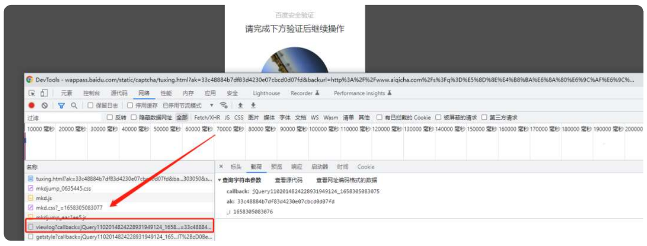
2023百度旋转验证码纯python逆向代码完结

原创 三哥a 何三笔记 2023-01-05 08:20 发表于河北

收录于合集
#python 70 #脚本 2

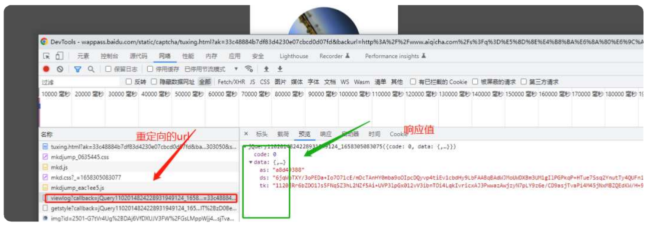
本文通过chrome浏览器分析百度验证码整个逻辑，然后通过python代码实现逆向打码过程

查看重定向的url,获取as, ds, tk 三个值



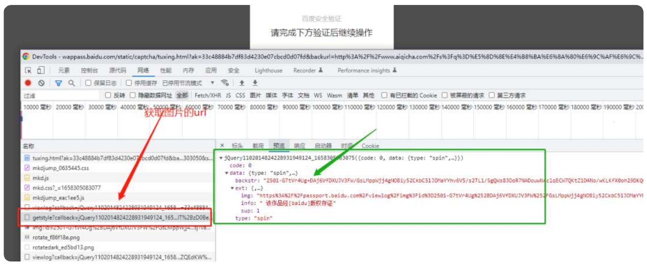
```
DevTools - wappass.baidu.com/static/captcha/tuxing.html?ak=33c48884b7df83d4230e07cbcd0d07fd&backurl=http%3A%2F%2Fwww.aiqicha.com%2Fs%3Fq%3D%E5%8D%8E%E4%B8%BA%E6%8A%80%E6%9C%AF%E6%9C%...

▾查询字符串参数
  callback: jQuery110201482422893194912 4_1658305083075
  ak: 33c48884b7df83d4230e07cbcd0d07fd
  _: 1658305083076
```



```
jQuery110201482422893194912 4_1658305083075({code: 0, data: {,…}})
  code: 0
  ▾data: {,…}
    as: "a8d49388"
    ds: "6jqWbTXY/3oPEDa+Io7O71cE/mDcTAnHY0mba9oOIpcDQyvp4tiEv1cbdHy9LbFAABqBAdWJMoUWDXBm3UM1gIlPGPkqP+HTue7Ssq2YnutTy4QUFn1
    tk: "1120fRr6bZDO1Js5FNqSZ3hL2NZf5Ai+UVP3lpGx0l2vV3ibnTOi4LqkIvricxAJ3PwwazAwjzyN7pLY9z6e/CD9asjTvaPi4M45jNxMBZQEdKW/H+9
```

获取旋转的原图 和 backstr 的值

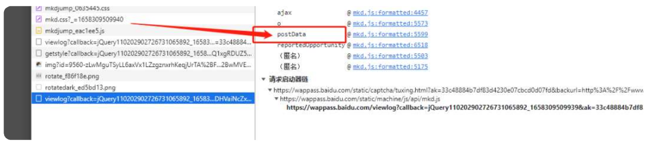


DevTools - wappass.baidu.com/static/captcha/tuxing.html?ak=33c48884b7df83d4230e07cbcd0d07fd&backurl=http%3A%2F%2Fwww.aiqicha.com%2Fs%3Fq%3D%E5%8D%8E%E4%B8%BA%E6%8A%80%E6%9C%AF%E6%9C%...

```
tuxing.html?ak=33c48884b7df83d4230e07cbcd0d07fd&ba...303050&s...
mkdjump_0635445.css
mkd.js
mkd.css?_=1658305083077
mkdjump_eac1ee5.js
getstyle?callback=jQuery1102014824228931949124_165...IT%28zD08e...
rotate_f86f18e.png
rotatedark_ed5bd13.png
viewlog?callback=jQuery1102014824228931949124_1658...ZQEdKW%...
```

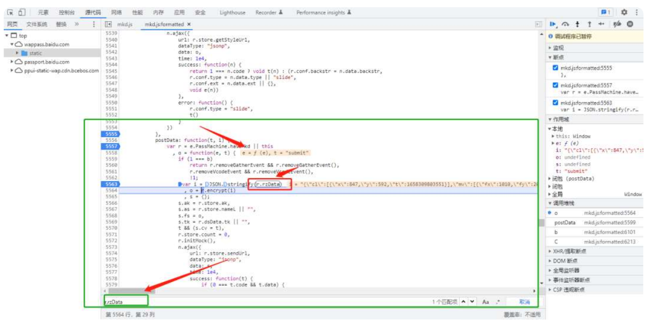
```
▼ jQuery1102014824228931949124_1658305083075({code: 0, data: {type: "spin",…}})
    code: 0
  ▼ data: {type: "spin",…}
      backstr: "2501-G7tVr4Ug+DAj6VfDXUJV3FW/GsLMppWjj4gNO8iy52CkbC51JOMaYYhv6V5/s2TL1/SgQWs8JOoR7NADouwN…clqECH7QKtZ1D4No/wKLKfX0oh29DKQ
    ▼ ext: {,…}
        img: "https%3A%2F%2Fpassport.baidu.com%2Fviewlog%2Fimg%3Fid%3D2501-G7tVr4Ug%252BDAj6VfDXUJV3FW%252FGsLMppWjj4gNO8iy52CkbC51JOMaYY
        info: " 该作品经[baidu]版权存证"
        sup: 1
      type: "spin"
```

旋转验证参数提交，dbug 调试参数跟进, 跟进来后发现重点在 r.rzData



```
mkdjump_0635445.css
mkd.css?_=1658309509940
mkdjump_eac1ee5.js
viewlog?callback=jQuery110202902726731065892_16583...=33c48884...
getstyle?callback=jQuery110202902726731065892_1658...Q1xgRDUZ5...
img?id=9560-zLwMguTSyLL6axVx1LZzgznxrhKeqjUrTA%2BF...2BwMVE...
rotate_f86f18e.png
rotatedark_ed5bd13.png
viewlog?callback=jQuery110202902726731065892_16583...DHVaiNcZx...
```

```
ajax                  @ mkd.js:formatted:4457
o                     @ mkd.js:formatted:5573
postData              @ mkd.js:formatted:5599
reportedOpportunity   @ mkd.js:formatted:6518
（匿名）               @ mkd.js:formatted:5503
（匿名）               @ mkd.js:formatted:5175
▼ 请求启动器链
  ▼ https://wappass.baidu.com/static/captcha/tuxing.html?ak=33c48884b7df83d4230e07cbcd0d07fd&backurl=http%3A%2F%2Fwww
    ▼ https://wappass.baidu.com/static/machine/js/api/mkd.js
        https://wappass.baidu.com/viewlog?callback=jQuery110202902726731065892_1658309509939&ak=33c48884b7df8
```

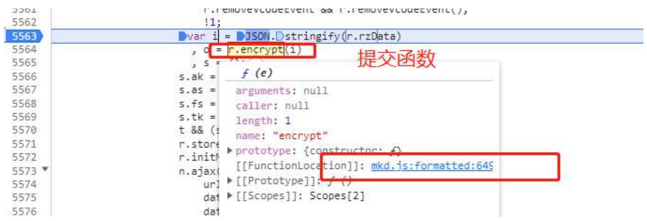


```
                n.ajax({
                    url: r.store.getStyleUrl,
                    dataType: "jsonp",
                    data: o,
                    time: 1e4,
                    success: function(n) {
                        return 1 === n.code ? void t(n) : (r.conf.backstr = n.data.backstr,
                        r.conf.type = n.data.type || "slide",
                        r.conf.ext = n.data.ext || {},
                        void e(n))
                    },
                    error: function() {
                        r.conf.type = "slide",
                        t()
                    }
                })
            },
            postData: function(t, i) {
                var r = e.PassMachine.hasMkd || this
                  , o = function(e, t) {  e = ƒ (e), t = "submit"
                    if (1 === b)
                        return r.removeGatherEvent && r.removeGatherEvent(),
                        r.removeVcodeEvent && r.removeVcodeEvent(),
                        !1;
                    var i = JSON.stringify(r.rzData)  i = "{\"cl\":[{\"x\":847,\"y\":592,\"t\":1658309803551}],\"mv\":[{\"fx\":1010,\"fy\":2
                      , o = r.encrypt(i)
                      , s = {};
                    s.ak = r.store.ak,
                    s.as = r.store.nameL || "",
                    s.fs = o,
                    s.tk = r.dsData.tk || "",
                    t && (s.cv = t),
                    r.store.count = 0,
                    r.initMock(),
                    n.ajax({
                        url: r.store.sendUrl,
                        dataType: "jsonp",
                        data: s,
                        time: 1e4,
                        success: function(t) {
                            if (0 === t.code && t.data) {
```

r.rzData　1 个匹配项　Aa　取消

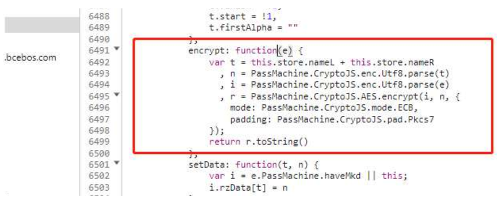
第 5564 行，第 29 列　覆盖率: 不适用

控制台打印 r.raData, 点击进入函数

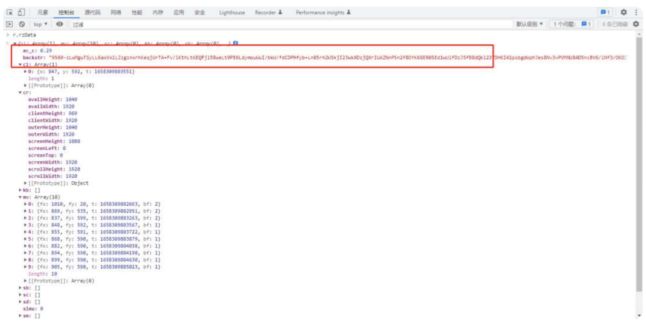


```
            r.removeVcodeEvent && r.removeVcodeEvent(),
            !1;
        var i = JSON.stringify(r.rzData)
          , o = r.encrypt(i)
          , s =
        s.ak =
        s.as =
        s.fs =
        s.tk =
        t && (s
        r.store
        r.initM
        n.ajax(
            url
            dat
            dat
```

```
ƒ (e)
 arguments: null
 caller: null
 length: 1
 name: "encrypt"
▸prototype: {constructor: ƒ}
 [[FunctionLocation]]: mkd.js:formatted:649
▸[[Prototype]]: ƒ ()
▸[[Scopes]]: Scopes[2]
```

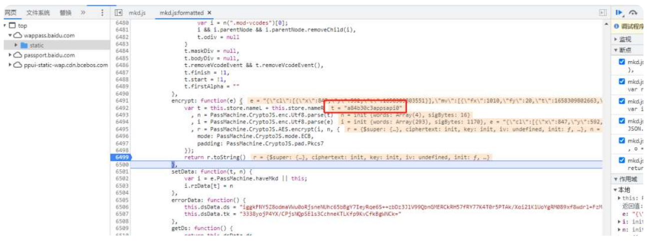


```
            t.start = !1,
            t.firstAlpha = ""
        },
        encrypt: function(e) {
            var t = this.store.nameL + this.store.nameR
              , n = PassMachine.CryptoJS.enc.Utf8.parse(t)
              , i = PassMachine.CryptoJS.enc.Utf8.parse(e)
              , r = PassMachine.CryptoJS.AES.encrypt(i, n, {
                mode: PassMachine.CryptoJS.mode.ECB,
                padding: PassMachine.CryptoJS.pad.Pkcs7
            });
            return r.toString()
        },
        setData: function(t, n) {
            var i = e.PassMachine.haveMkd || this;
            i.rzData[t] = n
```



```
> r.rzData
< ▾{cl: Array(1), mv: Array(10), sc: Array(0), kb: Array(0), sb: Array(0), …}
    ac_c: 0.29
    backstr: "9560-zLwMguTSyLL6axVx1LZzgznxrhKeqjUrTA+Fv/1KthLtKEQPj158weLt9PE6LdymouWuI/bkW/fdCDPHfyb+Ln05rh2W5kjI23wkXD
  ▾cl: Array(1)
    ▸0: {x: 847, y: 592, t: 1658309803551}
     length: 1
    ▸[[Prototype]]: Array(0)
  ▾cr:
     availHeight: 1040
     availWidth: 1920
     clientHeight: 969
     clientWidth: 1920
     outerHeight: 1040
     outerWidth: 1920
     screenHeight: 1080
     screenLeft: 0
     screenTop: 0
     screenWidth: 1920
     scrollHeight: 1920
     scrollWidth: 1920
    ▸[[Prototype]]: Object
  ▸kb: []
  ▾mv: Array(10)
    ▸0: {fx: 1010, fy: 20, t: 1658309802663, bf: 2}
    ▸1: {fx: 869, fy: 535, t: 1658309802951, bf: 2}
    ▸2: {fx: 837, fy: 599, t: 1658309803263, bf: 2}
    ▸3: {fx: 848, fy: 592, t: 1658309803567, bf: 1}
    ▸4: {fx: 855, fy: 591, t: 1658309803722, bf: 1}
    ▸5: {fx: 868, fy: 590, t: 1658309803879, bf: 1}
    ▸6: {fx: 882, fy: 590, t: 1658309804038, bf: 1}
    ▸7: {fx: 894, fy: 590, t: 1658309804190, bf: 1}
    ▸8: {fx: 899, fy: 590, t: 1658309804630, bf: 1}
    ▸9: {fx: 905, fy: 590, t: 1658309805023, bf: 1}
     length: 10
    ▸[[Prototype]]: Array(0)
  ▸sb: []
  ▸sc: []
  ▸sd: []
   simu: 0
  ▸sm: []
```

在r.rzData中ac_c是检测的关键，ac_c=round((o / 212),2)，而o是滑动的距离，o=angle*212/360 （angle）是识别的角度。然后backstr是前面返回的，其他的所有参数都可固定，包括轨迹fs是对r.rzData进行aes加密的结果（key是ac+'appsapi0'）



```
                var i = n(".mod-vcodes")[0];
                i && i.parentNode && i.parentNode.removeChild(i),
                t.odiv = null
            }
            t.maskDiv = null,
            t.bodyDiv = null,
            t.removeVcodeEvent && t.removeVcodeEvent(),
            t.finish = !1,
            t.start = !1,
            t.firstAlpha = ""
        },
        encrypt: function(e) {  e = "{\"cl\":[{\"x\":847,\"y\":592,\"t\":1658309803551}],\"mv\":[{\"fx\":1010,\"fy\":20,\"t\":1658309802663,\
            var t = this.store.nameL + this.store.nameR  t = "a84b30c3appsapi0"
              , n = PassMachine.CryptoJS.enc.Utf8.parse(t)  n = init {words: Array(4), sigBytes: 16}
              , i = PassMachine.CryptoJS.enc.Utf8.parse(e)  i = init {words: Array(293), sigBytes: 1170}, e = "{\"cl\":[{\"x\":847,\"y\":592,
              , r = PassMachine.CryptoJS.AES.encrypt(i, n, {  r = {$super: {…}, ciphertext: init, key: init, iv: undefined, init: ƒ, …}, n =
                mode: PassMachine.CryptoJS.mode.ECB,
                padding: PassMachine.CryptoJS.pad.Pkcs7
            });
            return r.toString()  r = {$super: {…}, ciphertext: init, key: init, iv: undefined, init: ƒ, …}
        },
        setData: function(t, n) {
            var i = e.PassMachine.haveMkd || this;
            i.rzData[t] = n
        },
        errorData: function() {
            this.dsData.ds = "iggkFNY5Z8odmaVWu0oRjsneNUhc65bBgY7IeyRqe6S++zbDz3J1V99QbnGMERCkRH57fRY77K4T0r5PTAk/Xoi21K1UoYgRM089xf8wdr1+FzM
            this.dsData.tk = "3338yojP4YX/CPjsNQpSE1s3CchneKTLKfp9KvCfkBgWNCk="
        },
        getDs: function() {
```

完整python代码如下：

```python
from urllib.parse import unquote
import requests
import time
import json
import re
import urllib3
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)
import base64
from Crypto.Cipher import AES
import urllib

BLOCK_SIZE = 16  # Bytes
pad = lambda s: s + (BLOCK_SIZE - len(s) % BLOCK_SIZE) * \
                chr(BLOCK_SIZE - len(s) % BLOCK_SIZE)
unpad = lambda s: s[:-ord(s[len(s) - 1:])]
def aesEncrypt(key, data):
    '''
    AES的ECB模式加密方法
    :param key: 密钥
    :param data:被加密字符串（明文）
    :return:密文
    '''
    key = key.encode('utf8')
    # 字符串补位
    data = pad(data)
    cipher = AES.new(key, AES.MODE_ECB)
    # 加密后得到的是bytes类型的数据，使用Base64进行编码,返回byte字符串
    result = cipher.encrypt(data.encode())
    encodestrs = base64.b64encode(result)
    enctext = encodestrs.decode('utf8')
    # print(enctext)
    return enctext

def parseCookiestr(cookie_str):
    """解析cookie"""
    cookielist = []
    for item in cookie_str.split(';'):
        try:
            cookie={}
            itemname=item.split('=')[0]
            iremvalue=item.split('=')[1]
            cookie['name']=itemname
            cookie['value']=urllib.parse.unquote(iremvalue)
            cookielist.append(cookie)
        except:
            pass
    return cookielist

def dama_api(img_b64):
    token = '打码token' #打码接口非常便宜，有需要请联系 v：466867714
    api = f'http://api.h3blog.com/yzm_api/xuanzhuan/{token}/b64'
    result = requests.post(api,data={"img":img_b64}).text
    print('打码返回数据：', result)
    result = json.loads(result)
    return result['data']


class BaiduAiqichaRotate:
    def __init__(self):
        self.session = requests.session()
        self.headers = {
            'Accept': 'text/javascript, application/javascript,
application/ecmascript, application/x-ecmascript, */*; q=0.01',
            'Accept-Language': 'zh-CN,zh;q=0.9',
            'Connection': 'keep-alive',
            'Sec-Fetch-Dest': 'empty',
            'Sec-Fetch-Mode': 'cors',
            'Sec-Fetch-Site': 'same-origin',
            'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64)
AppleWebKit/537.36 (KHTML, like Gecko) '
                          'Chrome/103.0.0.0 Safari/537.36',
            'X-Requested-With': 'XMLHttpRequest',
            'Referer': 'https://ziyuan.baidu.com/',
            'sec-ch-ua': '".Not/A)Brand";v="99", "Google Chrome";v="103",
"Chromium";v="103"',
            'sec-ch-ua-mobile': '?0',
            'sec-ch-ua-platform': '"Windows"',
        }
        cookie = requests.cookies.RequestsCookieJar()
        cookies = '百度cookie'
        cookielist  = parseCookiestr(cookies)
        for item in cookielist:
            cookie.set(item['name'], item['value'])
        self.session.cookies.update(cookie)

    def get_image_request_data(self):
        """
        :return: 获取需要获取图片的参数
        """
        url = "https://passport.baidu.com/viewlog"
        params = {
            "callback": "jQuery110205449684422426735_" +
str(int(time.time() * 1000)),
            "ak": "33c48884b7df83d4230e07cbcd0d07fd",
            "_": str(int(time.time() * 1000))
        }
        response = self.session.get(url, headers=self.headers,
params=params)
        res_data = re.findall(r'.*?(\{.*?})\)', response.text)[0]
        res_data = json.loads(res_data)
        item = {
            "tk": res_data['data']['tk'],
            "as": res_data['data']['as'],
            "ds": res_data['data']['ds']
        }
        return item

    def get_img(self, item):
        url = "https://passport.baidu.com/viewlog/getstyle"
        params = {
            "callback": "jQuery110205449684422426735_" + str(time.time()
* 1000),
            "ak": '3de47787fd60b30420f868ffbf4dbccd',
            "tk": item["tk"],
            "isios": "0",
            "type": "spin",
            "_": str(time.time() * 1000)
```

```python
        }
        response = self.session.get(url, headers=self.headers,
params=params)
        ret_data = re.findall(r'.*?(\{.*?})\)', response.text)[0]
        ret_data = json.loads(ret_data)
        item_img = {
            "img_url": unquote(ret_data['data']['ext']['img']),
            "backstr": ret_data['data']['backstr'],
            "tk": item["tk"],
            "as": item["as"]
        }
        response = self.session.get(item_img['img_url'], verify=False)
        with open('img.png', 'wb')as f:
            f.write(response.content)
        return item_img

    def build_fs(self, angle,ass,backstr)->str:
        tt = {
            "cl": [
                {
                    "x": 862,
                    "y": 287,
                    "t": 1657760616916
                }
            ],
            "mv": [
                {
                    "fx": 987,
                    "fy": 149,
                    "t": 1657760613905,
                    "bf": 2
                },
                {
                    "fx": 979,
                    "fy": 370,
                    "t": 1657760615529,
                    "bf": 2
                },
                {
                    "fx": 948,
                "fy": 339,
                "t": 1657760615688,
                "bf": 2
            },
            {
                "fx": 911,
                "fy": 321,
                "t": 1657760615848,
                "bf": 2
            },
            {
                "fx": 892,
                "fy": 309,
                "t": 1657760616008,
                "bf": 2
            },
            {
                "fx": 880,
                "fy": 299,
                "t": 1657760616176,
                "bf": 2
            },
            {
                "fx": 869,
                "fy": 290,
                "t": 1657760616440,
                "bf": 2
            },
            {
                "fx": 864,
                "fy": 288,
                "t": 1657760616641,
                "bf": 2
            },
            {
                "fx": 862,
                "fy": 287,
                "t": 1657760616866,
                "bf": 2
            },
            {
                "fx": 864,
                "fy": 288,
                "t": 1657760617026,
                "bf": 1
            },
            {
                "fx": 877,
                "fy": 293,
                "t": 1657760617186,
                "bf": 1
            },
            {
                "fx": 882,
                "fy": 295,
                "t": 1657760617360,
                "bf": 1
            },
            {
                "fx": 891,
                "fy": 298,
                "t": 1657760617537,
                "bf": 1
            },
            {
                "fx": 900,
                "fy": 300,
                "t": 1657760617688,
                "bf": 1
            },
            {
                "fx": 908,
                "fy": 301,
                "t": 1657760617864,
                "bf": 1
            },
            {
                "fx": 910,
                "fy": 301,
                "t": 1657760618585,
                    "bf": 1
                }
            ],
            "sc": [],
            "kb": [
                {
                    "key": "a",
                    "t": 1657760606047
                }
            ],
            "sb": [],
            "sd": [],
            "sm": [],
            "cr": {
                "screenTop": 0,
                "screenLeft": 0,
                "clientWidth": 1920,
                "clientHeight": 979,
                "screenWidth": 1920,
                "screenHeight": 1080,
                "availWidth": 1920,
                "availHeight": 1050,
                "outerWidth": 1920,
                "outerHeight": 1050,
                "scrollWidth": 1920,
                "scrollHeight": 1920
            },
            "simu": 0,
            "ac_c": round((angle * 212 / 360 / 212),2),
            "backstr": backstr
        }

        tt = json.dumps(tt)
        # print(tt)
        # print("key = ", ass + 'appsapi0')
        return aesEncrypt(ass+'appsapi0',tt)

    def verify_data(self, item):
        url = "https://passport.baidu.com/viewlog"
        print("angle: ", item['angle'])
```

```python
        print("as: ", item['as'])
        # with open('get_encrypt.js', 'r', encoding='utf-8') as f:
        #     js_text = f.read()
        # fs = execjs.compile(js_text).call('encrypt_',
str(item['angle']), str(item['as']), str(item['backstr']))
        # print("fs: ", fs)
        fs = self.build_fs(int(item['angle']), item['as'],
item['backstr'])

        params = {
            "callback": "jQuery110204100787474351779_" + str(time.time()
* 1000),
            "ak": "3de47787fd60b30420f868ffbf4dbccd",
            "as": item['as'],
            "fs": fs,
            "tk": item['tk'],
            "cv": "submit",
            "_": str(time.time() * 1000)
        }
        response = self.session.get(url, headers=self.headers,
params=params)
        ret_data = re.findall(r'.*?(\{.*?})\)', response.text)[0]
        ret_data = json.loads(ret_data)
        # print("验证结果: ", ret_data)
        return ret_data




if __name__ == '__main__':
    bs = BaiduAiqichaRotate()
    item = bs.get_image_request_data()
    item_img = bs.get_img(item)
    with open('img.png',mode='rb') as f :
        content = f.read()
    base64_data = base64.b64encode(content)
    angle = dama_api(base64_data)
    item_img['angle'] = angle
    # print(item_img)
    ret_data = bs.verify_data(item_img)
```

```
        if 1 == ret_data['data']['op']:
            print("验证通过")
```

更多干活，请关注“何三笔记”公众号

收录于合集 #python 70

下一篇 · Python：AES加密与解密（ECB模式）详解

阅读原文