OPPO 广告主转化数据接入说明

方案设计

1、背景

oCPX 智能出价是 OPPO 广告系统通过优化转化率预估模型，帮助广告主提升推广精准度，降低转化成本的一种投放方式；而转化率的预估模型，需要广告主回传相关转化数据；该接口文档主要描述了接口的定义，对接方式以及相关注意事项。

2、接口说明

接口采用 https + post + json 的方式提交数据;
编码方式统一使用 UTF-8;
Content-Type 设置:application/json;charset=UTF-8
Salt: e0u6fnlag06lc3pl
Base64Key: XGAXicVG5GMBsx5bueOe4w==

2.1 转化数据同步接口

接口地址：https://api.ads.heytapmobi.com/api/uploadActiveData

2.1.1 请求信息
(1)请求头(header):

字段	类型	说明	是否必须
signature	string	根据签名算法计算得到的字符串，见签名算法 2.2	是
timestamp	string	时间戳转换为字符串（long 毫秒）	是

(2)数据字段(post body)：

字段	类型	说明	是否必须	样例
imei	string	客户端 imei 经过 AES 加密后的值,加密说明见 2.3	**三选一，但优先用 imei 和 ouId，最后考虑用 androidId;** **ouId 就是 oaid;**	加密前：868123039927020 加密后：XJMyaLt8fDlv4a9b8/0RNQ==
ouId	string	开发者 ID 经过 AES 加密后的值,加密说明见 2.3		
androidId	string	安卓 ID 经过 AES 加密后的值,加密		

		说明见 2.3		
mac	string	客户端 mac 经过 AES 加密后的值，加密说明见 2.3	否	加密前：d7:1b:3e:00:14:b3 加密后：TEViR6jSgD/lECBl3Ah70eNy2gUQrQlekHkWqEGkZsU=
clientIp	string	事件发生时的客户端 ip，如 210.210.210.210	否	127.0.0.1
timestamp	long	事件发生的时间戳(毫秒)，如 1522221766623	是	1571995483916
pkg	string	包名，如 com.xxx	是	com.oppo.test，**要填投放应用的包名，不要依赖监测链接上报的 pkg 字段**
dataType	int	转化数据类型：1、激活，2、注册，3、游戏付费，4、次留，5、应用内授信 6、应用内下单（电商）7、应用付费 8、自定义目标 9、第 3 日留存 10、第 4 日留存 11、第 5 日	是	1

		留存 12、第 6 日留存 13、第 7 日留存 14、第 8 日留存 15、拉活		
customType	int	自定义目标类型：dataType 填了 8 之后补充，枚举值与客户沟通后补充	否	
channel	int	渠道：1、OPPO，2、一加，0、其他	是	1
type	int	Imei 原始加密类型 1： md5 加密 0： 无加密（默认为 0） 如果传 oaid，type 值填 0	是	1
appType	int	应用类别：1 应用 2 游戏 3 快应用 0 其他，默认 1 应用	否	1
payAmount	long	付费金额（单位：分）	否	100
ascribeType	int	归因类型：1：广告主归因，0：OPPO 归因（默认或者不填即为 0），2：助攻归因	是	1
adId	long	广告主回传转化数据时，附带已经归因好的广告 id	是	10109 7648
requestId	string	请求 id	否	

2.1.2 请求示例

(1)Headers	
signature	ce14fcc22abd7461e860263a8da983eb （签名获取方式详见 2.2 签名算法）
timestamp	1571995483916
(2)Body	
{"payAmount":100,"adId":101097648,"appType":1,"clientIp":"127.0.0.1","dataType":1,"ascribeType":1,"channel":1,"imei":"XJMyaLt8fDlv4a9b8/0RNQ==","type":1,"pkg":"com.oppo.test","mac":"TEViR6jSgD/lECBl3Ah70eNy2gUQrQlekHkWqEGkZsU=","timestamp":1571995483916}	

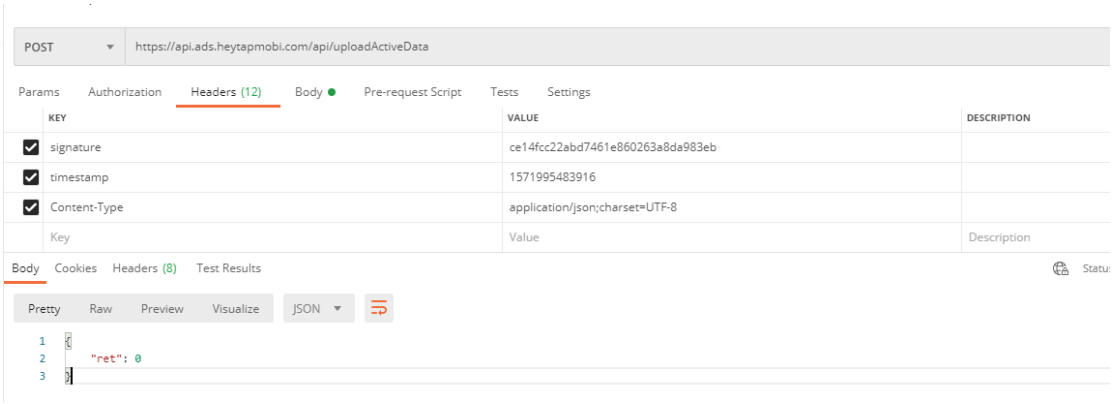
2.1.3 返回参数：

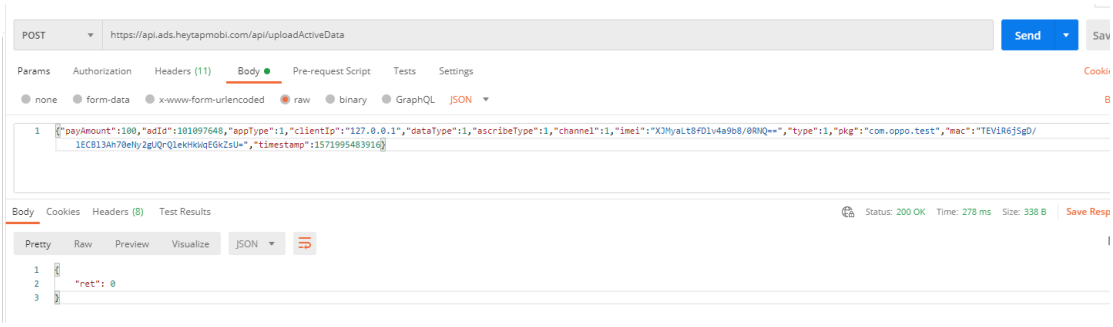
字段	类型	说明	是否必须
ret	int	返回码，0 正常，1001 参数校验失败，1002 未知错误 注意：如果是签名错误的话会返回 http status 403	是
msg	string	如果有错误，则返回错误信息	否

2.1.4 返回示例：

```
{
    "ret":0
}
```

2.1.5 postman 截图：





2.2 签名算法

(1) 签名方式

使用 md5 对数据进行签名： md5(postData + timestamp + salt)

字段	解释
postData	post 的 JSON 格式数据
timestamp	时间戳
salt	OPPO 提供

(2)签名示例

1)数据准备:

- postData 的获取：根据测试样例中获取的 postData 为 {"payAmount":100,"adId":101097648,"appType":1,"clientIp":"127.0.0.1","dataType":1,"ascribeType":1,"channel":1,"imei":"XJMyaLt8fDlv4a9b8/0RNQ==","type":1,"pkg":"com.oppo.test","mac":"TEViR6jSgD/lECBl3Ah70eNy2gUQrQlekHkWqEGkZsU=","timestamp":1571995483916}

注意：postData 为需要上报字段组成 json，该 json 与 2.1.2（2）中 body 一致【注意：该处的一致指 key 和 value 的顺序和值都保持一致】，其中 imei、ouId、androidId、mac 需要进行 AES 加密，具体加密方式可见 2.3 加密算法。使用该 json 计算签名时，不可对该 json 进行添加任何空格和转义符。

- timestamp 的获取：1571995483916
- salt 的获取：e0u6fnlag06lc3pl

2）md5 加密

- 生成加密内容：加密内容由 postData、timestamp、salt 拼接而成，拼接时不需要添加任何字符。如测试用例中根据 postData、timestamp、salt

拼接结果（contents）为：

{"payAmount":100,"adId":101097648,"appType":1,"clientIp":"127.0.0.1","dataType":1,"ascribeType":1,"channel":1,"imei":"XJMyaLt8fDlv4a9b8/0RNQ==","type":1,"pkg":"com.oppo.test","mac":"TEViR6jSgD/lECBl3Ah70eNy2gUQrQlekHkWqEGkZsU=","timestamp":1571995483916}1571995483916e0u6fnlag06lc3pl

- 使用 md5 加密：md5(contents)
- 得到签名结果：ce14fcc22abd7461e860263a8da983eb

注意：签名后结果需转为小写

2.3 加密算法

请求中的 imei 与 mac 字段均需通过 AES 算法进行加密，并且将加密后的二进制数据通过 base64 编码为字符串。为了方便配置，示例中的密钥是经过 base64 处理后的字符串(base64Key)。base64Key 在接入时由 OPPO 提供。

（1）java 示例

```
import java.security.GeneralSecurityException;
import java.security.Key;
import javax.crypto.Cipher;
import javax.crypto.spec.SecretKeySpec;
import org.apache.commons.codec.binary.Base64;

public class Demo {
    public static String encode(byte[] data, String base64Key) throws
GeneralSecurityException {
        final Key dataKey = new SecretKeySpec(Base64.decodeBase64(base64Key),
"AES");

        Cipher cipher = Cipher.getInstance("AES/ECB/PKCS5Padding");
        cipher.init(Cipher.ENCRYPT_MODE, dataKey);
        byte[] encryptData = cipher.doFinal(data);

        return Base64.encodeBase64String(encryptData).replaceAll("\r",
"").replaceAll("\n", "");

    }
}
```

（2）php 示例(仅供参考)

```
public function encrypt($input, $base64Key){
    return openssl_encrypt($input, 'AES-128-ECB',base64_decode($base64Key),0,'');
}
```

3、其他说明

3.1 签名校验说明

若签名校验失败，直接返回 http status code : 403。