Misc

灵活奋斗

> description:
>
> 做累了题目就来小摆一会吧

这题拿到一个 摆烂.gif，可以逐帧查看动图，发现有四帧是带有二维码的

扫码得到

```
6CPZDC7QT6I2H4E7SGG7BH4QXHYJ7EMI6CPZDDHQT6IYL4E7SGGPBH4RRHYJ7ENF6CPZDK7QT6IY74E7SGC7BH4QX7YJ7EMJ
6CPZDIPQT6IZR4E7SC57BH4QXXYJ7ENM6CPZDEPQT6IKR4E7SCYPBH4RUXYJ7EMY6CPZDDXQT6IZD4E7SGO7BH4RTDYJ7EMP
6CPZDBPQT6IZ34E7SGEPBH4RRXYJ7EFQ6CPZDJPQT6IYJ4E7SC6PBH4QWDYJ7ENC6CPZDD7QT6IKV4E7SGN7BH4RT7YJ7EM4
6CPZDDPQT6ILB4E7SCT7BH4RQTYJ7EF56CPZBMHQT6I2F4E7SGIPBH4RR7YJ7EM36CPZDIXQT6IZT4E7SC57BH4RQXYJ7EFQ
```

一眼base32，解得

```
👌👎👍🐯👈👌🆒👌👉👭👩‍👩‍👦👌🆒🔪👉🛌🏠🐻🙈👩‍👩‍👧‍👧🐯🐰🦉🏠🏆🚟🗃🏠👌🆒🗃👈👍🐰👭👄
🐼🐰🦵👌🦙🐑🛥🔒👍🐰🧚‍♀️👄🙈🐰🦵🤲👌🐑🦵🐮🐯🆒🐰
```

emoji加密就那么几种，尝试后发现是base100，解得

```
TlVBQUNURntXNHRjaDFuZ19naWZfaXNfQV9nME9kX3dheV90MF9kYXdkbDN9
```

一眼base64，解得

```
NUAACTF{W4tch1ng_gif_is_A_gOOd_way_tO_dawdl3}
```

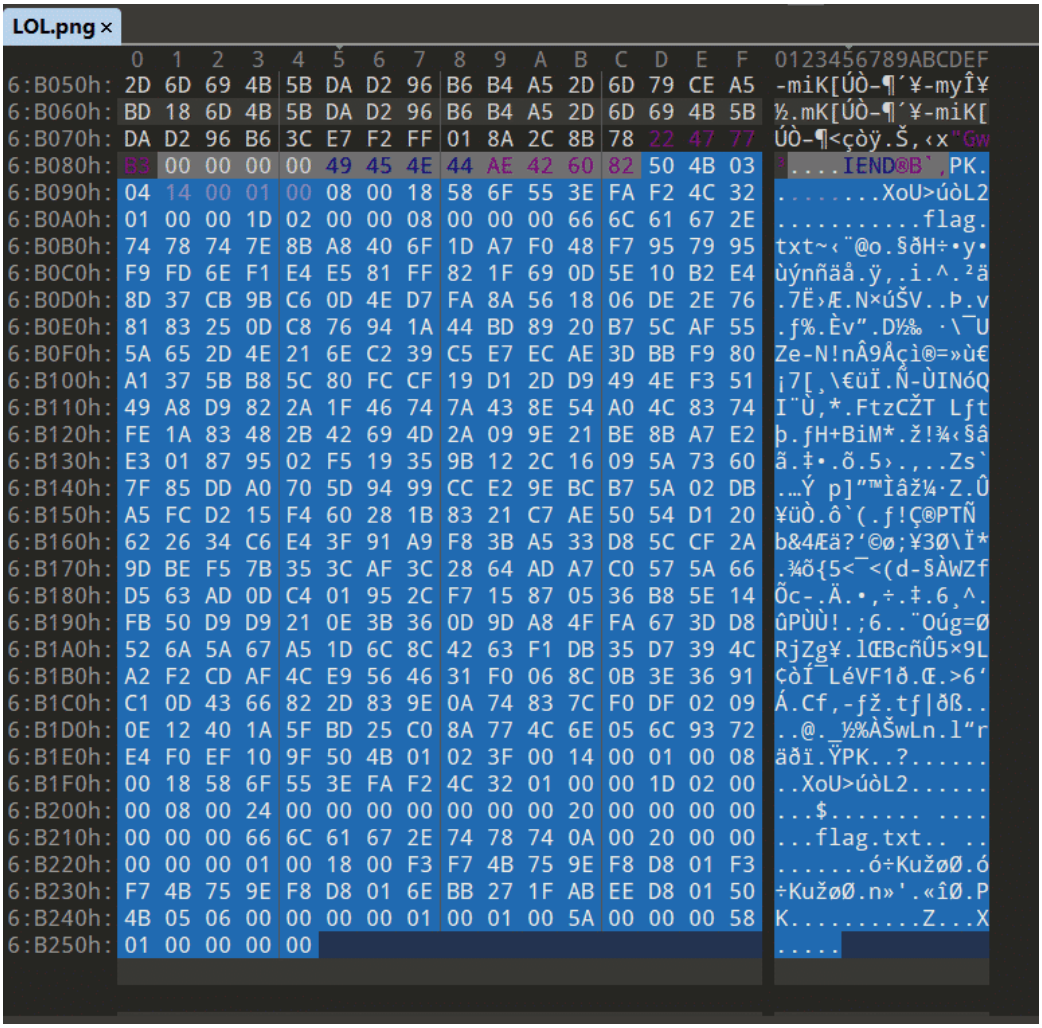
> 看见现场很多选手被三层base折磨，出题人很愧疚，下次一定套更多（

英雄联盟

> description:
>
> 这是一张平平无奇的战绩图，刚入坑的$2zz用寒冰打出了0-14的逆天战绩，但其中好像还藏着什么秘密。好的题目，爱来自瓷器，英雄联盟

首先拿到 LOL.png，然后发现 $2zz的0-14 和 dw的2-16 是如此显眼（bushi

小小分析一手，010 editor 或者 winhex 打开图片，会发现末尾附加了一个压缩包



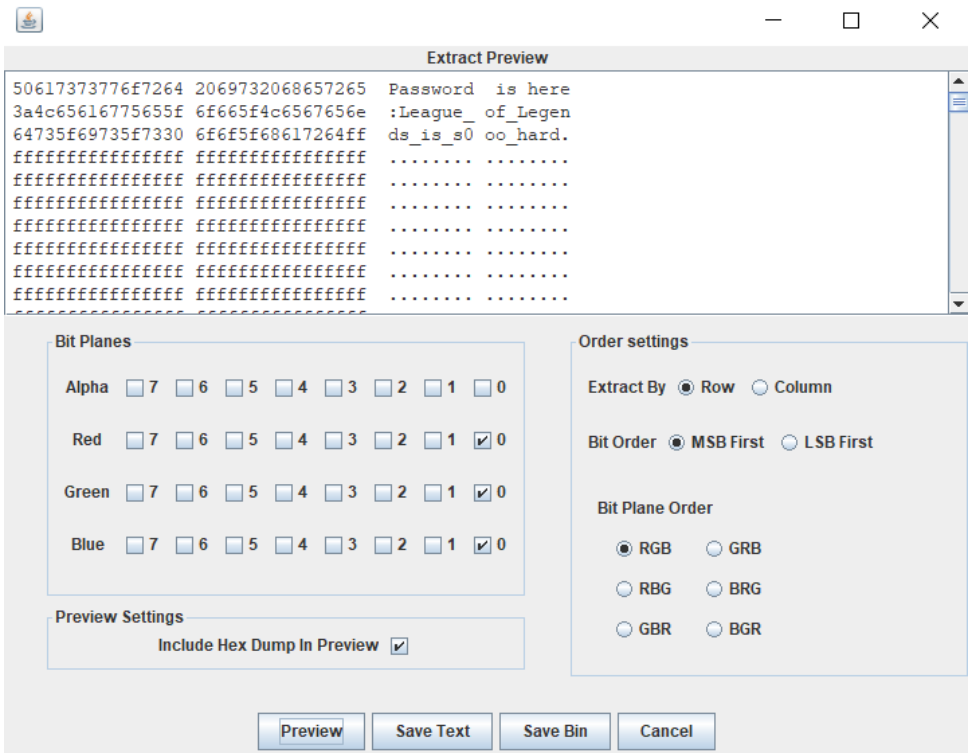
一般png图片结尾是 AE426082，而这里正常png结尾之后多了一串 504B0304 开头的内容，而504B0304是zip压缩包的文件头

所以把后面这一部分提取出来，保存为zip压缩包

> 或者以上这些步骤，也可以在kali虚拟机里，用binwalk+foremost完成分离

总之，分离出zip压缩包之后，里面是flag.txt，但是需要密码

stegsolve查看图片RGB通道的最低位，发现密码 `League_of_Legends_is_s0oo_hard`



> 这一步也可以用kali虚拟机中的zteg来完成，原理都是一样的，就是查看图片各个通道的情况

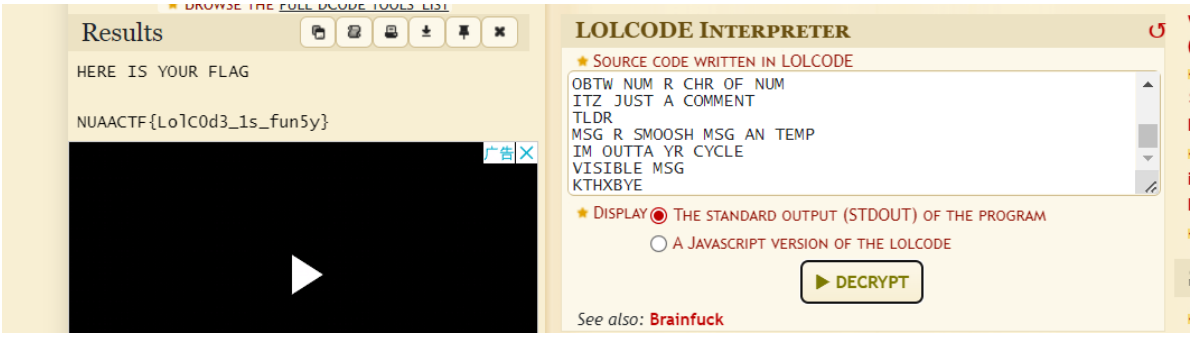
用密码解压之前的压缩包，得到一串奇怪的字符

```
HAI NUAACTF code
VISIBLE "HERE IS YOUR FLAG:)"
I HAS A CODE ITZ "LSBBDRGyJmjD1b4]2q]dsl6w{"
I HAS A MSG ITZ ""
I HAS A INDEX ITZ 0 BTW ITZ INDEX
I HAS A NUM ITZ
IM IN YR CYCLE UPPIN YR INDEX WILE INDEX SMALLR THAN LEN OF CODE
I HAS A C ITZ CODE!INDEX
NUM R ORD OF C
NUM R SUM OF NUM AN -3
I HAS A TEMP
NUM BIGGER THAN 69, O RLY?
YA RLY
NUM R SUM OF NUM AN 5
NO WAI
NUM R SUM OF NUM AN 2
OIC
TEMP R CHR OF NUM
OBTW NUM R CHR OF NUM
ITZ JUST A COMMENT
TLDR
MSG R SMOOSH MSG AN TEMP
IM OUTTA YR CYCLE
VISIBLE MSG
KTHXBYE
```

尝试搜索里面的某些语句，会发现这是 lolcode 语言，这是一种esolang（奇特的编程语言）

也会搜索到相似的ctf题 [第四届"安洵杯"网络安全挑战赛 CyzCC_loves_LOL]

找到解码网站 LOLCODE Language - Compiler - Online Decoder, Encoder, Translator (dcode.fr) 运行即可



这里有些人可能需要代理才能上这个网站，有些人不需要代理直接能上，很玄学（

其实不用上这个解码网站也能解出题目，有一队学妹直接对着lolcode手搓出了对应的python脚本（tql

> lolcode 是用网络上一些缩写流行语来完成编写的，比如BTW 是 by the way 的缩写，by the way 经常用于换个话题，相当于常说的 “对了…”，在lolcode里就代表注释。
>
> 题目描述中 “好的题目，爱来自瓷器，英雄联盟” 也是玩了翻译梗，因为 lol 不仅有英雄联盟的意思，在网络流行语中还代表 “笑” 的意思，这里考点就是 lolcode。
>
> 所以 lolcode 可阅读性很高，可以直接阅读然后猜出里面的意思和逻辑。
>
> 出题人是直接仿照上面 **第四届”安洵杯”网络安全挑战赛 CyzCC_loves_LOL** 那题，稍微改动一下出的。但是原题写的和最正统的lolcode就有区别，很多语法官方文档里没有，所以很多选手找到了lolcode在线运行的网站，但是无法运行，必须得用上面的解密网站才能正确运行。

给出一个对应的python脚本，可以和 lolcode 对应一下，可以发现是真的简单易懂

```python
print("HERE IS YOUR FLAG\n")
code = "LSBBDRGyJmjD1b4]2q]dsl6w{"
msg = ""
index = 0
# 定义变量num
while(index < len(code)):
    c = code[index]
    num = ord(c)
    num = num + (-3)
    # 定义变量temp
    if(num > 69):
        num = num + 5
    else:
        num = num + 2
    temp = chr(num)
    msg = msg + temp

    index += 1

print(msg)
# HERE IS YOUR FLAG
# NUAACTF{LolCOd3_1s_fun5y}
```

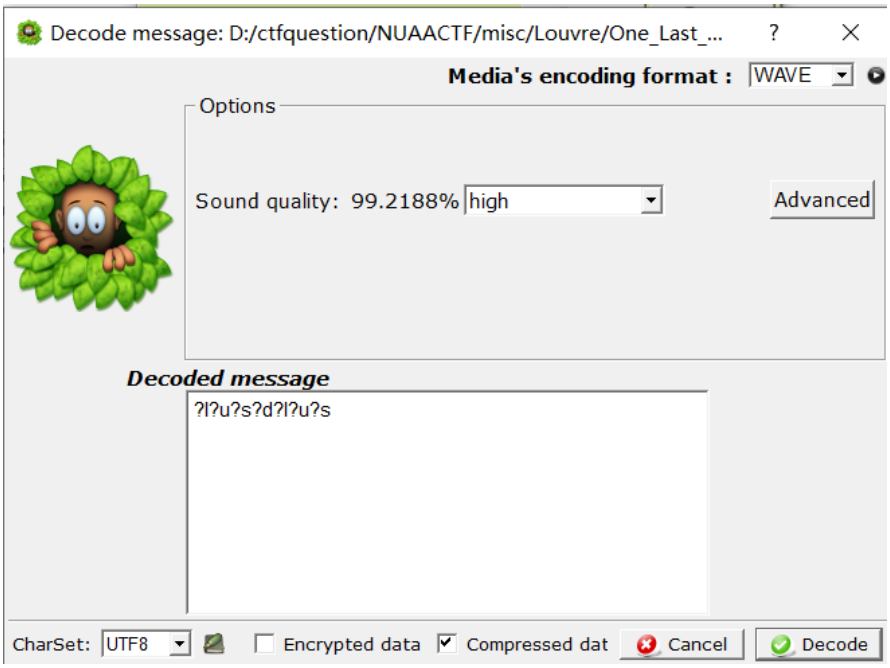
Louvre

> description:
>
> 卢浮宫的深处是蒙娜丽莎寂静的目光
>
> hint:
>
> 约翰的英文是?

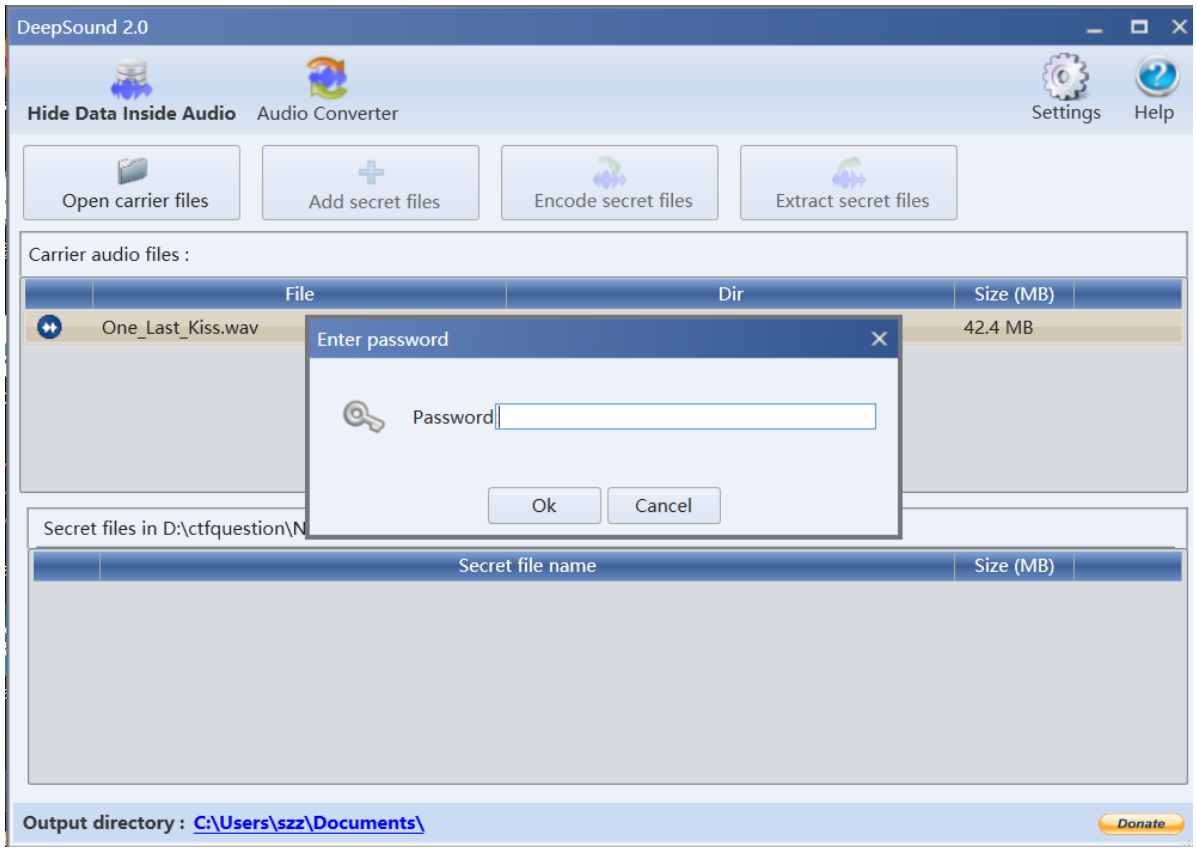
拿到一个One_Last_Kiss.mp3，但其实是个wav音频文件，小改一下后缀名

然后根据题目描述，很容易想到两个wav音频的隐写工具，silenteye 和 deepsound

先用silenteye解密，发现 `?l?u?s?d?l?u?s`



然后拖入deepsound发现要密码



那么用silenteye解出的那一串东西，肯定是和deepsound的密码有关

hint给出了john

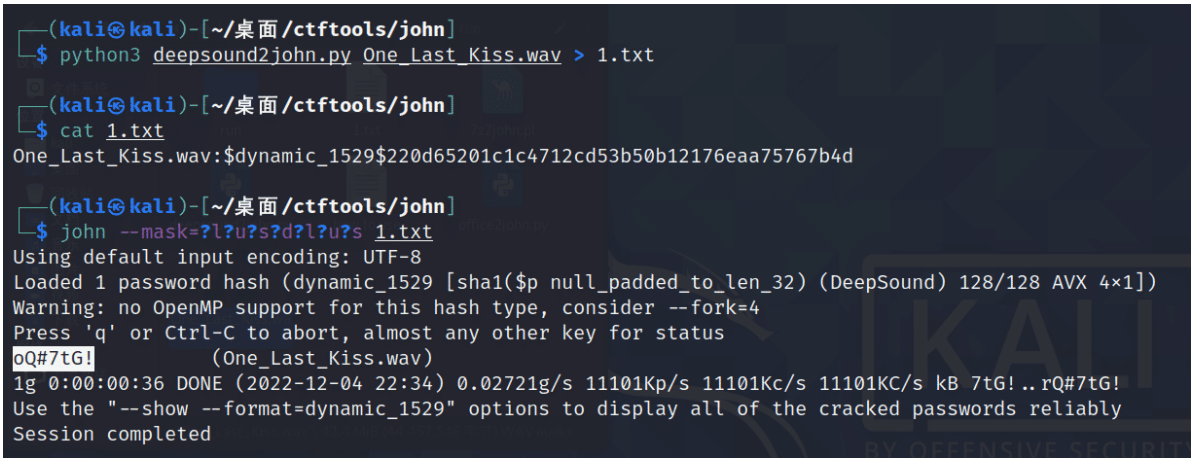
搜索后发现了john爆破deepsound密码的操作，例题可参考`[INSHack2018] (not) so deep`

那一串东西是掩码，因此是john掩码爆破deepsound密码

> john是一种密码破解工具，kali上是自带的

然后如下操作，先用 deepsound2john.py 提取john要爆破的hash，然后直接掩码爆破

```
python3 deepsound2john.py One_Last_Kiss.wav > 1.txt
john --mask=?l?u?s?d?l?u?s 1.txt
```



得到密码`oQ#7tG!`，deepsound解一下就行

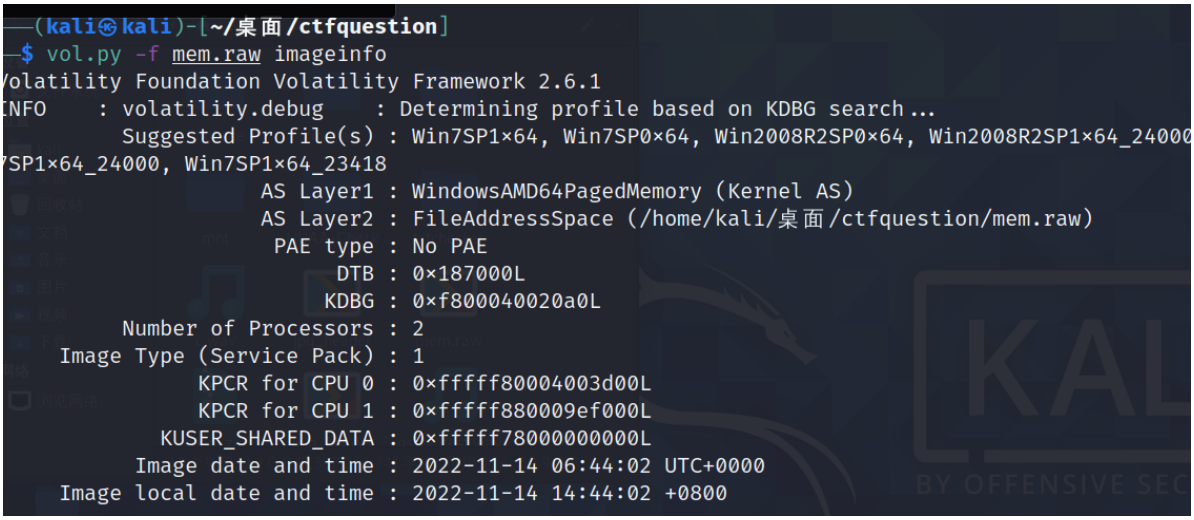
> 这题本来出题人只是想简单地出一个silenteye + deepsound，silenteye解出来的就是deepsound的密码
>
> 但是后来发现南邮新生赛0xGame2022有一个相同的
>
> 所以引入了john爆破变化了一下，但是难度似乎太大了（

ez_Forensics

.raw文件，上内存取证

> kali2020及之后不再自带volatility，可以自行安装
>
> 可以选择从源码安装，也可以选择直接下载release的二进制文件 Volatility 2.6 Release (volatilityfoundation.org)

先查看imageinfo，得到 Win7SP1x64，是个win7系统



`vol.py -f mem.raw --profile=Win7SP1x64 filescan|grep "Desktop"`

发现桌面上有一个flag.zip，注释里提供了5位掩码，爆破一下，得到

> NUAACTF{this_is_fake_flag}
>
> It's fake! I wouldn't put a secret in such an obvious place 🤣
> I hid it somewhere on the Internet and no one can find it 🤓

flag是假的，但是提示了 `hid it somewhere on the Internet`

`vol.py -f mem.raw --profile=Win7SP1x64 netscan`，查看网络连接状况

```
0x7ddcd010       TCPv6    :::49152                :::0                   LISTENING        408      wininit.exe
0x7eb47710       UDPv4    127.0.0.1:52790         *:*                                     2688     iexplore.exe    2022-11-14 06:43:45 UTC+0000
0x7eb9e010       UDPv4    0.0.0.0:3702            *:*                                     1308     svchost.exe     2022-11-14 06:43:41 UTC+0000
0x7eb9e010       UDPv6    :::3702                 *:*                                     1308     svchost.exe     2022-11-14 06:43:41 UTC+0000
0x7ed7fef0       TCPv4    0.0.0.0:49152           0.0.0.0:0              LISTENING        408      wininit.exe
0x7ea254d0       TCPv4    192.168.20.137:49160    23.203.70.208:80       ESTABLISHED      2688     iexplore.exe
0x7f067550       TCPv4    192.168.20.137:49163    42.192.224.58:8002     CLOSED           2652     iexplore.exe
0x7fc3e3a0       UDPv4    0.0.0.0:52310           *:*                                     1060     svchost.exe     2022-11-14 06:43:55 UTC+0000
0x7fccb3a0       UDPv4    0.0.0.0:52310           *:*                                     1060     svchost.exe     2022-11-14 06:43:55 UTC+0000
0x7fd583a0       UDPv4    0.0.0.0:52310           *:*                                     1060     svchost.exe     2022-11-14 06:43:55 UTC+0000
```

发现 iexplore.exe 浏览器曾经和 42.192.224.58:8002 建立过连接，很可疑

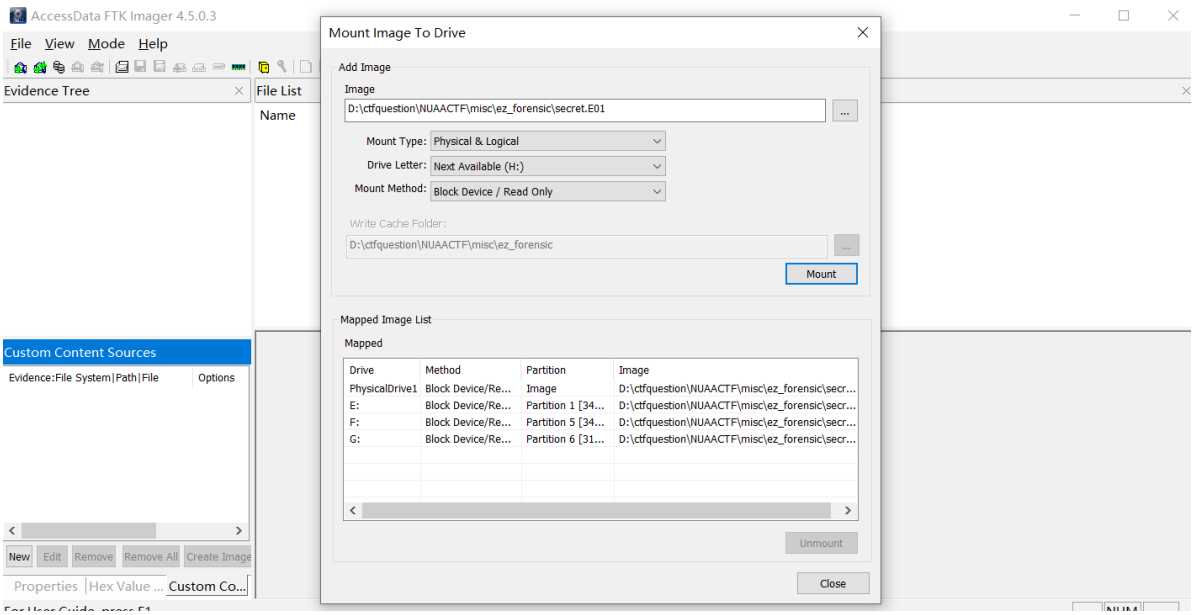
查看ie浏览器历史记录，抓取一下这个网址

```
┌──(kali㉿kali)-[~/桌面/ctfquestion]
└─$ vol.py -f mem.raw --profile=Win7SP1x64 iehistory | grep "42.192.224.58:8002"
Volatility Foundation Volatility Framework 2.6.1
URL: S2zz@http://42.192.224.58:8002
Location: Visited: S2zz@http://42.192.224.58:8002/favicon.ico
Location: Visited: S2zz@http://42.192.224.58:8002/favicon.ico
Location: Visited: S2zz@http://42.192.224.58:8002/real_target.zip
Location: Visited: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002/real_target.zip
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
Location: Visited: S2zz@http://42.192.224.58:8002/favicon.ico
Location: Visited: S2zz@http://42.192.224.58:8002/favicon.ico
Location: Visited: S2zz@http://42.192.224.58:8002/real_target.zip
Location: Visited: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002/real_target.zip
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
Location: Visited: S2zz@http://42.192.224.58:8002
Location: Visited: S2zz@http://42.192.224.58:8002/real_target.zip
Location: Visited: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
Location: :2022111420221115: S2zz@http://42.192.224.58:8002/real_target.zip
Location: :2022111420221115: S2zz@http://42.192.224.58:8002
```

发现从这个网站下载了一个real_target.zip，那就访问一下

发现确实有一个real_target.zip，下载解压后得到 secret.E01

接下来是磁盘取证分析，用FTK Imager挂载

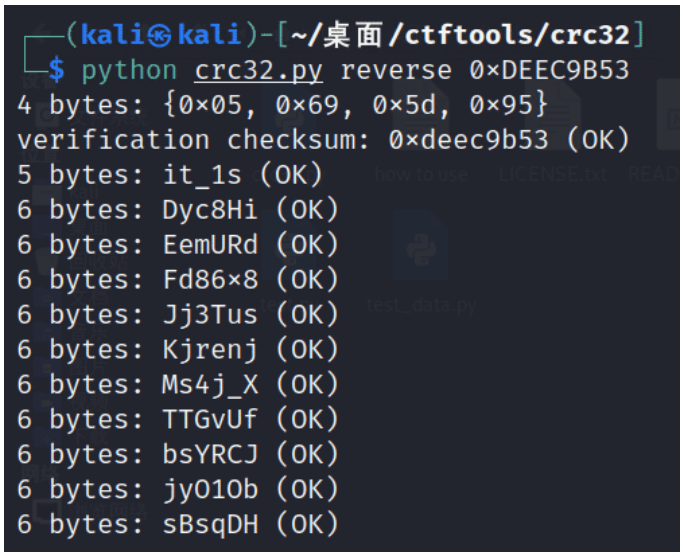


挂载出来的盘里，可以得到一个real_flag.zip

key被分成了3部分，但是每个部分都不超过6字节大小，太小了，可以利用CRC32的值进行爆破

可以利用github上现有的脚本: [theonlypwner/crc32: CRC32 tools: reverse, undo/rewind, and calculate hashes (github.com)]

比如跑key1.txt，得到第一段key1: `it_1s`



也可以自己手搓一个crc32爆破脚本（

三个跑完，得到完整密码 `it_1s_safest_p4ss`，解密real_flag.txt，得到flag

> 这题就是简单的内存取证和磁盘取证合在了一块，有点小坑但是并不是很难，可惜没有人做，出题人很伤心

Blockchain

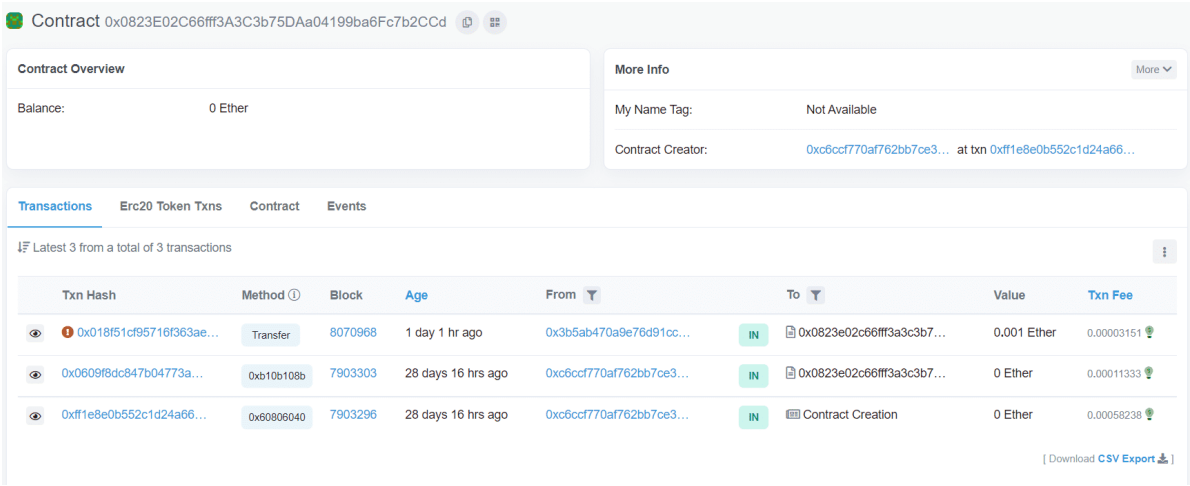
Forever Love

> description:
>
> $2zz的女朋友听说区块链上的数据永远不可能被篡改，缠着他在区块链上写了一个浪漫的表白话语，你能找到是什么吗 0x0823E02C66fff3A3C3b75DAa04199ba6Fc7b2CCd@Goerli
>
> hint:
>
> 可以看看题目地址上交易的具体信息，可能秘密就藏在交易里

这题只需要一个浏览器就能做

区块链浏览器 etherscan.io 能够查看链上的各种信息

直接去etherscan.io，切换 goerli测试网，搜索这个地址

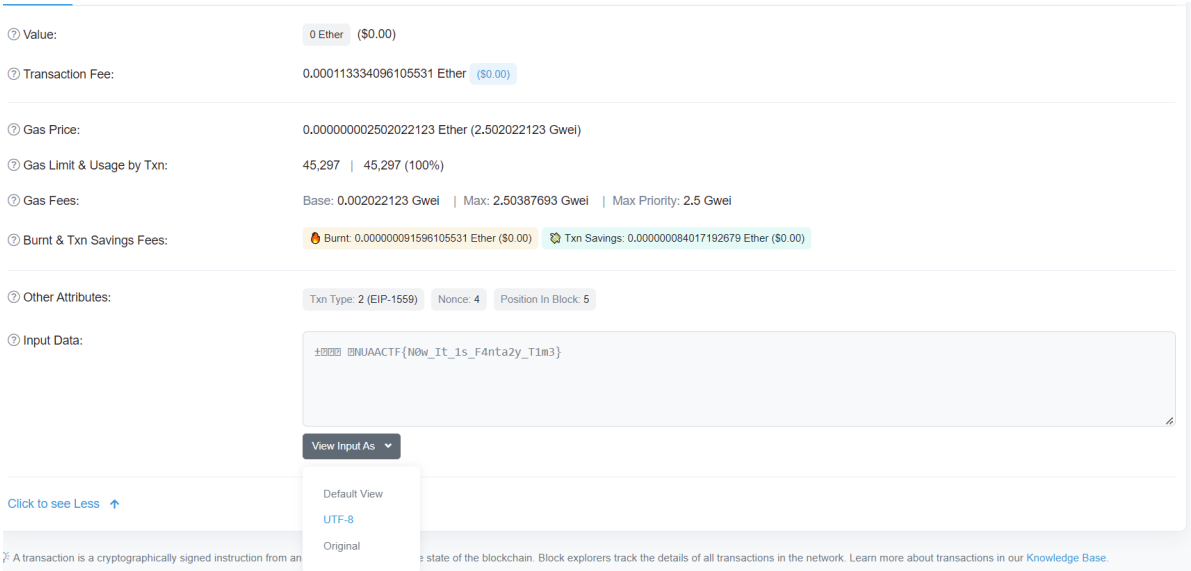


怎么还有人给这个地址转账（

这是个合约地址，不是账户地址，转账显然会失败（

根据hint，看每一笔交易的具体信息，其中第二笔交易，即0x0609f8... 那一笔交易

查看input data，直接utf-8格式查看，就能得到flag



`NUAACTF{NOw_It_1s_F4nta2y_T1m3}` 现在是，幻想时间!

> 如此简单清纯，居然只有三支队伍写了，我很伤心

SecretNumber 1

> description:
>
> 猜中这个神秘的数字　Goerli水龙头：https://goerlifaucet.com/ 也可以找出题人领取测试币
>
> nc 42.192.224.58 20000
>
> hint:
>
> 怎么查看链上合约的公共变量呢?

这里稍微需要一点solidity的知识，一点remix交互的知识。我会写的很详细。

nc连接上之后，先查看源码

```
pragma solidity 0.8.7;

contract HiddenSecret1 {
        uint8 nonce;
        uint256[] public secrets;

        event isSolved();

    constructor(uint256 _secret) public {
        nonce = 0;
        secrets.push(_secret);
    }

    function guess(uint256 _answer) public {
        nonce = nonce + 1;
        require( nonce <= 3 );
        if( keccak256(abi.encodePacked(secrets[0])) ==
keccak256(abi.encodePacked(_answer)) ) {
                emit isSolved();
        }
    }
}
```

简单解释一下，constructor构造函数，把输入 _secret 压入了secrets数组

guess函数，如果 _answer 和 secrets[0] 相等，就会触发isSolved()事件，我们的目标就是触发这个事件

所以要得到 secrets[0] 这个值，而 secrets 是 public 属性，是能够直接交互获取的

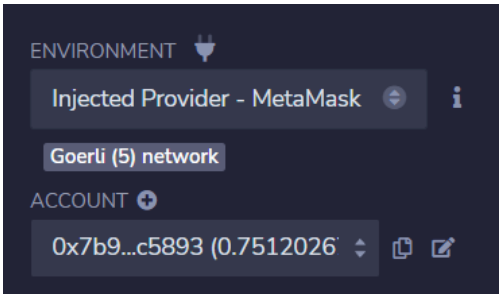
先nc连接选择1，给deployer account转点钱，这样题目就能部署合约用于挑战（metamask钱包的操作就不教了，可以自己搜索

转账完成后选择2，会给出部署好的合约地址

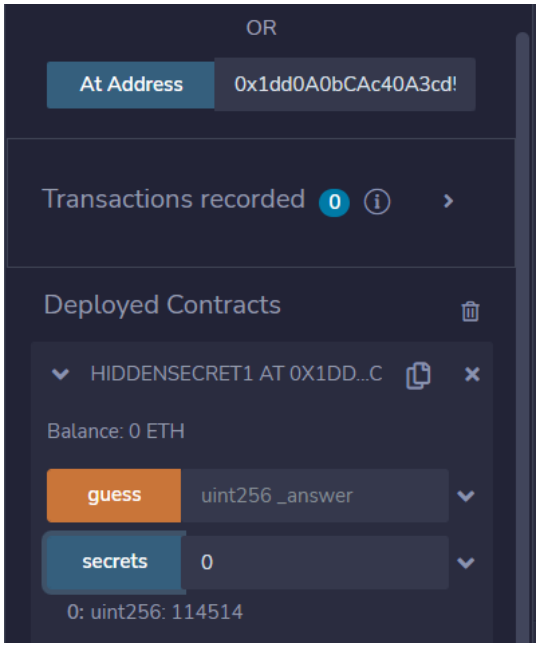
```
┌──(kali㉿kali)-[~]
└─$ nc 42.192.224.58 20000
I hide my secret number in the contract. I think it's very easy for you!
Your goal is to make isSolved() event emitted!

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 1
[+] deployer account: 0×b28107C47433C20aFF2A6790De12770E7D3C3C3c
[+] token: v4.local.nbM5dLZYtdJKurG2EDlWp8S8UjYT3YEfLVjxNZdsv57BdwQd396f9SpwDuxdO7rxDVsOp1a3TUbSqiUX9iejsjU_s-PxE1pi0hkxhs6rb_WUG
TuwBwgIR3iCNF1HR43R3IRjCFEteJysVNh1Wzz4KFwbokRQDJFb3r0YGfRD5Mu2zw
[+] please transfer 0.001 test ether to the deployer account for next step

┌──(kali㉿kali)-[~]
└─$ nc 42.192.224.58 20000
I hide my secret number in the contract. I think it's very easy for you!
Your goal is to make isSolved() event emitted!

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 2
[-] input your token: v4.local.nbM5dLZYtdJKurG2EDlWp8S8UjYT3YEfLVjxNZdsv57BdwQd396f9SpwDuxdO7rxDVsOp1a3TUbSqiUX9iejsjU_s-PxE1pi0h
kxhs6rb_WUGTuwBwgIR3iCNF1HR43R3IRjCFEteJysVNh1Wzz4KFwbokRQDJFb3r0YGfRD5Mu2zw
[+] contract address: 0×1dd0A0bCAc40A3cd513C2571De97937d2BFCc082
[+] transaction hash: 0×eba563db9680df5f04d52d395b6782baf96acfab11bb538b424d2b4b152896d7
```

这时候用 remix 编译一下题目合约，然后使用**At Address**功能，直接和链上已部署的合约交互

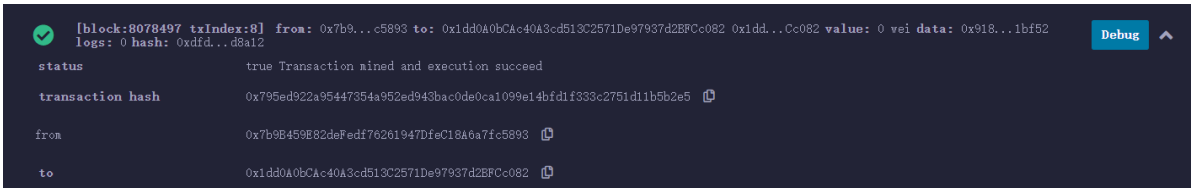
记得要选择测试网环境，从metamask钱包获取



然后直接交互，获取 secrets[0] 的值，是114514



然后guess，metamask会跳出交易确认，看到remix跳出这个说明已执行完毕



然后nc连接，选择3，把 transaction hash 发过去就能获取flag了

```
┌──(kali㉿kali)-[~]
└─$ nc 42.192.224.58 20000
I hide my secret number in the contract. I think it's very easy for you!
Your goal is to make isSolved() event emitted!

[1] - Create an account which will be used to deploy the challenge contract
[2] - Deploy the challenge contract using your generated account
[3] - Get your flag once you meet the requirement
[4] - Show the contract source code
[-] input your choice: 3
[-] input your token: v4.local.nbM5dLZYtdJKurG2EDlWp8S8UjYT3YEfLVjxNZdsv57BdwQd396f9SpwDuxdO7rxDVsOp1a3TUbSqiUX9iejsjU_s-PxE1pi0h
kxhs6rb_WUGTuwBwgIR3iCNF1HR43R3IRjCFEteJysVNh1Wzz4KFwbokRQDJFb3r0YGfRD5Mu2zw
[-] input tx hash that emitted isSolved event: 0x795ed922a95447354a952ed943bac0de0ca1099e14bfd1f333c2751d11b5b2e5
[+] flag: NUAACTF{N3ver_Put_y0ur_s3crets_1n_th3_pub1ic}
```

> 这里其实还有另一个比较取巧的方法，直接看合约创建的那一笔交易，会有constructor构造函数输入参数的信息
>
> 
>
>
> input data的最后是0x1bf52，就是114514

SecretNumber 2

> description:
>
> 众所周知，私有变量是不可以被外部读取的，所以藏在里面一定很安全吧
>
> nc 42.192.224.58 20001

源码

```
pragma solidity 0.8.7;

contract HiddenSecret2 {
        uint8 nonce;
        uint256[] private secrets;

        event isSolved();

    constructor(uint256 _secret) public {
        nonce = 0;

 secrets.push(uint256(keccak256(abi.encodePacked(_secret+block.number))));
    }

    function guess(uint256 _answer) public {
        require(tx.origin != msg.sender);
        nonce = nonce + 1;
        require( nonce <= 3 );
        if( keccak256(abi.encodePacked(secrets[0])) ==
keccak256(abi.encodePacked(_answer)) ) {
                emit isSolved();
        }
    }
}
```

和上一题的区别就是，这里的secrets数组是private私有的，不能直接通过remix上交互来获取里面的值。

顺便还在constructor构造函数里把secrets[0]复杂化了一下，防止上一题的取巧做法出现（当然也是能做，但是比较费劲

还有guess里多了一个 `require(tx.origin != msg.sender);` 条件

主要是考察合约私有变量的读取，需要web3js或者web3py的一点知识，以及对以太坊变量存储规则的一些了解。

虽说是私有的，但是链上所有数据都是公开的，仍然是可以读取的。

solidity变量存储规则可以看看文档: 状态变量在储存中的布局 — Solidity中文文档)

> 数据的存储方式是从位置 `0` 开始连续放置在 存储槽slot 中，对于每个变量，根据其类型确定字节大小
>
> 假设 动态数组根据上述存储规则最终可确定某个位置 `p`，数组的元素会从 `keccak256(p)` 开始

比如这题，secrets变量在第1个，占用slot1

> 第0个变量是nonce，nonce只有1字节，虽然没有把slot0占满，但是下一个变量是数组，数组本身要占32字节来保存数组长度，因此secrets占用slot1

那么元素 secrets[i] 的位置在 slot(keccak256(1)+i) 处，知道了数据的位置就可以进行数据读取

先是和上一题一样，正常转钱等待部署，拿到合约地址

然后读取secrets[0]的位置，即 keccak256(1)+0 的存储槽

这里给出一个web3py读取数据的脚本

```
from web3 import Web3

w3 = Web3(Web3.HTTPProvider('https://goerli.infura.io/v3/xxx'))
# infura提供公开以太坊和测试节点，可以利用infura提供的api访问以太坊，快速接入以太坊
# 可以去infura.io注册一个账号，xxx填你的API KEY

address = w3.toChecksumAddress("contract address")
slot = 1
sload = w3.solidityKeccak(['uint256'], [slot]).hex()
print(sload) # sload是secrets[0]的存储位置
s = w3.eth.getStorageAt(address, sload).hex()
print(s)    # s是secrets[0]的值
```

得到secrets[0]的值之后，进行guess调用

但是需要绕过 `require(tx.origin != msg.sender);`

这里很简单，tx.origin是整笔交易链的第一发起人，msg.sender是当前交易的发送者

可以编写一个合约，通过这个合约来调用题目合约的guess函数，这样tx.origin是你的账户地址，msg.sender是你编写的合约地址

```
pragma solidity 0.8.7;

contract HiddenSecret2 {
    //...the content of the HiddenSecret2 contract
}

contract Exp {
    HiddenSecret2 instance;
    constructor() {
        instance = HiddenSecret2(0xf181781980ceac49b129847c3531fc7922f1944e);
        //the address of the HiddenSecret2 contract
```

```
    }
    function hack(uint256 _answer) public {
        instance.guess(_answer);
    }
}
```

把Exp合约用remix部署，然后调用hack函数，输入参数是之前读取出来的secrets[0]的值

最后nc连接，提交 transaction hash， 就能获取flag了