网易云音乐登录

加密算法

核心过程如下：

```
text = {
        'username': username,
        'password': password,
        'rememberLogin': 'true'
    }
    text = json.dumps(text)
    secKey = createSecretKey(16)
    encText = aesEncrypt(aesEncrypt(text, nonce), secKey)
    encSecKey = rsaEncrypt(secKey, pubKey, modulus)
    data = {
        'params': encText,
        'encSecKey': encSecKey
    }
```

其中`modulus`、`nonce`、`pubKey`均为已知，算法先通过`createSecretKey`生成一个16位的随机字符串作为密钥`secKey`，然后将明文`text`进行两次`AES`加密获得密文`encText`,因为`secKey`是在客户端上生成的，所以还需要对其进行`RSA`加密再传给服务端。

AES加密部分

AES加密的具体算法为: `AES-128-CBC`，输出格式为 `base64`
AES加密时需要指定 iv： `0102030405060708`
AES加密时需要 padding

```
def aesEncrypt(text, secKey):
    pad = 16 - len(text) % 16
    text = text + pad * chr(pad)
    encryptor = AES.new(secKey, 2, '0102030405060708')
    ciphertext = encryptor.encrypt(text)
    ciphertext = base64.b64encode(ciphertext)
    return ciphertext
```

这里使用了`pycrypto`，相比`nodejs`的`crypto`模块来说，需要手动做`padding`参考 [stackoverflow]

RSA 加密部分

这样加密出来的密文有个特点：加密过程没有随机因素，明文多次加密后得到的密文是相同的
然而，我们常用的 RSA 加密模块均不支持此种加密，所以需要手写一段简单的 RSA 加密
加密过程 `convertUtf8toHex(reversedText)^e%N`
输入过程中需要对加密字符串进行 hex 格式转码
RSA 加密采用非常规填充方式，既不是`PKCS1`也不是`PKCS1_OAEP`，网易的做法是直接向前补`0`

```python
def rsaEncrypt(text, pubKey, modulus):
    text = text[::-1]
    rs = int(text.encode('hex'), 16)**int(pubKey, 16)%int(modulus, 16)
    return format(rs, 'x').zfill(256)
```

附录

需要的 `modulus` 、 `nonce` 、 `pubKey`

```
modulus =
'00e0b509f6259df8642dbc35662901477df22677ec152b5ff68ace615bb7b725152b3ab17a876aea8a5aa76d2e417629e
c4ee341f56135fccf695280104e0312ecbda92557c93870114af6c9d05c4f7f0c3685b7a46bee255932575cce10b424d81
3cfe4875d3e82047b97ddef52741d546b8e289dc6935b3ece0462db0a22b8e7'
nonce = '0CoJUm6Qyw8W8jud'
pubKey = '010001'
```

Python 随机数生成

```python
def createSecretKey(size):
    return (''.join(map(lambda xx: (hex(ord(xx))[2:]), os.urandom(size))))[0:16]
```