

AWFUL

神仙打架周。。。

Misc

分数 200: ngc's wifi

描述: 破解 ngc 的路由器

hint1:怎么那么多人用手机号来做路由器密码呢

hint2:潍坊是吧

ps:获得密码后加上 hgame{}即可

更新一下真的 url: <http://p1kaloi2x.bkt.clouddn.com/hgame/cap/flag2.cap>

URL: <http://p1kaloi2x.bkt.clouddn.com/hgame/cap/flag2.cap>

下载得到一个流量包,通过后缀名可以确定是为加密包(也可以根据题目描述或者打开流量包查看特征来推断)。根据 hint 可以推断出需要爆破,而且密码为潍坊的手机号码。

查询手机号的信息得知,中间四位即第 4 位到第 7 位是地区编号,但是之后没有查到任何关于可靠的地区编号信息。

尝试自己写程序生成字典,但是由于太慢了最终放弃。

最后在网上找到全国手机号码生成器[1],打开后设置如下,位置随意



之后使用 aircrack 来破解密码,将之前生成的文件放在 aircrack 里面的 bin 目录下,然后在 cmd 端输入命令:

aircrack-ng flag2.cap -w zidian.txt (其中 zidian.txt 为字典的名字,生成器生成的文件并不是这个,但是为了方便将其名字改成这个)

```
E:\aircrack\bin>aircrack-ng flag2.cap -w zidian.txt
Opening flag2.cap
Read 109554 packets.

# BSSID ESSID Encryption
1 88:25:93:1E:99:D2 TP-LINK_99D2 WPA (0 handshake)
2 64:CC:2E:75:7D:27 ngc's wifi2 WPA (1 handshake)
3 A8:6B:7C:2F:BB:1B 360WiFi-111 WPA (0 handshake)
4 54:36:9B:00:46:85 lixin WPA (1 handshake)
5 7C:B5:40:0F:BB:B0 ChinaMobile_5055 WPA (0 handshake)
6 6F:B2:D6:E6:A4:DB Unknown
7 B4:49:16:72:B7:97 Unknown
8 A5:23:F9:82:56:56 Unknown

512 05:A0:27:67:B7:10
513 83:87:30:23:DA:3C

Index number of target network ?
```

它会询问数字,这里选择 2

```

eading packets, please wait...
Aircrack-ng 1.2 rc1

[00:10:03] 2069536 keys tested (3605.96 k/s)

KEY FOUND! [ 13375369512 ]

Master Key      : 9B 5C 3B 5B 3F 25 69 62 69 B5 BA 68 33 46 ED 67
                  FA F0 OD 16 9A B4 76 E4 A2 BB 11 B8 ED 68 77 E2

Transient Key   : 70 91 68 49 97 2C 1B 3D A9 FE 1C CF CC 6C 35 E8
                  FB 81 40 A0 10 5D 17 E7 D8 FC AF 21 B7 43 76 78
                  2C 3A D5 CB B4 27 81 C0 C8 61 D3 18 40 A7 D4 96
                  D5 85 76 05 F0 32 31 51 DD 3B 9D A6 2E 96 35 06

EAPOL HMAC     : B1 83 7E F7 06 7B 37 98 34 B6 3B 45 FC F2 D8 9D

```

最后结果可以看到密码，得到 flag

hgame{13375369512}

题外话：当初没仔细看题目，以为还需要解码，但是解出来的流量包没有什么内容，卡在这里一小会儿，最终错过了前三。

参考文章

[1] <https://www.52pojie.cn/thread-590846-1-1.html>

crypto

分数 200: ezECC

描述：我们来学习一下椭圆曲线吧

URL: <http://p3xlhyup6.bkt.clouddn.com/ecc>

下载文件的内容

```

1 p = 1026347883361447
2 a = 499590297305427
3 b = 30115568120981
4 G = (817367249716330, 483834901818242)
5 k = 622849
6 pub = (x, y)
7 flag = hgame{x + y} (数学意义上的加号)
8

```

既然都说是椭圆曲线，那就去学习呗。先给出文章[1]，得到以下结论：（摘抄自文章）

关于 p, a, b 的介绍

同时，并不是所有的椭圆曲线都适合加密。 $y^2=x^3+ax+b$ 是一类可以用来加密的椭圆曲线，也是最为简单的一类。下面我们就把 $y^2=x^3+ax+b$ 这条曲线定义在 F_p 上：

选择两个满足下列条件的小于 p (p 为素数)的非负整数 a 、 b

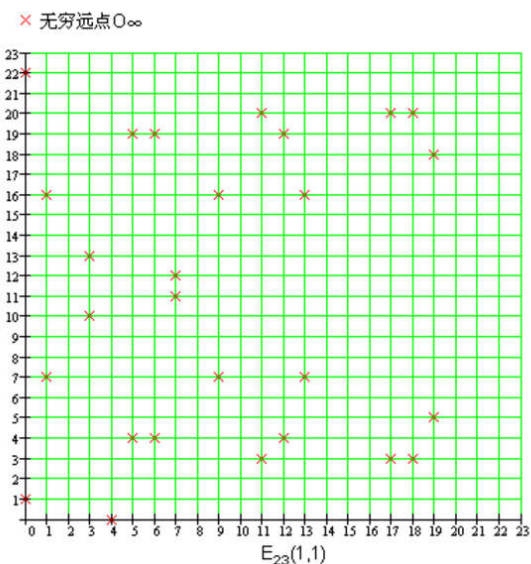
$$4a^3+27b^2 \not\equiv 0 \pmod{p}$$

则满足下列方程的所有点 (x, y) ，再加上 无穷远点 O_∞ ，构成一条椭圆曲线。

$$y^2=x^3+ax+b \pmod{p}$$

其中 x, y 属于0到 $p-1$ 间的整数，并将这条椭圆曲线记为 $E_p(a, b)$ 。

我们看一下 $y^2=x^3+x+1 \pmod{23}$ 的图像



G, k 的介绍以及加密过程：（注意里面的第 5 点和第 7 点，里面的加减乘除不是简单数学意义上的加减乘除，可以去网站上面了解更多）

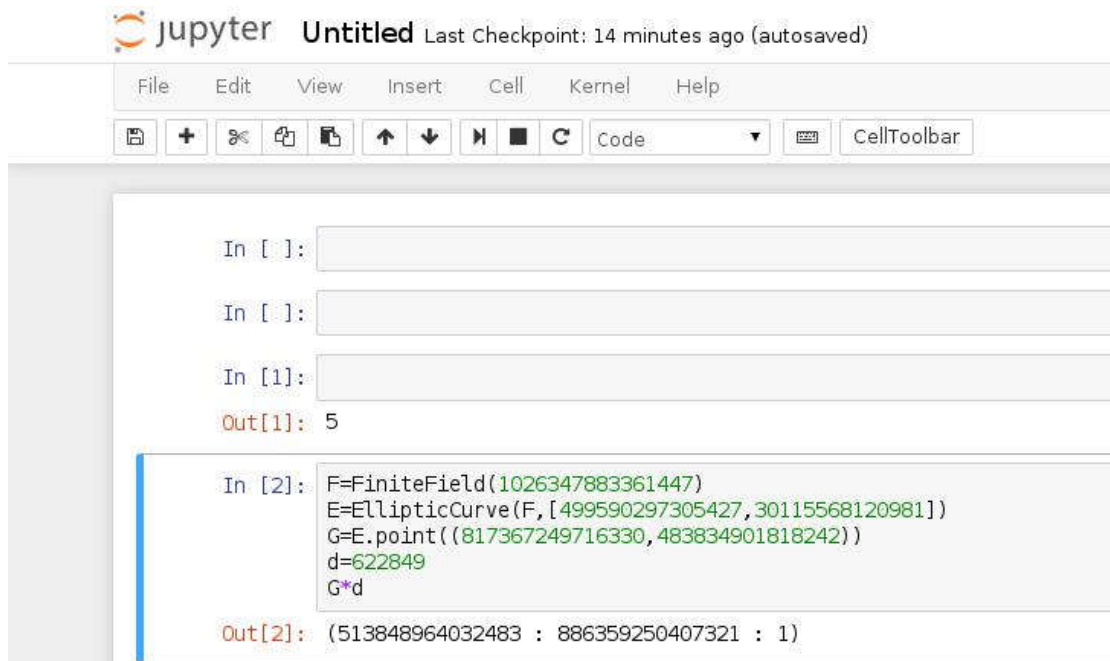
现在我们描述一个利用椭圆曲线进行加密通信的过程：

- 1、用户A选定一条椭圆曲线 $E_p(a, b)$ ，并取椭圆曲线上一点，作为基点 G 。
- 2、用户A选择一个私有密钥 k ，并生成公开密钥 $K=kG$ 。
- 3、用户A将 $E_p(a, b)$ 和点 K, G 传给用户B。
- 4、用户B接到信息后，将待传输的明文编码到 $E_p(a, b)$ 上一点 M （编码方法很多，这里不作讨论），并产生一个随机整数 r ($r < n$)。
- 5、用户B计算点 $C_1=M+rK$ ； $C_2=rG$ 。
- 6、用户B将 C_1 、 C_2 传给用户A。
- 7、用户A接到信息后，计算 C_1-kC_2 ，结果就是点 M 。因为

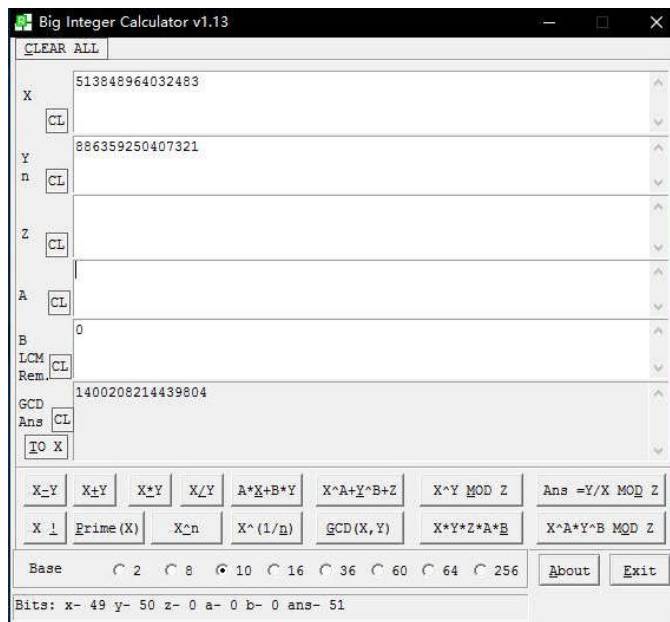
$$C_1-kC_2=M+rK-k(rG)=M+rK-r(kG)=M$$

再对点 M 进行解码就可以得到明文。

通过以上的摘要，那么只需要知道最后一个我们要求的東西即 $pub(x,y)$ 是什麼意義了，但是整篇文章並沒有提到過這個東西。在網上查詢，推測 pub 可能是公钥英文的縮寫，谷歌了一下得到一篇文章[2]，里面的 wp 提到需要用 $sagemath$ ，到官网下载 ova 文件[3]（虚拟机文件），打开系统输入命令可以看到结果



将前面两对数字提取出放到 big Integer Calculator 里面计算得到结果和 flag



hgame{1400208214439804}

参考文章:

- [1] <https://www.pediy.com/kssd/pediy06/pediy6014.htm>
- [2] <https://ehsande.com/pico2014/cryptography/ecc.html>
- [3] <http://sagemath.mirror.ac.za/win/index.html>

Misc

分数 300: so jb ez captcha

描述: 出题人啥也没说

URL: <http://115.159.33.58/sojbezcaptcha>

访问后为一个 json 文件, 可以理解为服务器发回来的消息的一种特殊格式。

JSON	原始数据	头
保存	复制	
flag:	null	
tips:	"please request with your token"	
status:	false	

JSON	原始数据	头
保存	复制	
status:	false	
tips:	"please request with your token using get method"	
flag:	null	

而服务器这边发送的文件有两种，如上图所示。从第二张图可以推测请求的目标就是这个服务器，方式是 GET，即往 URL 里面加入 token 参数。在 <http://115.159.33.58/sojbezcaptcha> 后面加入参数 token，里面的值随意

Load URL

Split URL

Execute

☐ Enable Post data

☐ Enable Referrer

JSON

原始数据

头

保存

复制

status:

true

captcha_png_base64:

"iVBORw0KGgoAAAANSUHEugAAAAAAAPCAyAAADd/140AAAA3U1EQVR4nJXSoYqEYBQF4DMyZpPBYLWI+g5iFYxisZisvoJFd/ENBIP3YBVBf0QEgyBn0zrManAP3HL4wrlwXySJB5GeoP9DIQTIOIamaVAUBa7rommaKwyCAPN8o6oq9H0Px3HgeR7GcfzItm0Jgf3X8TfhcVBVVRZfcXZv0zTx93BjkIDLmtZ1/ZS8SVmwBMC6rs/uaQdpq7rTJLka/+Cy7LQtM36vs993++hEIKWZTEHwis64TAMNayDaZryOI67tY1t22gYBqMougUnzPOcAG4ny7ITvshn3/MDubgohXPhcBQAAAAASUVORK5CYII="

tips:

"submit your answer with get parameter submit. if captcha is number 7 then request url?token=xxx&submit=7"

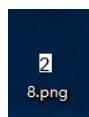
flag:

null


根据 captcha_png_base64 和 tip，推测 captcha_png_base64 里面的内容在进行 base64 解码之后为一个 png 图片的内容，为了证实这个想法，将 captcha_png_base64 里面的内容进行 base64 解码，放到一个空白文件中，然后以十六进制打开

Address	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f	Dump
00000000	89	50	4e	47	0d	0a	1a	0a	00	00	0d	49	48	44	52		PNG.....IHDR[
00000010	00	00	00	0a	00	00	00	0f	08	06	00	00	00	dd	ff	5e?^□
00000020	0e	00	00	00	dd	49	44	41	54	78	9c	95	d2	a1	8a	84軚DATx滑摇绍[
00000030	60	14	05	e0	33	32	66	93	c1	60	b5	88	fa	0e	62	15	`...?2f掄`祝?b.□□
00000040	8c	62	b1	98	ac	be	88	c5	77	f1	0d	04	83	c9	60	15	宐? 埃w?.兩`.□□
00000050	41	7f	44	04	ab	20	67	d3	3a	cc	6a	70	0f	dc	72	f8	A.D.?g?麤p.駝?□□
00000060	c2	bd	70	5f	24	89	07	91	9e	a0	ff	43	21	04	e2	38	諗n s2个脣 c1 2nr


可以看到 IHDR 字样，推测为 png 图片保存下来，可以看到验证码（例子中验证码为 2）




根据提示将验证码提交

 Load URL

http://115.159.33.58/sojbezcaptcha?token=98989898&submit=2

 Split URL

 Execute

☐ Enable Post data
 ☐ Enable Referrer

JSON

原始数据

头

保存

复制

```

status:    false
tips:     "timeout"
flag:     null
        
```

提示超时，重新试了几次都是这样，看来从发出验证码到提交的时间非常的短，那么写一个脚本来解决问题

```

# encoding=utf-8
import urllib
import urllib2
import re
from pyquery import PyQuery as pq
from lxml import etree
import os
import string
import StringIO
import cookielib
import random
import base64
from PIL import Image
import pytesseract

radomtxt = ""
hosturl = "http://115.159.33.58/sojbezcaptcha"
aspxsession = ""
base64txt = ""
code = ""
count = ""
response = ""
count1 = '1'

def init():
    global radomtoken
    #radomtoken = random.randint(1000000000,9999999999)
    #radomtoken = bytes(radomtoken)
    radomtoken='98989898' #经过确认服务器判定是否是同一个人连续成功提交验证码是根据 token 而不是 IP

def tryagain():
    global response
    try:
        response = urllib2.urlopen(hosturl+'?token='+radomtoken)
        #如果在获取服务器的验证码途中爆出 500 错误，则重新提交
    except:
        tryagain()

def tryagain1():
    global response
    try:
        response = urllib2.urlopen(hosturl+'?token='+radomtoken+"&submit="+code)
        #如果在提交验证码的途中爆出 500 错误，则重新提交
    except:

```

```

tryagain1()

def getbase64():
    global base64txt,response
    loginpage = "
    headers = {'User-Agent': 'Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like
    Gecko) Mobile/14D27 QQ/6.7.1.416 V1_IPH_SQ_6.7.1_1_APP_A Pixel/750 Core/UIWebView NetType/4G QBWebViewType/1',
               'Referer' : loginpage}

    tryagain()
    text = response.read()
    S = re.findall(r'base64": "(.+?)",',text)
    #获取 captcha_png_base64 里面的内容
    base64txt = S[0]
    #print base64txt

def img():
    imgdata=base64.b64decode(base64txt)
    file=open('8.png','wb')
    file.write(imgdata)
    file.close()
    #将内容 base64 解码后，写入这个脚本同一目录下的 8.png 文件保存。如果没有则生成一个新文件，如果有的则会覆盖

def getcode():
    global code
    code=pytesseract.image_to_string(Image.open('8.png'),lang = None,config='-psm 6', output_type="STRING")
    #通过图片识别里面验证码
    #print(text)

def postdata():
    #构造数据包
    global count
    loginpage = "
    headers = {'User-Agent': 'Mozilla/5.0 (iPhone; CPU iPhone OS 10_2_1 like Mac OS X) AppleWebKit/602.4.6 (KHTML, like
    Gecko) Mobile/14D27 QQ/6.7.1.416 V1_IPH_SQ_6.7.1_1_APP_A Pixel/750 Core/UIWebView NetType/4G QBWebViewType/1',
               'Referer' : loginpage}

    tryagain1()
    text = response.read()
    S = re.findall(r'status": (.+?)',text)
    text1 = S[0]
    print text1
    if text1=='true':
        S = re.findall(r'flag": (.+?)',text)
        text = S[0]

```

```

# print u'成功'

# print text
if 'null' not in text:
    print u'flag 为'
    print text
    count=1000000 # 停止发送

def start():
    global count
    count = 0
    while (count < 1000000):
        init()
        getbase64()
        img()
        getcode()
        postdata()
        count = count + 1
    raw_input(unicode('按回车键退出...', 'utf-8').encode('gbk'))

# 程序开始
if __name__ == '__main__':
    start()

```

运行得到 flag（少了一个"}”，手动加上



```

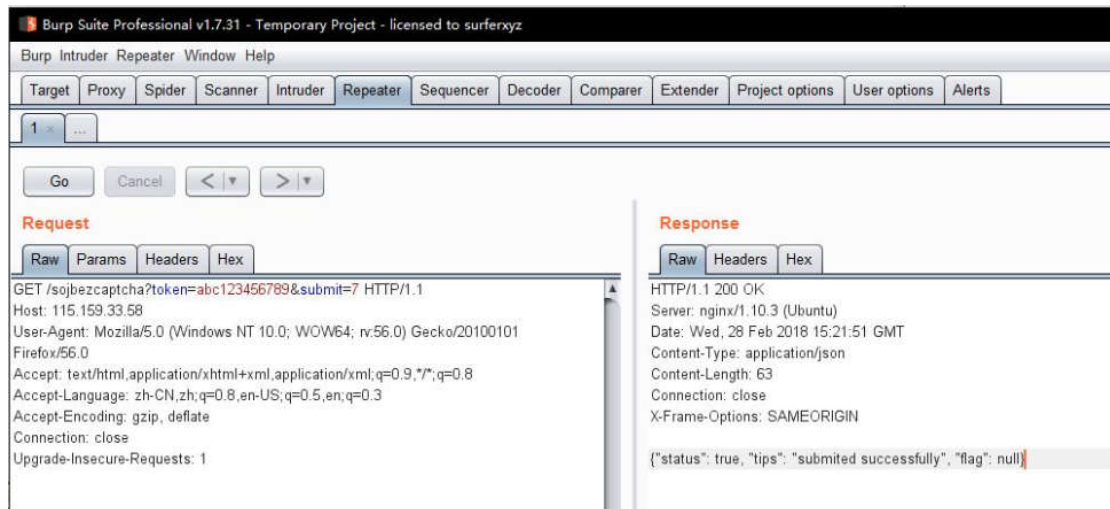
C:\Users\...\Desktop>python zuizhong.py
false
true
true
true
true
flag为
hgame {hammerissojbcute
按回车键退出...

```

过程（有点长，所以单独拉出讲）：

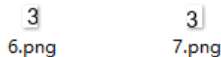
从未接触过 json 文件，当初用 360 浏览器访问的时候是提示是下载一个文件，刚开始从来没有想到 flag 是会藏在下载文件的那个网站上面，起初一看 tip，说用 token 请求，那么对什么请求，自己的 token 又是什么，怎么请求，基本上是那种完全没有思路的那种，后来用 firefox 访问后是直接将 json 文件显示出来而不是提示要下载 json 文件。询问大佬，得知 json 里面的内容可以理解为服务器返回的信息，由于解题途中多次放弃，每次都是重新下载 json 文件，突然发现好像下载的 json 文件貌似有不同的，通过刷新才知道有两种 json 文件也就是最上面的两张图。通过两种文件内容得知需要用 token 参数用 GET 方式提交，访问后得到 base64 编码，有了进展，从而确定目标。

在测试提交验证码过程中，可以看出目标服务器会动不动爆 500 的错误，但是多刷新几次就好了，而从接收到 base64 编码到提交验证码经过测试，大概在一秒左右之后就会爆出 time out，那么将 base64 解码放到文件保存，通过肉眼得到验证码再提交的方法显然时间是不够的，本着能不写脚本就不写的思想，验证码只有一位，第一想法是爆破，验证码只有 10 种，多试几次，验证码就可以蒙对，当初的方法如下，通过抓包软件，得到 base64 编码，马上提交固定验证码（比如 7），试了很多次最终返回一下结果



可以看到接下来就没有任何的提示了。卡了一段时间，无奈去询问出题人，得知服务器报 500 代码是正常的而且需要连续成功提交多次才能获得 flag。。。。

看来的一定要写脚本了，当初以为图片识别文字需要很高的技术，什么神经网络，深度学习啊什么的。当时的思路为 10 种验证码，如果只有 10 种图即 10 种 base64 编码的可能，那么就不要图片识别文字。但是经过测试发现：



可以看到即使是两个相同的验证码图片却是不同的，这个想法直接被打消掉了。。。

后来通过网上查询才知道由于题目给出的验证码实际上由于各种不同的因素，实际上识别起来非常简单[1]。由于最初配置 OCR 环境一直再报错而且还被某些网站的脚本坑了，弄了几乎一天。最后完成了脚本。（当时测试途中使用的是随机 token，但是一直没有结果，推测既然出题人说需要连续成功多次，那么怎么判断是同一个主机连续多次呢，可能是 IP 或者是 token，IP 的话随机 token 是不会有影响的，那么只可能是相同的 token，调整代码后的到 flag）

参考：

[1] http://blog.csdn.net/qiushi_1990/article/details/78041375

[2] http://download.csdn.net/download/qiushi_1990/9987023

[3]最后一个位置献给那位大佬