUE9m6TmXyo21JSflUuD9kITBXRmDwKm2g
FaBcCCCl3xXPQbKyKNzUl3cYiJOQIYetrPOfJQRR3AAtoDQeVLDcLqDH6mLPPjKWPpMrNUSZdQjyHl
%2FNg0njzMl9ofo05uTmFzhaWL4oGAsBaSWQ5kAHKYcbSg%3D%3D
order_amount=10.80
order_no=20171123143002529
order_time=20171123143002
pay_time=20171123143845
sign_type=RSA
sno=2017112305264161
timestamp=2017112314384
```

2.2 退款结果通知

2.2.1 功能概述

方向：银行→商户

方法：POST

由兴业银行收付直通车发往商户监听地址，属于事件通知，无需返回，仅退款成功后台异步发送通知。

2.2.2 接口说明

参数名称	类型	必输	示例值	描述
event	varchar(60)	√	NOTIFY_REFUND_SUCCESS	事件类型：NOTIFY_REFUND_SUCCESS
timestamp	char(14)	√	20140831212747	时间戳
以下为退款成功通知内容				
appid	char(8)	√	A0000012	兴业银行分配给商户的

				AppID
order_no	varchar(32)	√	20141023163955812	由商户分配的订单号
order_amount	dec(13,2)	√	10.50	订单金额，单位为元，即：十位整数，两位小数
order_date	char(8)	√	20141023	商户端订单生成时间 yyyyMMdd
refund_date	char(8)	√	20141210	退款时间 yyyyMMdd
sno	char(16)	√	2014102300233000	收付直通车支付网关流水号
以下为签名相关信息				
mac	varchar(40)	√		消息校验码
sign_type	varchar(4)		RSA	RSA

2.2.3 接口示例

【输出示例】

```
appid=Q0001388
event=NOTIFY_REFUND_SUCCESS
mac=Sd5yOhbAwDLNwfc6lQzEdxQfm4wWB%2B%2BFM2spo3cTimV9YsmUAmiFBEPP2byGQqgDqUCICJ
0%2F0h4sy5NZ1D%2B9aU7EUGPtkxce5%2Fi7qXZM4X6IZKXQiVuul5IQWVbcRU0gFL4MLnHH89%2Fj
9Kaxsfmo%2BPKOQIleHcweW9awdjxuAIitbHefoT2mjV7IXbN8q3c6bTxvFM8JOQe8Qeiej%2FVWqI
6OEQ9achG49koSm6chzVVynslf80SD4o9Q4T57Z%2BtUymfsYjvspzm%2FdO8zRVKY6w1vKVztT93u
uST3iJtBl4s4vLG4TIg79CHuVbuuVJaahxTg9tFEoj5Owa4qOCrbdw%3D%3D
order_amount=10
order_date=20171123
order_no=20171123145242063
refund_date=20171123
sign_type=RSA
sno=2017112305264175
timestamp=20171123150410
```

2.3 支付失败通知

2.3.1 功能概述

方向：银行→商户

方法：POST

由兴业银行收付直通车发往商户监听地址，属于事件通知，无需返回。仅针对内部商户开放，网关支付有通知，快捷支付无通知。

2.3.2 接口说明

支付成功/失败均向客户系统发起通知，客户主动终止时，不会向商户发送成功通知。

参数名称	类型	必输	示例值	描述
event	varchar(60)	√	NOTIFY_ACQUIRE_FAIL	事件类型： 失败通知事件：NOTIFY_ACQUIRE_FAIL
timestamp	char(14)	√	20140831212747	时间戳
以下为支付失败通知内容				
appid	char(8)	√	A0000012	兴业银行分配给商户的AppID
order_no	varchar(32)	√	20141023163955812	下游系统跟踪号
errcode	varchar(10)	√	20303	错误代码，20303 账户密码或身份信息不正确
errmsg	varchar(255)	√	错误信息	发送错误时返回的错误信息
以下为签名相关内容				
mac	varchar(40)	√		消息校验码
sign_type	varchar(4)		RSA	只支持 RSA

2.3.3 接口示例

使用 https 协议将参数 post 到签约商户监听地址。

【通知正文】

```
appid=Q0001388
errcode=20303
errmsg=账户密码或身份信息不正确!
event=NOTIFY_ACQUIRE_FAIL
mac=IU%2FhZk7Y5Q8vH6r74W%2BrlbsrwSUDrafpVzDE9ZTNYvPe6jSWXVKbVJ6be6fE0VXxes0oz1
MeaRHHxFOqOzKWl1dP%2FMMJlgsDz%2B0yS15GUa5a5WI2OY8r2u7bL6b
AfAV3p0likRlIhfz1MJ6JD562u0HLFpsYRGK9okunWqf%2B229fT7OGZHN74vP16c3T73mTmaIli9h
DcfwB7y1fXW%2FwOr%2FkAtKGSKatReOLHTARLU8RZVuO4PVEg4f36djgyyoDKk3PbaiIseSIZA1fh
6cbL3J9P9Ta6nn2YmLi2c7GkXr5UYyG3cKLuo61NKCdjDfrO1QVn%2BOs23svTGdX7smi
lA%3D%3D
order_no=20171123161529615
sign_type=RSA
timestamp=20171123170001
```

2.4 账户认证通知

2.4.1 功能概述

方向：银行→商户

方法：POST

由兴业银行收付直通车发往商户监听地址，属于事件通知，无需返回,仅认证成功后后台异步发送通知。

2.4.2 接口说明

参数名称	类型	必输	示例值	描述
event	varchar(60)	√	NOTIFY_AUTH_SUCCESS	事件类型：NOTIFY_AUTH_SUCCESS
timestamp	char(14)	√	20170824212747	时间戳
以下为成功时通知内容				
appid	char(8)	√	Q0000870	兴业银行分配给商户的AppID
trac_no	varchar(32)	√	20170824165455000001	下游系统跟踪号
card_no	varchar(32)		6222021402000200	卡号
user_name	varchar(60)		张三	姓名
cert_type	char(1)		0	证件类型
cert_no	varchar(32)		350101199101011234	证件号码
card_phone	varchar(11)		13812345678	手机号码
bank_no	varchar(14)		301290000007	银行代码
bank_type	char(3)			银行类型
以下为签名相关信息				
mac	varchar(40)	√		消息校验码

2.4.3 接口示例

【输出示例】

```
event=NOTIFY_AUTH_SUCCESS
timestamp=20170824212747
appid= Q0000870
trac_no=20170824165455000001
card_no=6222021402000200
user_name=张三
cert_type=1
cert_no=350101199101011234
card_phone=13812345678
bank_no=301290000007
bank_type=301
mac=39745ABAB3B84C9FD28090D49AEB1EA95551D8C2
```