

WEB

ezHTTP

来自vidar.club、UA要求阿巴阿巴阿巴、来自本地（提示不是XFF，其他的都试一下发现是X-Real-IP）、最后解jwt得到flag



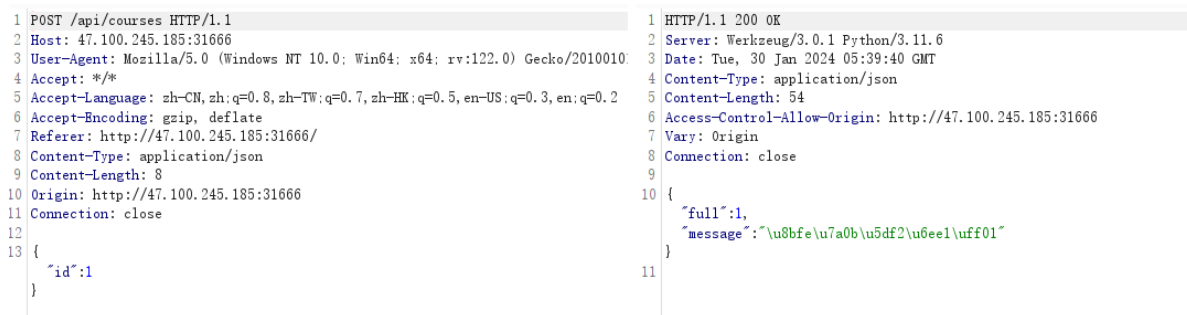
Bypass it

有个注册按钮但是无法跳转过去，查看一下注释发现里面有个JavaScript写的 `alert('很抱歉，当前不允许注册');``top.location.href='login.html'`

那么禁用一下浏览器的js注册一个账号，再开启js即可，火狐浏览器的方法为url框输入`about:config`搜索`javascript.enabled`，把`true`改成`false`即可

Select Courses

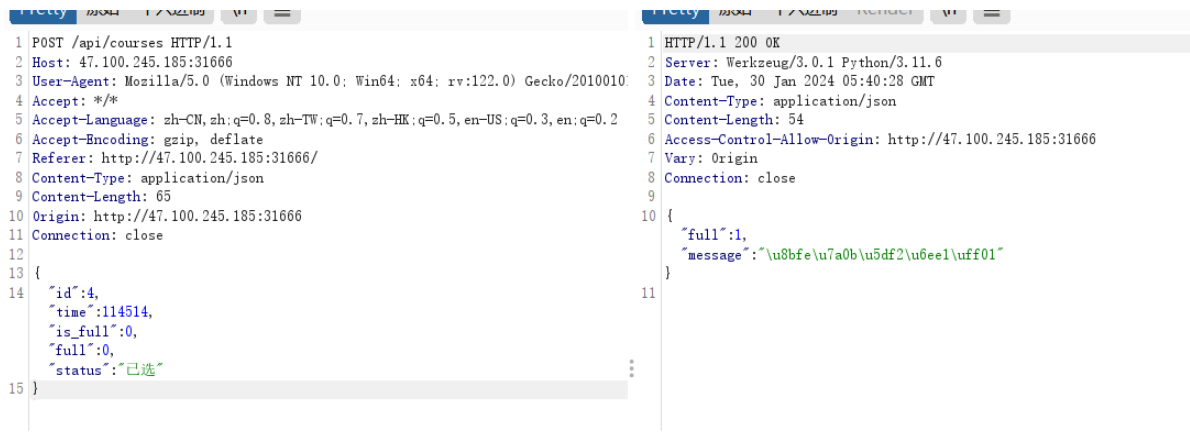
最迷惑的一集



然后想着里面有个full，源码里面还看到了is_full这些

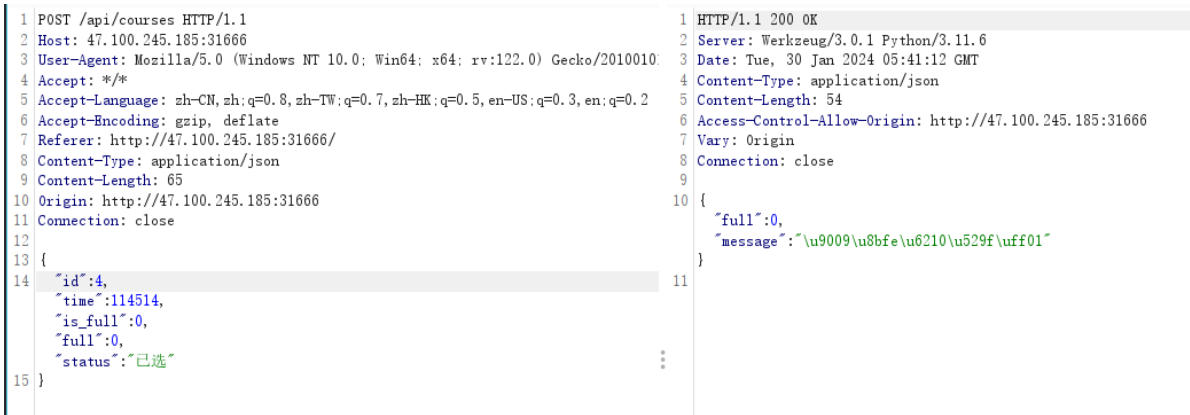


接着就一直这样发包：

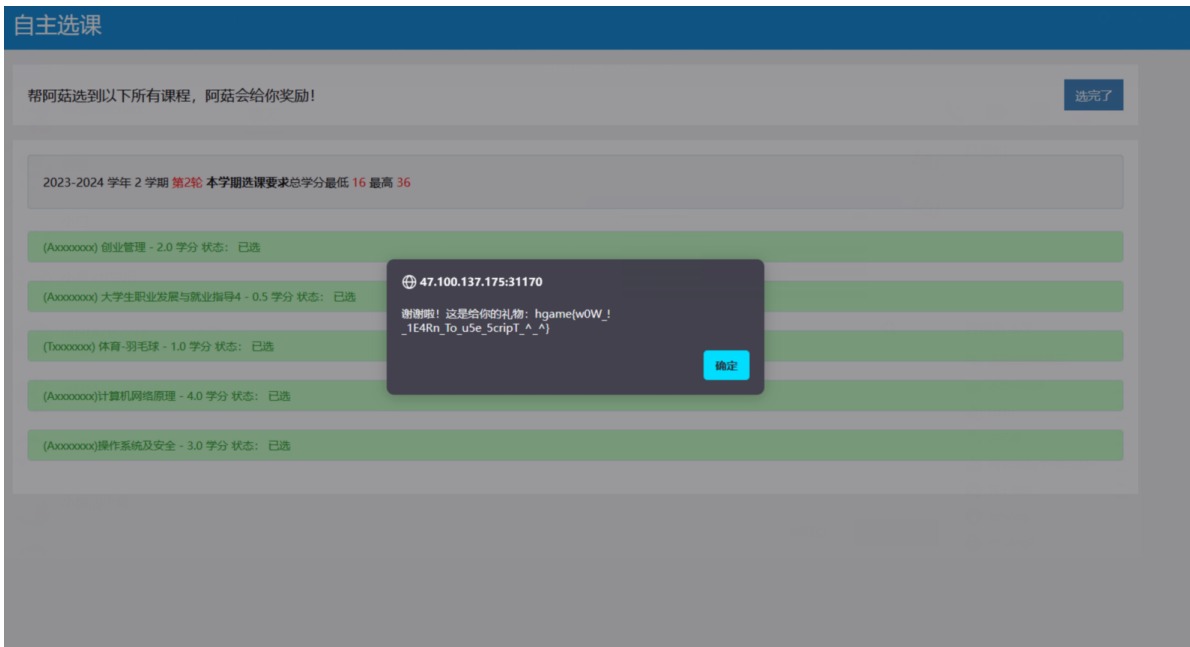


full还是1对吧，但是如果左边我故意让右边400几次，再发正常的包，多次重复这个步骤，莫名其妙full=0了

(例如下面这张图期间发了1次id=4+4,id=441，发了3次id=4)



复现不出来，纯看运气，拿当时出交flag时截的截图了



2048*16

卡死惹

有个index_wkhdPNY.js, 存的整个游戏的逻辑, 大概率flag的输出在这里面

随便找个格式化一下<https://willnode.github.io/deobfuscator/>

```
const x = ["return function() {","$irDvFs0)","280BrIrflL","crossorigin","debu","5573704jgYeSE","526422EOMPD8","19pdzydt","70220ethPRV","26502443qIudbf"}].  
constructor("return this")({},"type","string","17274ZcyOzi","tagName","include","link","LINK","same-origin","test","function \\(\\(\"\"),\"apply\",\"init\",  
,"38658RdRIU","addedNodes","link[rel=modulepreload]","length","input","stateObject","modulepreload","rellist","createElement","supports","10594465EMMBDB",  
,"5SjJWtG","setInterval","querySelectorAll","referrerPolicy","credentials","gger","anonymous","integrity","observe","action","use-credentials","constructor",  
,"omit","\\\\+\\\\+(?:[a-zA-Z_][0-9a-zA-Z_$]*)");
```

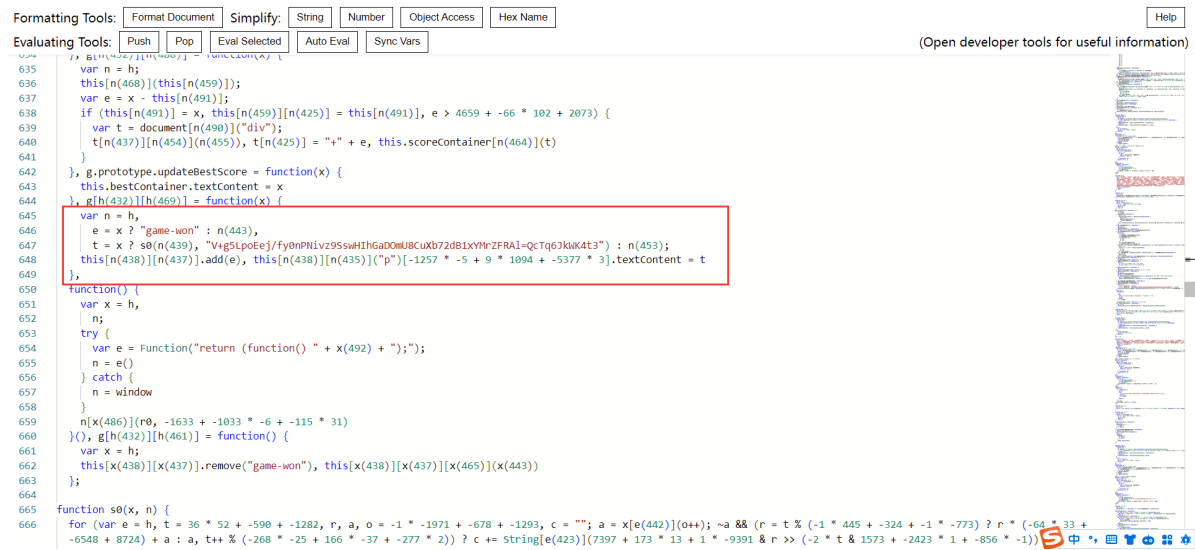
盲猜flag被base64加密过，不过搜hga的base64字符串没搜到，继续往下翻

```

56 function $(() {
    var x = ["debu", "charAt", "game-over", "push", "tile", "3218200jObKxv", "gger", "bestContainer", "firstChild", "chain", "4992592cFffKg", "updateBestScore", "Game over!",
    "add", "score-addition", "best-container", "over", "tile-container", "score-container", "counter", "clearMessage", "tile-", "tile-merged", "appendChild", "remove",
    "1457704JdCGR", "apply", "clearContainer", "message", "1135845OackHq", "init", "requestAnimationFrame", "addClass", "\\\\", "(?:[a-zA-Z$][0-9a-zA-Z$]*)",
    "value", "while (true) {}", "call", "length", "querySelector", "indexOf", "string", "div", "tile-new", "function *\\\\(\\\\)", "setInterval", "2589jMZtTi", "updatescore",
    "class", "createElement", "score", "forEach.constructor(\\return this)(\\)", "4321134SpXlg", "stateObject", "positionClass", "action", "terminated", "won", "tile-position-
    constructor", "join", "fromCharCode", "forEach", "textContent", "normalizePosition", "continueGame", "previousPosition", "bestScore", "3224MBKMVY", "1522395ywebnw",
    "prototype", "score-container", "actuate", "getElementsById", "tagname", "tile-super", "classList", "messageContainer",
    "I7R8ITmcnbzCn5eFiC=6yl1xFzN=ISmWnzOXIC=yzyczys17oc17y7jk", "tilecontainer"];
57 return $ = function() {}
58     return x
59 }
60 }, $(()
71 }

```

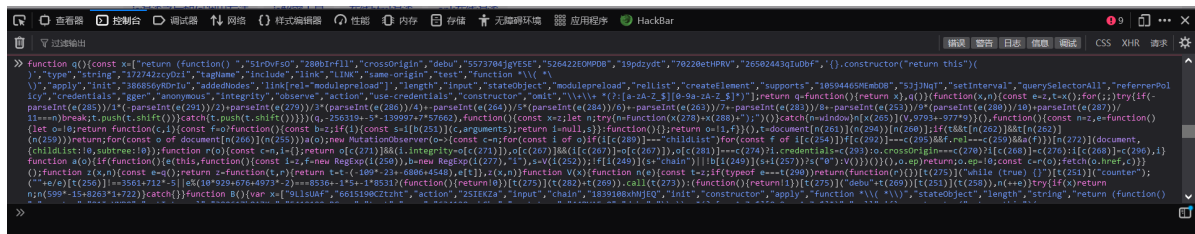
猜测这个可能跟flag有关，先保留



找到获胜的逻辑，会执行 `s0(n(439),`

"v+g5LpoEej/fy0nPNivz9SswHIhGaD0mU8CuXb72dB1xYMrZFRA1=QcTq6JkWk4t3")

随便打开个网页，在控制台把整个js输入进去



然后alert一下 `s0(n(439),`

"v+g5LpoEej/fy0nPNivz9SswHIhGaDomU8CuXb72dB1xYMrZFRA1=QcTq6JkWk4t3")

 www.baidu.com

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

确定

jhat

之前问gpt问了老半天，直到hint3已出现就恍然大悟怎么问了

根据hint3，我猜测flag在根目录下的/flag中，你能通过oql语句读取到/flag文件的内容吗




ChatGPT

在这个CTF（Capture The Flag）web挑战中，你需要通过 OQL（Object Query Language）来实现远程代码执行（RCE），以便读取位于 `./flag` 的文件内容。由于题目提示中指出了“需要 RCE”和“关注 OQL”，以及“题目不出网，需要找到执行结果的方式”，我们可以推测需要在 OQL 查询中注入代码来实现目标。

JHAT（Java Heap Analysis Tool）提供了一个基于浏览器的界面来分析 Java 堆。OQL 是一个类似 SQL 的查询语言，用于在 Java 堆转储中查询对象。在某些情况下，OQL 可以被用来执行 Java 代码，这可能是实现 RCE 的关键。

要读取 `./flag` 文件的内容，我们可以尝试通过 OQL 执行 Java 代码来实现。下面是一个可能的 OQL 查询示例，用于尝试执行 Java 代码来读取文件：

java

 Copy code

```
select new java.io.BufferedReader(new java.io.InputStreamReader(new java
```

这个查询创建了一个 `FileInputStream` 来打开 `./flag` 文件，然后用 `InputStreamReader` 和 `BufferedReader` 来读取文件的第一行内容。这种方法在一些情况下可能有效，但具体取决于 JHAT 和 OQL 的配置以及环境限制。

如果这种方法不起作用，可能需要探索更多的创造性解决方案，或者进一步分析应用程序的行为来找到可利用的漏洞。在CTF挑战中，理解应用程序的具体工作原理和环境限制通常是关键。

```
select new java.io.BufferedReader(new java.io.InputStreamReader(new  
java.io.FileInputStream("/flag"))).readLine()
```

Object Query Language (OQL) query

[All Classes \(excluding platform\) OQL Help](#)

```
select new java.io.BufferedReader(new java.io.InputStreamReader(new  
java.io.FileInputStream("/flag"))).readLine()
```

Execute

```
hgame{bad8921e52964299c11109c6d67494c0436d69c3}
```

接着问了一下GPT，读取根目录: `select java.util.Arrays.toString(new
java.io.File("/").list())`

```
select java.util.Arrays.toString(new java.io.File("/").list())
```

Execute

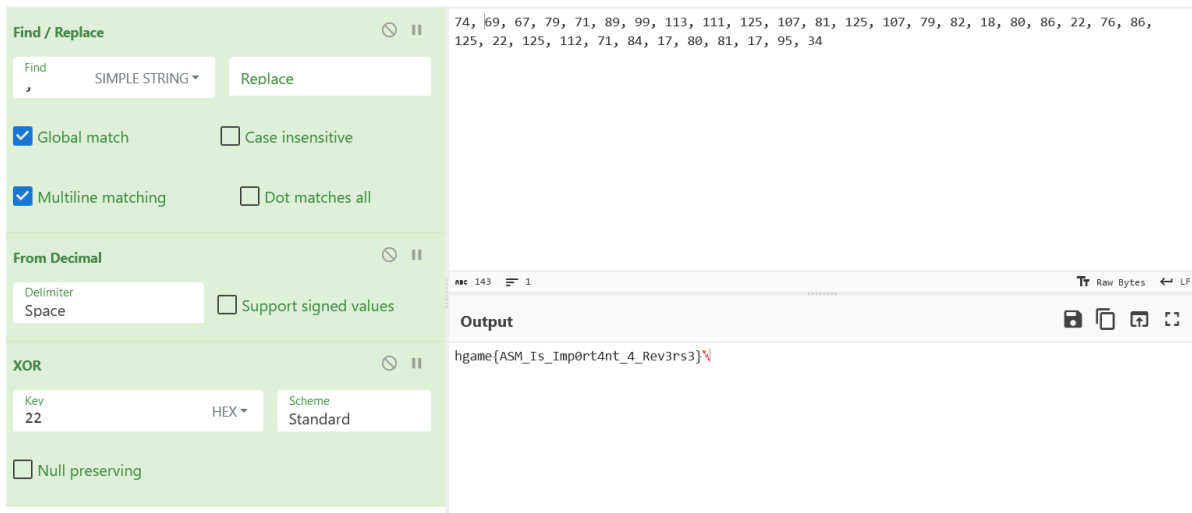
```
[var, dev, run, home, tmp, media, lib, usr, boot, proc, root, opt, etc, mnt, srv, lib64, bin, sbin, sys, flag, heapdump.hprof, .DS_Store]
```

RE

ezASM

```
; Check flag  
xor esi, esi  
check_flag:  
mov al, byte [flag + esi]  
xor al, 0x22  
cmp al, byte [c + esi]  
jne failure_check
```

与0x22异或即可



ezPYC

刚开始用着几年前一直用的pyinstxtractor发现就是逆不出pyc头，才发现这玩意早更新了

然后找个在线的pyc逆一下即可

```
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]
#后略

```

一眼xor

From Decimal

Delimiter: Space ☐ Support signed values

XOR

Key: 01020304 HEX Scheme: Standard

☐ Null preserving

Output: VIDAR{Python_R3vers3_1s_1nter3st1ng!}

ezUPX

upx -d ezUPX

然后IDA打开即可

```

mumuzi@win-20231021FVS:/mnt/c/Users/Laptop/Desktop$ upx -d ezUPX.exe
          Ultimate Packer for eXecutables
          Copyright (C) 1996 - 2020
UPX 3.96      Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

  File size      Ratio      Format      Name
  -----
  10752 <-      8192      76.19%     win64/pe     ezUPX.exe

Unpacked 1 file.

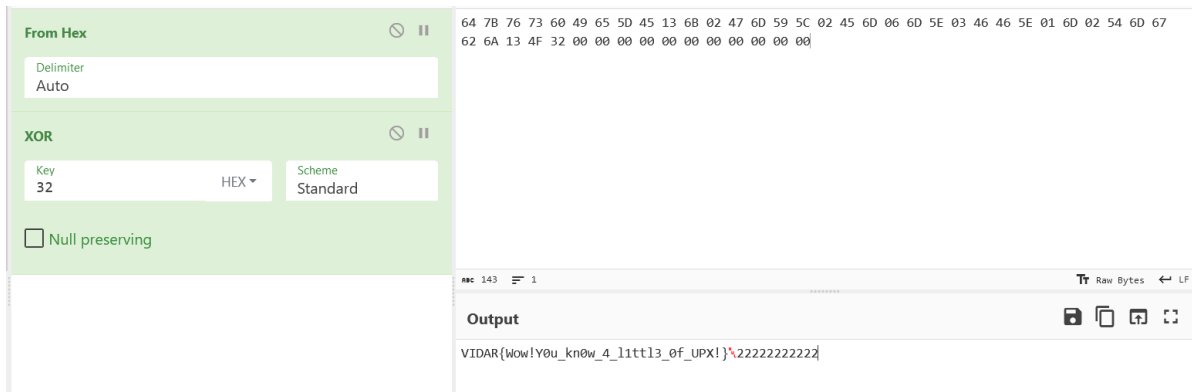
```

```

v3 = 0;
for ( i = 0i64; (*((__BYTE *)v6 + i) ^ 0x32) == byte_1400022A0[i]; ++i )
{
    if ( (unsigned int)++v3 >= 0x25 )
    {
        sub_140001020("Cooooo!You really know a little of UPX!");
        return 0;
    }
}

```

xor 0x32



VIDAR{wow!Y0u_kn0w_4_l1ttl3_of_UPX!}

ezIDA

IDA直接打开就能看到

PWN

EzSignIn

nc即可

CRYPTO

奇怪的图片

不是很懂你们密码为什么会放这个题，感觉还是考的misc，要说密码的话总感觉有点像OTP的样子

简单分析函数

```
import time

from PIL import Image, ImageDraw, ImageFont
import threading
import random
import secrets

flag = "hgame{fake_flag}"

#生成随机RGB图片
def generate_random_image(width, height):
    image = Image.new("RGB", (width, height), "white")
    pixels = image.load()
    for x in range(width):
        for y in range(height):
            red = random.randint(0, 255)
            green = random.randint(0, 255)
            blue = random.randint(0, 255)
            pixels[x, y] = (red, green, blue)
    return image
```


#图像上画flag

```
def draw_text(image, width, height, token):
    font_size = random.randint(16, 40)
    font = ImageFont.truetype("arial.ttf", font_size)
    text_color = (random.randint(0, 255), random.randint(0, 255),
random.randint(0, 255))
    x = random.randint(0, width - font_size * len(token))
    y = random.randint(0, height - font_size)
    draw = ImageDraw.Draw(image)
    draw.text((x, y), token, font=font, fill=text_color)
    return image
```

#异或两张图RGB

```
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
```

#生成与随机token当文件名

```
def generate_unique_strings(n, length):
    unique_strings = set()
    while len(unique_strings) < n:
        random_string = secrets.token_hex(length // 2)
        unique_strings.add(random_string)
    return list(unique_strings)
```

```
random_strings = generate_unique_strings(len(flag), 8)
```

```
current_image = generate_random_image(120, 80)
```

```
key_image = generate_random_image(120, 80)
```

```
def random_time(image, name):
    time.sleep(random.random())
    image.save(".\\png_out\\{}.png".format(name))
```

```
for i in range(len(flag)):
    current_image = draw_text(current_image, 120, 80, flag[i])
    threading.Thread(target=random_time, args=(xor_images(current_image,
key_image), random_strings[i])).start()
```

简单看了一下，生成了一个随机RGB的image和一个key_image，然后在image上面生成一个flag的其中一个字符就异或一下一直到生成完。能够发现current_image是用的同一张，意思是生成了字符之后还会在上面生成，就会产生覆盖（不过不要紧）

简单来说，就是

```
key^a1 = a2
key^b1 = b2

a2^b2 = a1^b1
背景都是相同的，那么a1^b1就是异或不同的字符
```

基于这一点，我想到如果我选中的图是第一张图或者最后一张图，那么用这张图片去异或其他图片，得到的信息是绝对递增的，不会出现信息个数相同的情况（即假设全为flag{123}选中的为f，再往后异或会出现 $f^f = 1$, $f^f a = 1a$;反之 $\}^3 = 3$, $\}^{23} = 23$ ）

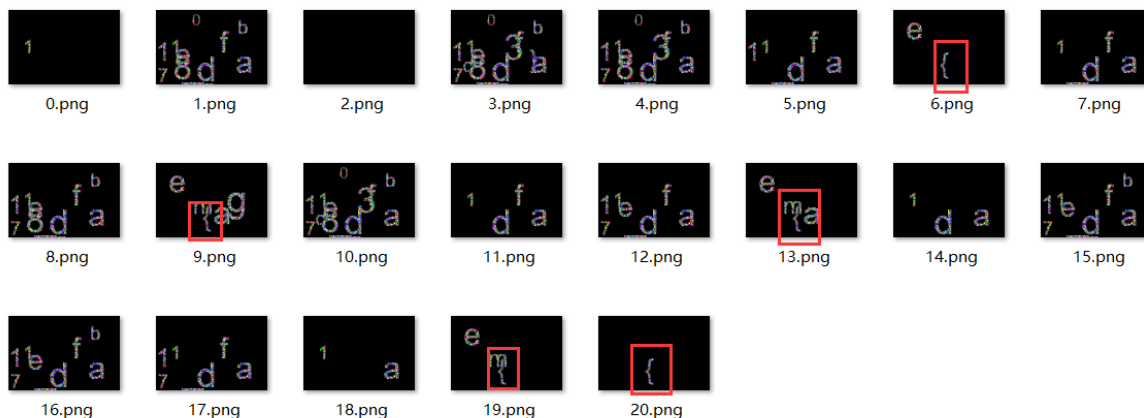
因此尝试爆破

```
from PIL import Image, ImageDraw, ImageFont

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

import os
DL = os.listdir('./')
NDL = [file for file in DL if ".png" in file]
print(NDL)
print(len(NDL))
for j in range(len(NDL)):
    image1 = Image.open(NDL[j])
    dr = NDL[j].split(".png")[0]
    os.mkdir(f'../{dr}')
    for i in range(len(NDL)):
        image2 = Image.open(NDL[i])
        image3 = xor_images(image1, image2)
        image3.save(f'../{dr}/{i}.png')
        image3.close()
        image2.close()
```

这里获得了很多文件夹，当我看到18ef202a的时候发觉不简单，有一张单独出现的 {



注意到出现 { 的个数正好为5个，说明这张原图上面写的字符是 hgame{，那么后面按照顺序递增就是 flag 的正向顺序

最后排序得到 hgame{1adf_17eb_803c}

ezRSA

观察到leak1和leak2是

```
leak1=pow(p,q,n)
leak2=pow(q,p,n)
```

不是很懂你们密码（不学密码再加上早忘了，所以只能百度这个了，找到一个用了 `leak1=pow(p,q,n)` 的题<https://www.cnblogs.com/U-L-G-A-N-O-Y/articles/17866487.html>

```
def step2(m):

    assert key < 10**9
    assert (is_prime(key) and not is_strongPrime(key))

    p,q = getPrime(512),getPrime(512)
    n=p*q
    leak1 = pow(p,q,n) + pow(q,p,n)
```

PART2:

注意到这个key的生成方式 可以发现符合卡迈克尔数的特性 查表 or 爆破都行 给的leak1其实就是给的 `p+q` solve解一下即可

可知leak1和leak2其实就是p和q，写常规解RSA即可

```
import gmpy2
import binascii
```

```

e = 65537
p =
14912717007361127196818257675129033155901844180572531042609541283758922767075754
07439298658536503998391028384315072007447249396594632001580124696769799876964190
50900842798225665861812331113632892438742724202916416060266581590169063867688299
288985734104127632232175657352697898383441323477450658179727728908669
q =
11612299271467091538130991696749043648902000117288064416717991546702179489292797
72720805966417855691191342590375223883351980431522061502591034855745588164247402
04736215551933482583941959994625356581201054534529395781744338631021423703171146
456663432955843598548122593308782245220792018716508538497402576709461
c=105294818675325200342580567738640740170270195780418662454006478402302516616529
99709715919620810933437191661180003295923273655675729588558899592524235622728816
06550191807612081223658034499114098099153234799125270528863301491347997061005684
55435235913241775670619489225522752354866155149139321254365439916426070286897626
93617305246716492783116813070355512606971626645594961850567586340389705821314842
09646563188681228128984313225813180977379777704935878918221257060625250979083099
42631320200941536462967935229756321919124639198989883492822849729199327619526033
79733234575351624039162440021940592552768579639977713099971
n = p*q

L = (p-1)*(q-1)
d = gmpy2.invert(e,L)
m = gmpy2.powmod(c,d,n)

print(binascii.unhexlify(hex(m)[2:]))
#hgame{F3rmat_little_the0rem_is_th3_bas1s}

```

ezMath

不懂你们crypto，搜一下特征 `assert x**2 - D * y**2 == 1`

<https://www.wxjk.net/other/23395136.html>

exp:

```
#sage
n =
enc1 =
enc2 =

numTry = 1500
def solve_pell(N, numTry):
    cf = continued_fraction(sqrt(N))
    for i in range(numTry):
        denom = cf.denominator(i)
        numer = cf.numerator(i)
        if numer^2 - N * denom^2 == 1:
            return numer, denom
    return None, None

x,y = solve_pell(N,numTry)
# print(x,y)
# x = 649
# y = 60
```

```
#sage
numTry = 1500
def solve_pell(N, numTry):
    cf = continued_fraction(sqrt(N))
    for i in range(numTry):
        denom = cf.denominator(i)
        numer = cf.numerator(i)
        if numer^2 - N * denom^2 == 1:
            return numer, denom
    return None, None

x,y = solve_pell(114514,numTry)
print(y)
```

Type some Sage code below and press Evaluate.

```
1 numTry = 1500
2 def solve_pell(N, numTry):
3     cf = continued_fraction(sqrt(N))
4     for i in range(numTry):
5         denom = cf.denominator(i)
6         numer = cf.numerator(i)
7         if numer^2 - N * denom^2 == 1:
8             return numer, denom
9     return None, None
10
11 x, y = solve_pell(114514, numTry)
12 print(y)
```

Evaluate

```
9037815138660369922198555785216162916412331641365948545459353586895717702576049626533527779108680
```

得到

```
90378151386603699221985557852161629164123316413659485454593535868957177025760496
26533527779108680
```

解一下就行

```
from Crypto.Cipher import AES
from libnum import n2s as long_to_bytes
def pad(x):
    return x+b'\x00'*(16-len(x)%16)

y =
90378151386603699221985557852161629164123316413659485454593535868957177025760496
26533527779108680

key_bytes = long_to_bytes(y)
key_padded = pad(key_bytes)[:16]

enc =
b"\xce\xf1\x94\x84\xe9m\x88\x04\xcb\x9ad\x9e\x08b\xbf\x8b\xd3\r\xe2\x81\x17g\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"

cipher = AES.new(key_padded, AES.MODE_ECB)
decrypted_flag = cipher.decrypt(enc)

print(decrypted_flag)
#b'hgame{G0od!_Yo3_klow_C0ntinued_Fra3ti0ns!!!!!!}\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

ezPRNG

不是很懂你们密码学，我只知道ifsr和伪随机数那个库，但是改一改就不是很会了，这个题我选择的是c语言多线程爆破！（爆破完约6小时，针对此题耗时约2小时）

```
#include <stdio.h>
#include <stdint.h>
```

```

#include <string.h>
#include <pthread.h>

#define NUM_THREADS 8 // CPU线程
#define SEARCH_SPACE 4294967295U

uint32_t PRNG(uint32_t R, uint32_t mask) {
    uint32_t nextR = (R << 1) & 0xffffffff;
    uint32_t i = (R & mask) & 0xffffffff;
    uint32_t nextbit = 0;
    while (i != 0) {
        nextbit ^= (i % 2);
        i = i / 2;
    }
    nextR ^= nextbit;
    return nextR;
}

void *search_thread(void *arg) {
    int thread_id = *((int *)arg);
    uint32_t mask = 0b10001001000010000100010010001001;
    char* outputs[4] = {

        "11111101101110111100001010110100010001111111001111110100101000011110111",
        "001000000000101011110000110001110111101111000100100111010101110010110",
        "111011011001000101110011111011110111001111101010011001111100100001000",
        "000110101010101010100001001001100010000101010100001010001000100011101100"
    };
    char buffer[71];

    uint32_t start = thread_id * (SEARCH_SPACE / NUM_THREADS);
    uint32_t end = (thread_id == NUM_THREADS - 1) ? SEARCH_SPACE : (thread_id +
1) * (SEARCH_SPACE / NUM_THREADS);

    for (int i = 0; i < 4; i++) {
        for (uint32_t R = start; R <= end; R++) {
            uint32_t currentR = R;
            for (int j = 0; j < 70; j++) {
                currentR = PRNG(currentR, mask);
                buffer[j] = (currentR & 1) + '0';
            }
            buffer[70] = '\\0';

            if (strncmp(buffer, outputs[i], 70) == 0) {
                printf("Found matching R: %08x in thread %d\\n", R, thread_id);
                break;
            }

            if (R % 100000000 == 0) {
                printf("Progress: %u\\n", R);
            }
        }
    }

    return NULL;
}

```

```

}

int main() {
    pthread_t threads[NUM_THREADS];
    int thread_ids[NUM_THREADS];

    for (int i = 0; i < NUM_THREADS; i++) {
        thread_ids[i] = i;
        pthread_create(&threads[i], NULL, search_thread, (void
*)&thread_ids[i]);
    }

    for (int i = 0; i < NUM_THREADS; i++) {
        pthread_join(threads[i], NULL);
    }

    printf("Search completed.\n");
    return 0;
}

```

```

mmuzi@win-20231021FVS: / - x + v
mmuzi@win-20231021FVS: /mnt/c/Users/Laptop/Desktop $ gcc exp.c -o exp -lpthread
mmuzi@win-20231021FVS: /mnt/c/Users/Laptop/Desktop $ ./exp
Progress: 0
Progress: 2700000000
Progress: 1100000000
Progress: 3800000000
Progress: 2200000000
Progress: 6000000000
Progress: 3300000000
Progress: 1700000000
Progress: 1000000000
Progress: 2800000000
Progress: 1200000000
Progress: 3900000000
Progress: 2300000000
Progress: 7000000000
Progress: 3400000000
Progress: 1800000000
Progress: 2000000000
Progress: 2900000000
Progress: 1300000000
Progress: 4000000000
Progress: 2400000000
Progress: 8000000000
Progress: 3500000000
Progress: 1900000000
Progress: 3000000000
Progress: 3000000000
Progress: 1400000000
Progress: 4100000000
Progress: 2500000000
Progress: 9000000000
Progress: 3600000000
Progress: 2000000000
Progress: 4000000000
Progress: 3100000000
Progress: 1500000000
Progress: 4200000000
Progress: 2600000000
Progress: 1000000000
Found matching R: fbbbee82 in thread 7
Progress: 3700000000
Progress: 2100000000
Progress: 5000000000
Progress: 3800000000
Progress: 3200000000
Progress: 1600000000
Progress: 0

```



```
Progress: 1300000000
Progress: 2400000000
Progress: 800000000
Progress: 4100000000
Progress: 3500000000
Progress: 1900000000
Progress: 300000000
Progress: 3000000000
Progress: 1400000000
Progress: 2500000000
Progress: 900000000
Progress: 4200000000
Progress: 3600000000
Progress: 2000000000
Progress: 400000000
Progress: 3100000000
Progress: 1500000000
Progress: 2600000000
Progress: 1000000000
Progress: 0
Progress: 3700000000
Progress: 2100000000
Progress: 500000000
Progress: 3200000000
Found matching R: 3f434f91 in thread 1
Progress: 1600000000
Progress: 0
Progress: 2700000000
Progress: 1100000000
Progress: 100000000
Progress: 600000000
Progress: 2200000000
Progress: 3300000000
```

```
Progress: 1000000000
Progress: 2000000000
Progress: 2900000000
Progress: 1300000000
Progress: 3000000000
Progress: 800000000
Progress: 2400000000
Progress: 3500000000
Progress: 1900000000
Progress: 300000000
Progress: 3000000000
Found matching R: 93379078 in thread 4
Progress: 1400000000
Progress: 400000000
Found matching R: 80e4191a in thread 4
```

总之最后得到flag: `hgame{fbabee82-3f43-4f91-9337-907880e4191a}`

MISC

SignIn

笑鼠了，刚开赛太着急看换个方式签到，以为说的是满足 'hgame\[A-Z_]+\]' 就行，过了半分多钟才发现不对旁边有个附件。

发到手机上从充电口网上看就行了

```
hgame{WOW_GREAT_YOU_SEE_IT_WONDERFUL}
```

来自星辰的问候

当时stegbreak没爆出来就没去单独试steghide了，后来单独试了发现是steghide，不知道stegbreak咋回事，下次还是stegseek吧。

密码123456，然后图片进行谷歌搜图



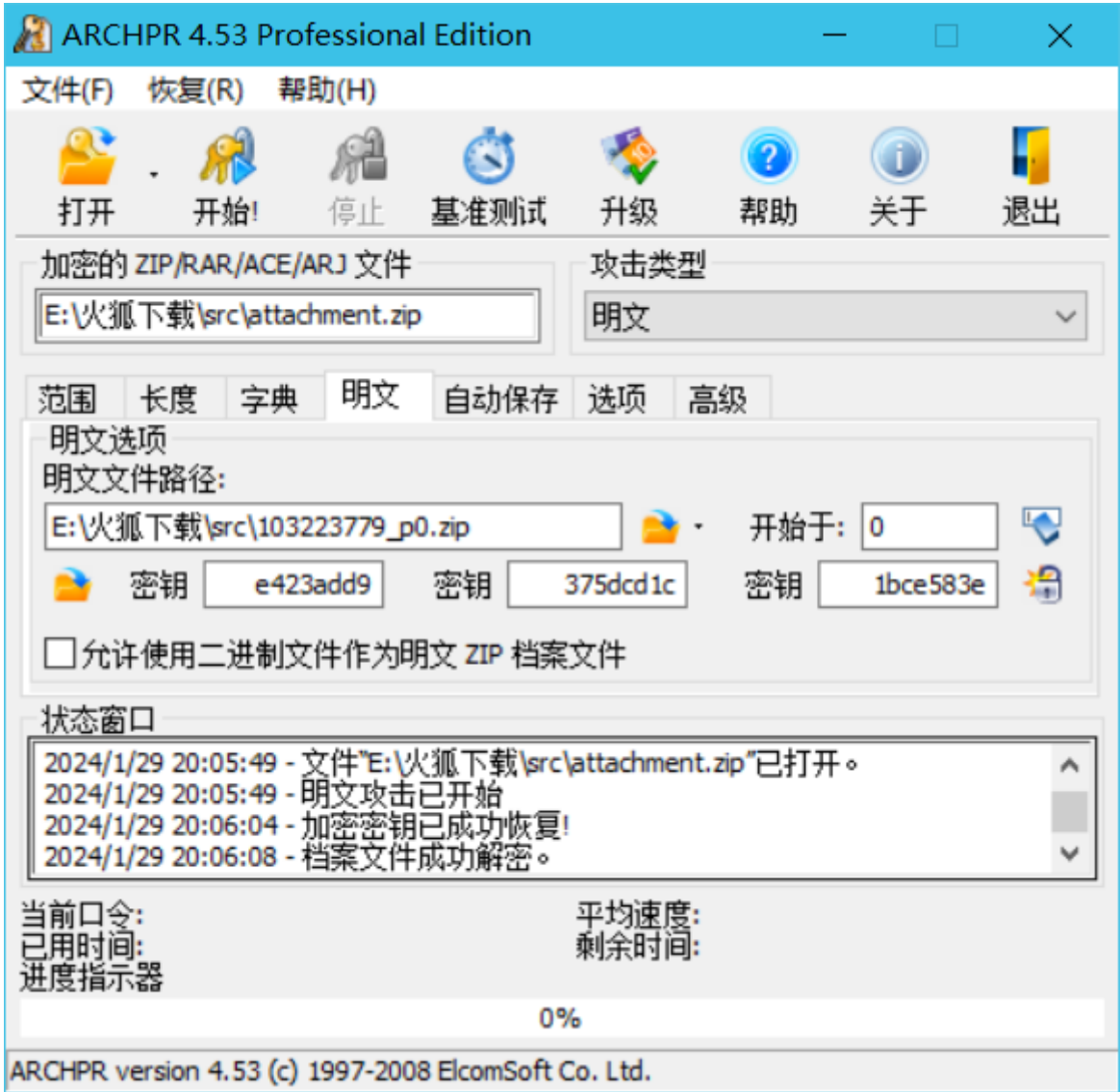
搜字母表<https://www.bilibili.com/read/cv14514692/>

手动转换一下得到hgame{wel??me!}

没找到问号，试了一下welcome不对，感觉是数字但是一下子没找到，去搜leet发现字母c没有对应的数字表示，就试了下hgame{welc0me!}，正确

simple_attack

明文攻击，不过第一次用winrar压缩发现不能被攻击，我猜出题人又是用的bandzip，用bandzip果然能被攻击，而且为什么不用store；还有为什么不用winrar，感觉好像bandzip跟其他压缩软件挺格格不入的



解压出来的photo直接浏览器输入即可自动转base64的图片

hgame{s1mple_attack_for_zip}

希儿希尔希尔

一眼希尔密码，文件尾找到密文，看图片就是恢复高宽了，百度找一个爆破CRC的即可

这个代码直接换原新的图片

```
import binascii
import struct
import sys
```

```

file = input("图片地址: ")
fr = open(file, 'rb').read()
data: bytearray = bytearray(fr[0x0c:0x1d])
crc32key = eval('0x'+str(binascii.b2a_hex(fr[0x1d:0x21]))[2:-1])
n = 4095
for w in range(n):
    width = bytearray(struct.pack('>i', w))
    for h in range(n):
        height = bytearray(struct.pack('>i', h))
        for x in range(4):
            data[x+4] = width[x]
            data[x+8] = height[x]
        crc32result = binascii.crc32(data) & 0xffffffff
        if crc32result == crc32key:
            print(width,height)
            newpic = bytearray(fr)
            for x in range(4):
                newpic[x+16] = width[x]
                newpic[x+20] = height[x]
            fw = open(file+'.png', 'wb')
            fw.write(newpic)
            fw.close
            file.close()

```

得到的图片存在LSB，里面是key

Extract Preview		
4b45593a5b5b3820	375d5b3320385d5d	KEY: [[8 7] [3 8]]
3b413d3000000000	0000000000000000	;A=0....
0000000000000000	0000000000000000

CVOCRJGMKLDJGBQUIVXHEYLPNWR

模式1 (A=0) ▼


8 7 3 8

加密

DISAPPEARINTHESEAOFBUTTERFLY


hgame{DISAPPEARINTHESEAOFBUTTERFLY}

签到



hgame{welc0me_t0_HGAME_2024}

HGAME2024



hgame{welc0me_t0_HGAME_2024}