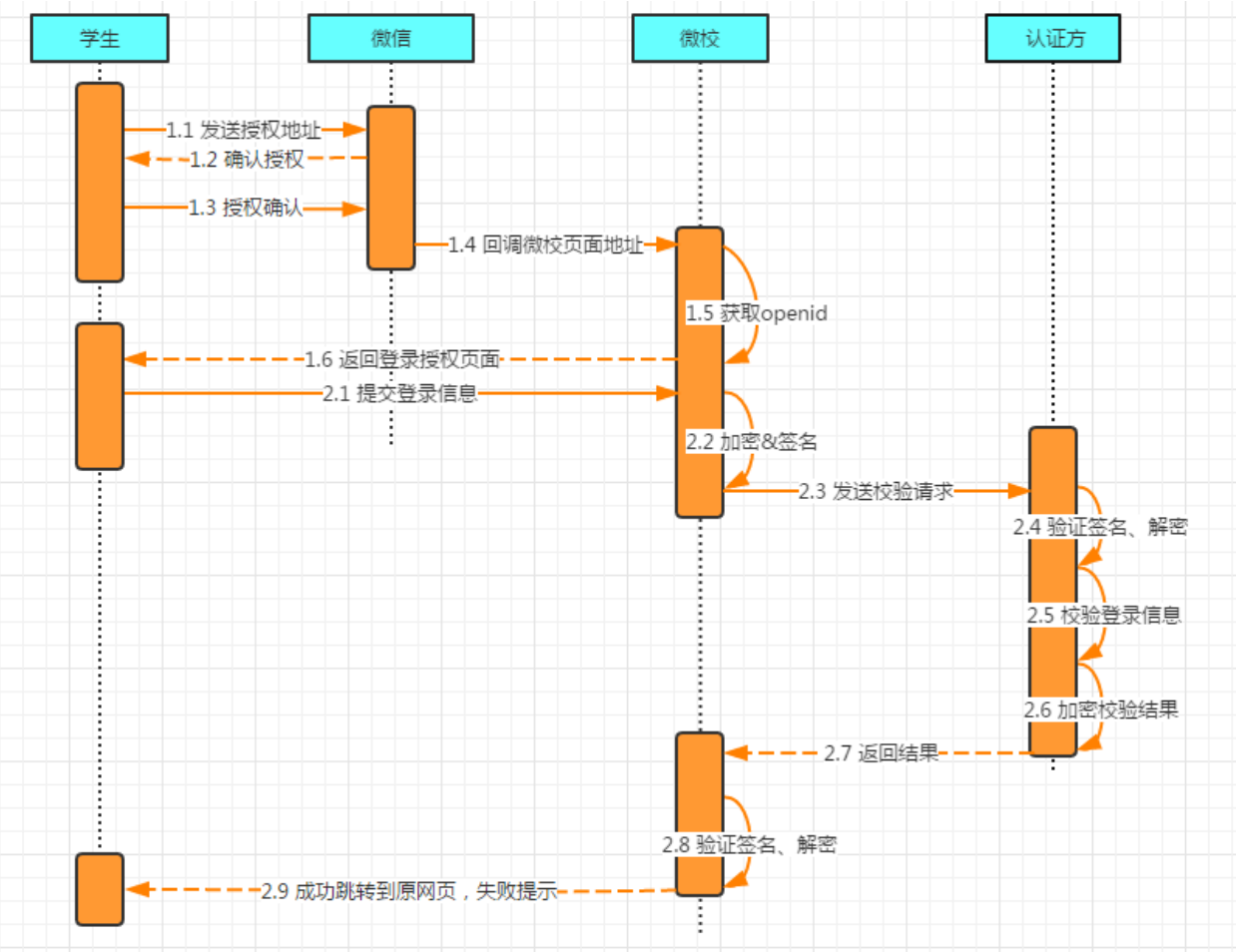
身份验证技术方案

1、身份认证流程

1.1 账号密码验证身份

身份认证是其他应用的基础，认证方（即：校方）需要提供一个验证接口，具体身份绑定流程如下：



身份认证绑定步骤：

- 学生在微信客户端打开应用，触发微信公众号授权（授权页面提示授权给腾讯微校）。
- 微信公众号授权后，回调跳转到微校身份绑定页面，输入校园账号（例如学号）以及相应的密码，
- 微校页面数据发送到微校后台（注:微校后台不会保存学生的账号和密码），微校后台把对应的信息加密同时附上签名发送到认证方的认证接口（认证方需提供校验接口）。
- 认证方验证签名、解密，验证学生身份，返回加密后的校验结果。
- 微校收到对应的验证消息，若成功则跳转到对应的应用页面，失败则做出相应提示。

1.2 签名算法

签名采用微信支付后端签名算法

1.2.1 签名生成的通用步骤

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。
特别注意以下重要规则：

- 参数名ASCII码从小到大排序（字典序）；

- 如果参数的值为空不参与签名；
- 参数名区分大小写；
- 验证调用返回或微信主动通知签名时，传送的sign参数不参与签名，将生成的签名与该sign值作校验。

在stringA最后拼接上key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。

签名算法示例（签名验证地址）：

```
private static function sign($param_array){
    $names = array_keys($param_array);
    sort($names, SORT_STRING);
    $item_array = array();
    foreach ($names as $name){
        $item_array[] = "{$name}={$param_array[$name]}";
    }
    $secret_key =  APP_SECRET;
    $str = implode('&', $item_array) . '&key=' . $secret_key;
    return strtoupper(md5($str));
}
```

1.2.2 APP_KEY & APP_SECRET

由微校生成，每个公众号拥有一对唯一的`APP_KEY`和`APP_SECRET`。
`APP_KEY`和`APP_SECRET`与公众号的关系是一对一。也就是说，不同的公众号，访问同一个身份认证接口，所用的`APP_KEY`跟`APP_SECRET`是不一样的。

1.3 数据加密

采用AES对称加密算法（AES/CBC/ZeroPadding 128位模式），具体算法见附件。

```
KEY = APP_KEY
IV = APP_SECRET 前16位。
```

2、身份认证接口

2.1 请求方式

身份验证接口采用`POST`的方式向认证方发送数据。

2.2 请求参数

原始数据`R`：

```
{
    "card_number":"3109005843",
    "password":"helloworld",
    "app_key":APP_KEY,
    "nonce_str":"",(32位随机字符串)
    "timestamp":"",(时间戳)
    "sign":"9A0A8659F005D6984697E2CA0A9CF3B7"//签名
}
```

通过加密`R`可以得到`R'`：`R' = AES_CBC_ENCRYPT(R)`。

```
{
    "raw_data":R',
    "app_key":APP_KEY
}
```

微校会把上面的数据以`POST`的方式发送到认证方提供的认证接口。

加密算法见附件。

2.3 数据返回

返回数据：

```
{
    "code":0,
    "message":"success",
    "raw_data":R',
    "app_key":APP_KEY
}
```

关键点：

- 认证成功，认证方需保证返回的`code`为 0。

- 认证失败，认证方需保证返回的`code`不为 0。
- `message`字段在`code`不为 0 时才会有意义。
- `app_key`跟请求时的`app_key`保持一致。
- `raw_data`对应的`R'`为加密后的数据。
  通过解密`R'`可得到`R`：`R = AES_CBC_DECRYPT(R')`

```
{
    "card_number":"07302590", //校园账号，一般是学号
    "name":"张三丰", //学生姓名，必填
    "grade":"2016", //年级，必填
    "college":"信息科学与技术学院", //学院
    "profession":"计算机系", // 专业
    "id_card":"4XXX***7", // 身份证号码
    "telephone":"137***8" // 手机号
}
```

其中`name`、`grade`为必填项。

3、附件

3.1 PHP 加解密

```php
<?php
class AES
{
    public static function decrypt($str, $key, $iv)
    {
        $decrypt = '';
        for ($i = 0; $i < strlen($str); $i += 2) {
            $decrypt .= chr(hexdec(substr($str, $i, 2)));
        }
        $decrypt = mcrypt_decrypt(MCRYPT_RIJNDAEL_128, $key,
$decrypt, MCRYPT_MODE_CBC, $iv);
        $str = '';
        $decrypt = str_split($decrypt);
        for ($i = 0; $i < count($decrypt); $i++) {
            ord($decrypt[$i]) === 0 or $str .= $decrypt[$i];
        }

        return $str;
    }

    public static function encrypt($str, $key, $iv)
    {
        $encrypt = '';
        $encrypt = mcrypt_encrypt(MCRYPT_RIJNDAEL_128, $key, $str, MC
RYPT_MODE_CBC, $iv);
        $encrypt = bin2hex($encrypt);
        return $encrypt;
    }
}
```

3.2 Java 加解密

```
import javax.crypto.Cipher;
import javax.crypto.spec.IvParameterSpec;
import javax.crypto.spec.SecretKeySpec;

public class AES
{
    public static void main(String args[]) throws Exception {
        /**
         * 加密用的Key 可以用26个字母和数字组成，最好不要用保留字符，
         * 虽然不会错，至于怎么裁决，个人看情况而定
         *
         * key == AppSecret
         */
        String cKey = "1234567890123456";
        // 需要加密的字串
        String cSrc = "{\"code\":\"0\",\"error_msg\":\"密码错误\",\"we
ixiao_openid\":\"12345678\",\"student_num\":\"888888888888\",\"name
\":\"洪丹丹测试\",\"sign\":\"5C6E844C23C8F0C15AF382081D0663DC\"}";
        // MD5(SHCOOL_ID) 取前16位;
        String cIv = "0123456789123456";
        System.out.println(cSrc);
        // 加密
        long lStart = System.currentTimeMillis();
        String enString = AES.Encrypt(cSrc, cKey, cIv);
        System.out.println("加密后的字串是：" + enString);

        long lUseTime = System.currentTimeMillis() - lStart;
        System.out.println("加密耗时：" + lUseTime + "毫秒");
        // 解密
        lStart = System.currentTimeMillis();
        String DeString = AES.Decrypt(enString, cKey, cIv);
        System.out.println("解密后的字串是：" + DeString);
        lUseTime = System.currentTimeMillis() - lStart;
        System.out.println("解密耗时：" + lUseTime + "毫秒");

    }

    public static String Encrypt(String sSrc, String sKey, String sI
v) throws Exception {

        Cipher cipher = Cipher.getInstance("AES/CBC/NoPadding");
        int blockSize = cipher.getBlockSize();

        byte[] dataBytes = sSrc.getBytes();
        int plaintextLength = dataBytes.length;
        if (plaintextLength % blockSize != 0) {
            plaintextLength = plaintextLength + (blockSize - (plainte
xtLength % blockSize));
```

```
        }

        byte[] plaintext = new byte[plaintextLength];
        System.arraycopy(dataBytes, 0, plaintext, 0, dataBytes.lengt
h);

        SecretKeySpec keyspec = new SecretKeySpec(sKey.getBytes(), "A
ES");
        IvParameterSpec ivspec = new IvParameterSpec(sIv.getBytes());

        cipher.init(Cipher.ENCRYPT_MODE, keyspec, ivspec);
        byte[] encrypted = cipher.doFinal(plaintext);

        return byte2hex(encrypted).toLowerCase();
    }

    public static String Decrypt(String sSrc, String sKey, String sI
v) throws Exception {

        byte[] encrypted1      = hex2byte(sSrc);

        Cipher cipher          = Cipher.getInstance("AES/CBC/NoPaddin
g");
        SecretKeySpec keyspec  = new SecretKeySpec(sKey.getBytes(), "AES");
        IvParameterSpec ivspec = new IvParameterSpec(sIv.getBytes());

        cipher.init(Cipher.DECRYPT_MODE, keyspec, ivspec);

        byte[] original = cipher.doFinal(encrypted1);
        String originalString = new String(original);

        return originalString;
    }

    public static byte[] hex2byte(String strhex) {
        if (strhex == null) {
            return null;
        }
        int l = strhex.length();
        if (l % 2 == 1) {
            return null;
        }
        byte[] b = new byte[l / 2];
        for (int i = 0; i != l / 2; i++) {
            b[i] = (byte) Integer.parseInt(strhex.substring(i * 2, i
 * 2 + 2),
                    16);
        }
```

```
            return b;
        }

        public static String byte2hex(byte[] b) {
            String hs = "";
            String stmp = "";
            for (int n = 0; n < b.length; n++) {
                stmp = (java.lang.Integer.toHexString(b[n] & 0XFF));
                if (stmp.length() == 1) {
                    hs = hs + "0" + stmp;
                } else {
                    hs = hs + stmp;
                }
            }

            return hs.toUpperCase();
        }
    }
```