Js 端加解密实现

一、Js 端生成一个长度为 16 的字符串作为 AESKey

```
<script type="text/javascript">
    var AESKey=getRandomNum(11111111,99999999)+'12345678';
    function getRandomNum(Min,Max)
        {
        var Range = Max - Min;
        var Rand = Math.random();
        return(Min + Math.round(Rand * Range));
        }
</script>
```

二、Js 使用 RSAPublic 公钥加密 AESKey

需要引入第三方的 js 包，下载地址：RSA and ECC in JavaScript

```
<script type="text/javascript" src="./js/rsa/jsbn.js"></script>
<script type="text/javascript" src="./js/rsa/prng4.js"></script>
<script type="text/javascript" src="./js/rsa/rng.js"></script>
<script type="text/javascript" src="./js/rsa/rsa.js"></script>
<script type="text/javascript" src="./js/rsa/base64.js"></script>
<script type="text/javascript">
        function rsaEncrypt(msg){
            var modulus = "E4210B0CB9C0FB75DBB263B8042161DDE8950
D125F56B63BE8DD0859AF5AE2E9E78701CA6C61A95F8AB6460D490E02B6C
285130635065336CF18A6D94F6E3F2C7641DBB99128643A6371845462F7C9
92698E41DBF9776468AE02AA7762D6645BCC16A70CFAF3ADCB630DF08BE2
7A5887268D19E780AAD73E52119093F3952D43";
            var e="10001";
            var rsa = new RSAKey();
            rsa.setPublic(modulus,e);
            var res0 = rsa.encrypt(msg);
            res = linebrk(hex2b64(res0),64);
            alert(res);
            return res;
        }
</script>
```

Js 代码中 modulus 及 e 的由来是根据 RSAPublic 公钥得来。E 是其中的 Exponent。

打印公钥信息：openssl rsa -pubin -in publickey.pem -text -modulus，如下：

ustc-zy@ustczy:~/js-workspace/node-learn/rsa/routes$ openssl rsa -pubin -in publickey.pem -text -modulus
Public-Key: (1024 bit)
Modulus:
    00:e4:21:0b:0c:b9:c0:fb:75:db:b2:63:b8:04:21:
    61:dd:e8:95:0d:12:5f:56:b6:3b:e8:dd:08:59:af:
    5a:e2:e9:e7:87:01:ca:6c:61:a9:5f:8a:b6:46:0d:
    49:0e:02:b6:c2:85:13:06:35:06:53:36:cf:18:a6:
    d9:4f:6e:3f:2c:76:41:db:b9:91:28:64:3a:63:71:
    84:54:62:f7:c9:92:69:8e:41:db:f9:77:64:68:ae:
    02:aa:77:62:d6:64:5b:cc:16:a7:0c:fa:f3:ad:cb:
    63:0d:f0:8b:e2:7a:58:87:26:8d:19:e7:80:aa:d7:
    3e:52:11:90:93:f3:95:2d:43
Exponent: 65537 (0x10001)submit
Modulus=E4210B0CB9C0FB75DBB263B8042161DDE8950D125F56B63BE8DD0859AF5AE2E9E78701CA6C61A95F8AB6460D490E02B6C285130635065336CF18A6D94F6E3F2C7641DBB99128643A6371845462F7C992698E41DBF9776468AE02AA7762D6645BCC16A70CFAF3ADCB630DF08BE27A5887268D19E780AAD73E52119093F3952D43
writing RSA key
-----BEGIN PUBLIC KEY-----
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDkIQsMucD7dduyY7gEIWHd6JUN
El9Wtjvo3QhZr1ri6eeHAcpsYalfirZGDUkOArbChRMGNQZTNs8YptlPbj8sdkHbuZEoZDpjcYRUYvfJkmmOQdv5d2RorgKqd2LWZFvMFqcM+vOty2MN8IvieliHJo0Z
54Cq1z5SEZCT85UtQwIDAQAB
-----END PUBLIC KEY-----

rsa 加密 AESKey 后需要进行转码，使用的第三方库加密后的格式为 hex，需要转为

base64，最后转为 string。

三、js 将加密后的发送给 android 端。

testtoolbar 项目中利用 js 调用 java 的方法将加密后的信息发送给 android 端。

Js 使用 AESKey 解密 android 发送过来的信息

Js 实现 aes 解密需要使用 google 的 crypto-js 库，中文 bootcdn(www.bootcdn.cn/cr

ypto-js/)站点可以获取。

```
<script type="text/javascript" src=".../rollups/aes.js"></script>
<script type="text/javascript" src=".../components/pad-zeropadding.js"></sc
ript>
<script type="text/javascript">
    var AESKey=getRandomNum(11111111,99999999)+'12345678';
    function aesDecrypt(msg){
        var key = CryptoJS.enc.Utf8.parse(AESKey);
        var iv  = CryptoJS.enc.Utf8.parse(AESKey);
        var decrypted = CryptoJS.AES.decrypt(msg,key,{iv:iv,
            padding:CryptoJS.pad.ZeroPadding
        });
        return decrypted.toString(CryptoJS.enc.Utf8);
    }
</script>
```

需要注意的是传递给 aesDecrypt 的 msg 是经过 Base64 编码的，即在 android 中：

```
byte[] encrypted = ....
msg=Base64.encodeToString(encrypted,Base64.DEFAULT)
```

Js 使用 AESKey 加密消息

```
<script type="text/javascript" src=".../rollups/aes.js"></script>
<script type="text/javascript" src=".../components/pad-zeropadding.js"></sc
ript>
<script type="text/javascript">
    var AESKey=getRandomNum(11111111,99999999)+'12345678';
    function aesEncrypt(msg){
        var key = CryptoJS.enc.Utf8.parse(tempKey);
        var iv  = CryptoJS.enc.Utf8.parse(tempKey);
        var encrypted = CryptoJS.AES.encrypt(msg, key, { iv: iv,mode:Crypt
oJS.mode.CBC,padding:CryptoJS.pad.ZeroPadding});
            var result = encrypted.toString();
            window.jsObject.communicate(result);
    }
</script>
```

aes 加密后密文的编码格式为 base64，所以直接转为 string，传给 Android 端，android

端利用自身的工具类(android.util.)Base64 对密文解码为 byte[]:

```
byte[] enc = Base64.decode(ciphertext, Base64.DEFAULT);
```